ELSEVIER

Information Sciences 141 (2002) 123–138

INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL

www.elsevier.com/locate/ins

A steganographic method based upon JPEG and quantization table modification

Chin-Chen Chang [a,*], Tung-Shou Chen [b,1], Lou-Zo Chung [a]

[a] *Department of Computer Science and Information Engineering, National Chung Cheng University, Chiayi 62107, Taiwan, ROC*

[b] *Department of Computer Science and Information Management, Providence University, Taichung 433, Taiwan, ROC*

Received 17 November 1999; received in revised form 23 December 2000; accepted 8 May 2001

Abstract

In this paper, a novel steganographic method based on joint photographic expert-group (JPEG) is proposed. The proposed method modifies the quantization table first. Next, the secret message is hidden in the cover-image with its middle-frequency of the quantized DCT coefficients modified. Finally, a JPEG stego-image is generated. JPEG is a standard image and popularly used in Internet. The stego-image will not be suspected if we could apply a JPEG image to data hiding. We compare our method with a JPEG hiding-tool Jpeg–Jsteg. From the experimental results, we obtain that the proposed method has a larger message capacity than Jpeg–Jsteg, and the quality of the stego-images of the proposed method is acceptable. Besides, our method has the same security level as Jpeg–Jsteg. © 2002 Elsevier Science Inc. All rights reserved.

Keywords: JPEG; Steganography; Data hiding; Jpeg–Jsteg; DCT

* Corresponding author. Fax: +886-5-2720-859.

E-mail addresses: ccc@cs.ccu.edu.tw (C.-C. Chang), ts.chen@taiwan.com (T.-S. Chen), lzc87@cs.ccu.edu.tw (L.-Z. Chung).

[1] Tel.: +886-4-6328001x3408; fax: 886-4-6324045.

PII: S0020-0255(01)00194-3

1. Introduction

Nowadays, there are many digital multimedia transmissions on the network. There could be some important data that need to be protected during transmission. Therefore, how to protect the secret messages during transmission becomes an important research issue [2]. Steganography [9] provides a kind of data hiding method that conceals the existence of the secret messages in the media. We select an image as the media to hide the secret message in. This image is called cover-image. The cover-image with the secret message embedded in it is called the stego-image. For an image, the image quality refers to the quality of the stego-image, and the message capacity concerns the question of how many secret messages can be embedded in the stego-image. If a stego-image has good image quality, it can avoid being suspected during transmission of hidden messages.

Data hiding methods for images can be categorized into two categories. They are spatial-domain methods and frequency-domain ones. In the spatial-domain [1,5–7], the secret messages are embedded in the image pixels directly. In the frequency-domain [3–7], however, the secret image is first transformed to frequency-domain, and then the messages are embedded in the transformed coefficients.

Joint photographic expert-group (JPEG) [8] is a famous file for images. It applies the discrete cosine transformer (DCT) to image content transformation. DCT is a widely used tool for frequency transformation. If we apply JPEG images to data hiding, the stego-image will not easily draw attention of suspect. There is a JPEG hiding-tool Jpeg–Jsteg [10]. In the Jpeg–Jsteg embedding method, secret messages are embedded in the least signification bits (LSB) of the quantized DCT coefficients whose values are not 0, 1, or −1. The main drawback of Jpeg–Jsteg is less message capacity. This is because, after the DCT transformation and quantization of JPEG, the coefficients are almost all zero and cannot hide messages according to the definition of Jpeg–Jsteg.

To improve the message capacity of Jpeg–Jsteg, a new data hiding method based on JPEG and quantization table modification is proposed. Our method was inspired by Hsu and Wu's approach [4]. Obviously, their scheme is aimed for the image copyright protection against illegal use by attackers while ours is aimed for security hiding image in a plain image. Again, we shall emphasize here that our method, the attacker is unable to retrieve secret messages form the plain image in which they were hidden. So he does not know the contents of secret image unless he has the ability to decipher the plain image. As for Hsu and Wu's approach, they propose a very robust watermarking technique in which the attacker is unable to remove or severely destroy the hidden watermarks even he knows what the contents of watermarks. From the above statements, we know that Hsu and Wu's approach, the contents of the hidden images (watermarks) are not secret data while in ours, they are. This method

will modify the quantization table of JPEG first and then embed the secret messages in the least two-signification bit (LTSB) of the quantized DCT coefficients that are located in the middle-frequency part. Our method generates a JPEG stego-image finally. Note that the secret messages are embedded in the quantized DCT coefficients in our method. Suppose the quantization table of JPEG is not changed in our method. The modification of the quantized DCT coefficients will be amplified if it is dequantized. Then there will be quite some distortion in the reconstructed image. Therefore, the quantization table of JPEG must be modified in our method.

The rest of this paper is organized as follows. Section 2 will review of Jpeg–Jsteg. Section 3 will propose our data hiding method in JPEG. In Section 4, some experimental results and security analyses will be listed and discussed. Finally, the conclusions will be presented in Section 5.

2. Jpeg–Jsteg

Jpeg–Jsteg [4] is a famous hiding-tool based on JPEG. In Jpeg–Jsteg, the secret messages are embedded in LSB of quantized DCT coefficients whose values are not 0, 1, or −1. Its execution steps are described briefly as follows.

First, JPEG partitions a cover-image into non-overlapping blocks of 8×8 pixels, and then it uses DCT to transform each block into DCT coefficients. The results of the DCT coefficients are scaled according to a quantization table. The standard quantization table is listed in Fig. 1, which is a matrix that contains 64 coefficients. The user can adjust those 64 coefficients. Next, Jpeg–Jsteg uses an encryption algorithm to protect the message. A message after encrypting is called secret message $\bar{S} = \{s_0, s_1, s_2, s_3, \ldots, s_n\}$, where s_i is a secret bit. After the above steps, Jpeg–Jsteg embeds s_i into LSB of quantize DCT coefficients whose values are not 0, 1, or −1. The embedding sequence employed in Jpeg–Jsteg is in the zigzag scan order, which is listed in Fig. 2.

$$\begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

Fig. 1. Standard quantization table.

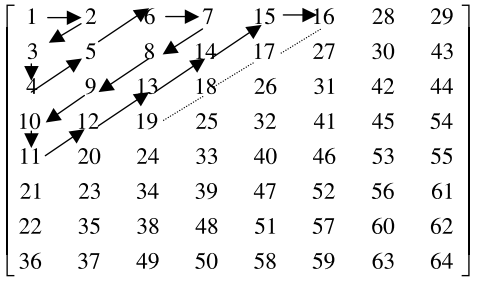


Fig. 2. Zigzag scans order.

After embedding the secret message in each block, Jpeg–Jsteg uses Huffman coding, Run-Length coding, and DPCM of JPEG entropy coding to compress each block. Finally, Jpeg–Jsteg obtains a JPEG stego-image.

For example, Fig. 3(a) shows a block of 8×8 pixels in an original cover-image. JPEG uses DCT to transform the block into DCT coefficients. The result of the DCT coefficients of the block is listed in Fig. 3(b). After DCT

$$\begin{bmatrix} 139 & 144 & 149 & 153 & 155 & 155 & 155 & 155 \\ 144 & 151 & 153 & 156 & 159 & 156 & 156 & 156 \\ 150 & 155 & 160 & 163 & 158 & 156 & 156 & 156 \\ 159 & 161 & 162 & 160 & 160 & 159 & 159 & 159 \\ 159 & 160 & 161 & 162 & 162 & 155 & 155 & 155 \\ 161 & 161 & 161 & 161 & 160 & 157 & 157 & 157 \\ 162 & 162 & 161 & 163 & 162 & 157 & 157 & 157 \\ 162 & 162 & 161 & 161 & 163 & 158 & 158 & 158 \end{bmatrix}$$

(a)

$$\begin{bmatrix} 1260 & -1 & -12 & -5 & 2 & -2 & -3 & 1 \\ -23 & -17 & -6 & -3 & -3 & 0 & 0 & -1 \\ -11 & -9 & -2 & 2 & 0 & -1 & -1 & 0 \\ -7 & -2 & 0 & 1 & 1 & 0 & 0 & 0 \\ -1 & -1 & 1 & 2 & 0 & -1 & 1 & 1 \\ 2 & 0 & 2 & 0 & -1 & 1 & 1 & -1 \\ -1 & 0 & 0 & -1 & 0 & 2 & 1 & -1 \\ -3 & 2 & -4 & -2 & 2 & 1 & -1 & 0 \end{bmatrix}$$

(b)

Fig. 3. An example of Jpeg–Jsteg. (a) A block of 8×8 pixel values. (b) The DCT coefficients. (c) The quantized DCT coefficients. (d) The result of the coefficients after the embedding step.

$$\begin{bmatrix} 79 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(c)

$$\begin{bmatrix} 78 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -3 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(d)

Fig. 3. (*Continued*).

transformation, JPEG uses the standard quantization table to quantize the DCT coefficients. The result of the quantized DCT coefficients is listed in Fig. 3(c). Jpeg–Jsteg embeds the secret messages in LSB of the quantized DCT coefficients whose values are not 0, 1, or −1. In this block, only two coefficients 79 and −2 can embed the secret message. Assume the secret message is 01_2. Then the result of this block after embedding will be listed in Fig. 3(d).

The message capacity of Jpeg–Jsteg is limited. If there are many quantified coefficients equal to 0, 1, or −1, then the message capacity of Jpeg–Jsteg will be decreased. Besides, in DCT transformation, most important coefficients are located around the low-frequency part. Jpeg–Jsteg modifies the quantized DCT coefficients right in the low-frequency part. Therefore, the image quality of Jpeg–Jsteg is degraded, especially when the cover-image undergoes a high compression ratio.

3. The proposed method

In steganography, the message capacity and the image quality of a stego-image are two important criteria. Unfortunately, the message capacity of

Jpeg–Jsteg does not seem to be quite a satisfying one. To make it better, we propose a new data hiding method based on JPEG.

3.1. The embedding procedure

In frequency-domain, JPEG is the most popular image standard in Internet. Suppose we apply a JPEG image to data hiding so that the stego-image will not be suspected by anyone. Our embedding procedure contains five phases. They are message encryption, image preprocessing, secret message embedding, JPEG entropy coding, and JPEG stego-image generation.

We apply a data encryption method with a secret key k to encrypt the message M in the first phase. Here the message M can be a text, a video, or an image, etc. The encrypted result is called the secret message $\bar{S} = \{s_1, s_2, s_3, \ldots, s_m\}$, where s_i is a secret bit containing 0 or 1 and m is the length of $\bar{S}$.

In the second phase, the proposed method uses the JPEG image preprocessing method upon the cover-image. We partition a cover-image O into non-overlapping blocks of 8×8 pixels, and then we use DCT to transform each block into DCT coefficients. The DCT coefficients are then scaled with a quantization table. The quantization table is listed in Fig. 4. This table is notably different from the quantization table of JPEG. This is because our secret message will be embedded in the middle-frequency part of the quantized DCT coefficients. If we use the same quantization table as Fig. 1 to quantize and dequantize DCT coefficients, then the quantized DCT coefficients will be amplified and then the reconstructed image will undergo quite some distortion. Therefore, the quantization table of JPEG needs a modification. In Fig. 4, there are 26 coefficients located in the middle part that are set to be one. They are $p[0,4]$, $p[0,5]$, $p[0,6]$, $p[0,7]$, $p[1,3]$, $p[1,4]$, $p[1,5]$, $p[1,6]$, $p[2,2]$, $p[2,3]$, $p[2,4]$, $p[2,5]$, $p[3,1]$, $p[3,2]$, $p[3,3]$, $p[3,4]$, $p[4,0]$, $p[4,1]$, $p[4,2]$, $p[4,3]$, $p[5,0]$, $p[5,1]$,

$$\begin{bmatrix} 16 & 11 & 10 & 16 & 1 & 1 & 1 & 1 \\ 12 & 12 & 14 & 1 & 1 & 1 & 1 & 55 \\ 14 & 13 & 1 & 1 & 1 & 1 & 69 & 56 \\ 14 & 1 & 1 & 1 & 1 & 87 & 80 & 62 \\ 1 & 1 & 1 & 1 & 68 & 109 & 103 & 77 \\ 1 & 1 & 1 & 64 & 81 & 104 & 113 & 92 \\ 1 & 1 & 78 & 87 & 103 & 121 & 120 & 101 \\ 1 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

Fig. 4. The modified quantization table.

$p[5,2]$, $p[6,0]$, $p[6,1]$, and $p[7,0]$. Here p is the modified quantization table and $p[a,b]$ is the value of the ath row and bth column element of p. Based on this quantization table, the secret messages can be reserved and the reconstructed image will not be too much distorted.

In the third phase, the secret message $\bar{S}$ will be embedded in the middle-frequency part of the quantized DCT coefficients for each block O_i. Let C_i be the modified quantized DCT coefficients, and let $C_i[a,b]$ denote the value of the ath row and bth column coefficients of the block C_i. Each coefficient in the middle-frequency part will embed two secret bits to increase the message load of the stego-image. Our method embeds s_1 and s_2 into LTSB of $C_1[0,4]$, s_3 and s_4 into LTSB of $C_1[0,5]$, s_5 and s_6 into LTSB of $C_1[0,6]$, s_7 and s_8 into LTSB of $C_1[0,7]$, s_9 and s_{10} into LTSB of C_1 [1,3], and so on. The embedding order is listed in Fig. 5.

We compress the embedded C_i in the fourth phase. After the secret message is embedded in each block, we employ the JPEG entropy coding (that contains Huffman coding, Run-Length coding, and DPCM) to compress each block. For each block after the entropy coding, we obtain a JPEG file that contains a quantization table p and some compressed data. They are our stego-image that satisfies the JPEG standard.

In the fifth phase, we output the JPEG stego-image E and transfer it to the receiver. The block diagram of the embedding procedure is shown in Fig. 6.

[*Algorithm of the embedding procedure*]
Input: A cover-image O, message M, and a secret key k.
Output: A stego-image E.
Step 1: Input a cover-image O. Suppose its size is $N \times N$ pixels. Partition the cover-image into non-overlapping blocks $\{O_1, O_2, O_3, \ldots, O_{N/8\times N/8}\}$. Each O_i contains 8×8 pixels.
Step 2: Use DCT to transform each block O_i into DCT coefficient matrix F_i, where $F_i[a,b] = \mathrm{DCT}(O_i[a,b])$, where $1 \leqq a, b \leqq 8$ and $O_i[a,b]$ is the pixel value in O_i.

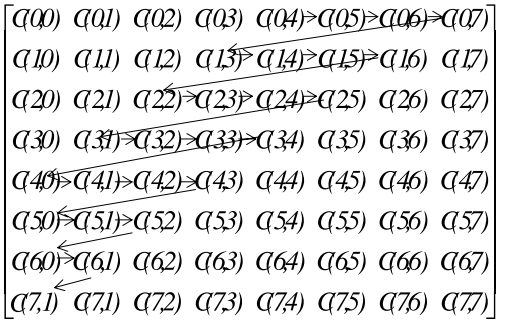


Fig. 5. Embedding sequence.

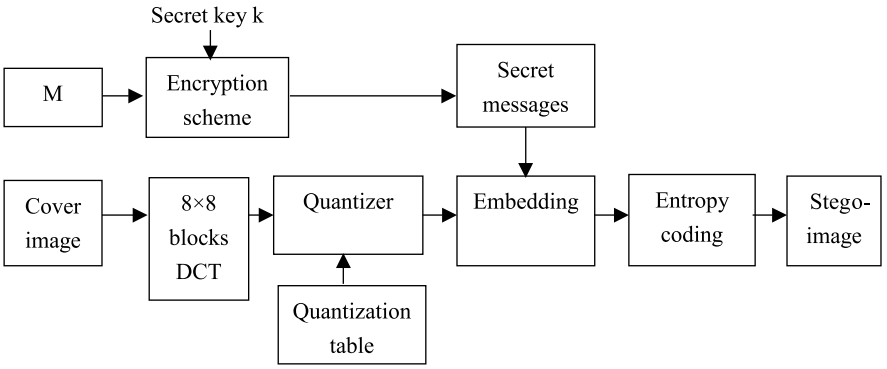


Fig. 6. The block diagram of the embedding procedure.

Step 3: Use modified quantization table p to quantize each F_i. The result can be represented as $C_i[a,b] = \text{truncate}\,(F_i[a,b]/P[a,b])$.
Step 4: Apply an encryption method with secret key k to encrypt the message M. The resulted message is $\bar{S} = \{s_1, s_2, s_3, \ldots, s_m\}$, where s_i is a secret bit and m is the length of $\bar{S}$.
Step 5: Select $C_i[a,b]$ to hide $\bar{S}$ respectively, where $[a,b]$ equals to [0,4], [0,5], [0,6], [0,7], [1,3], [1,4], [1,5], [1,6], [2,2], [2,3], [2,4], [2,5], [3,1], [3,2], [3,3], [3,4], [4,0], [4,1], [4,2], [4,3], [5,0], [5,1], [5,2], [6,0], [6,1], and [7,0], respectively. Each C_i $[a,b]$ embeds two secret bits into it.
Step 6: Apply JPEG entropy coding, which contains Huffman coding, Run-Length coding, and DPCM, to compress each block C_i. Collect the above results and generate a JPEG file E that contains the quantization table p and all the compressed data.
Step 7: Transfer the secret key k and the JPEG stego-image E to the receiver.

Consider the same example illustrated in Fig. 3. Assume that the original message is $101101101101100011000000_2$. Before embedding the message in the cover-image, we use an encryption method with secret key k to encrypt the message. Suppose the encrypted message is $100111001110010010010000_2$. Fig. 3(a) lists a block of 8×8 pixels in the original cover-image. We use DCT to transform the block into DCT coefficients. The DCT results of this block are listed in Fig. 3(b). After the DCT transformation, we use the quantization table p to quantize the DCT coefficients. The results of the quantized coefficients are listed in Fig. 7. The proposed method embeds the secret message in the middle-frequency part of the quantized DCT coefficients. After the embedding step is done, the result of this block is shown in Fig. 8. In this example, it is obvious that the message capacity of our method is larger than that of the Jpeg–Jsteg's.

$$\begin{bmatrix} 79 & 0 & -1 & 0 & 2 & -2 & -3 & 1 \\ -2 & -1 & 0 & -3 & -3 & 0 & 0 & 0 \\ -1 & -1 & -2 & 2 & 0 & -1 & 0 & 0 \\ 0 & -2 & 0 & 1 & 1 & 0 & 0 & 0 \\ -1 & -1 & 1 & 2 & 0 & 0 & 0 & 0 \\ 2 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Fig. 7. The results of the quantizer.

$$\begin{bmatrix} 79 & 0 & -1 & 0 & 2 & -1 & -3 & 0 \\ -2 & -1 & 0 & -3 & -2 & 1 & 0 & 0 \\ -1 & -1 & -2 & 1 & 0 & 0 & 0 & 0 \\ 0 & -2 & 0 & 1 & 1 & 0 & 0 & 0 \\ -1 & -1 & 1 & 2 & 0 & 0 & 0 & 0 \\ 2 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Fig. 8. The results of the block after embedding the message.

3.2. The extracting procedure

In our method, the extracting procedure contains three phases. The first phase is the JPEG entropy decoding, the second phase is the secret message extracting, and the last phase is the decryption of the secret message.

After receiving the JPEG stego-image and the private key k from the encoder, we use the inverse entropy coding to decode the compressed data in the JPEG file. The JPEG stego-image file contains a quantization table and all the compressed data. The inverse JPEG entropy coding contains Huffman decoding, Run-Length decoding, and DPCM decoding. Each block C_i can be reconstructed after all the compressed data are decoded.

Next, we extract the secret bits from LTSB of the 26 middle-frequency coefficients $C_i[0,4]$, $C_i[0,5]$, $C_i[0,6]$, $C_i[0,7]$, $C_i[1,3]$, $C_i[1,4]$, $C_i[1,5]$, $C_i[1,6]$, $C_i[2,2]$, $C_i[2,3]$, $C_i[2,4]$, $C_i[2,5]$, $C_i[3,1]$, $C_i[3,2]$, $C_i[3,3]$, $C_i[3,4]$, $C_i[4,0]$, $C_i[4,1]$, $C_i[4,2]$, $C_i[4,3]$, $C_i[5,0]$, $C_i[5,1]$, $C_i[5,2]$, $C_i[6,0]$, $C_i[6,1]$, and $C_i[7,0]$, where $1 \leqq i \leqq (N/8 \times N/8)$. Then we collect those bits to regenerate the secret message $\bar{S}$.

The proposed method then imports the secret key k to the decryption method to decrypt the secret message. Finally, we achieve the extraction of the original message M. The block extracting diagram is listed in Fig. 9.

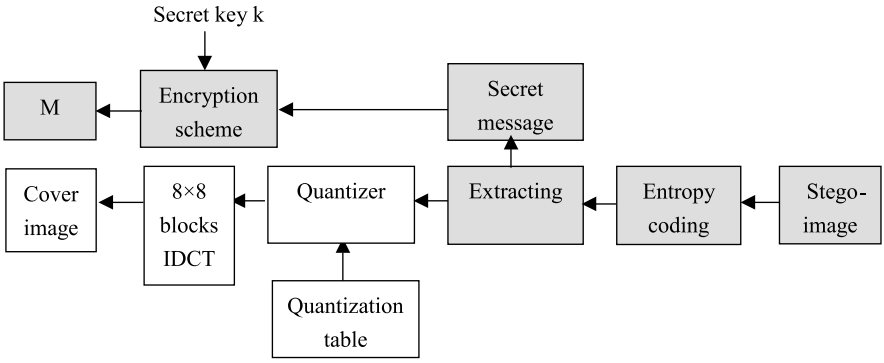


Fig. 9. The block diagram of the extracting procedure.

[*Algorithm of the extracting procedure*]
Input: A stego-image E and a secret key k.
Output: The hidden message M.
Step 1: Use the first phase of JPEG decoding procedure to decompression the JPEG file. The decoding procedure contains Huffman decoding, Run-Length decoding, and DPCM decoding.
Step 2: Extract the secret message $\bar{S}$ from LTSB of the 26 middle-frequency coefficients $C_i[0,4]$, $C_i[0,5]$, $C_i[0,6]$, $C_i[0,7]$, $C_i[1,3]$, $C_i[1,4]$, $C_i[1,5]$, $C_i[1,6]$, $C_i[2,2]$, $C_i[2,3]$, $C_i[2,4]$, $C_i[2,5]$, $C_i[3,1]$, $C_i[3,2]$, $C_i[3,3]$, $C_i[3,4]$, $C_i[4,0]$, $C_i[4,1]$, $C_i[4,2]$, $C_i[4,3]$, $C_i[5,0]$, $C_i[5,1]$, $C_i[5,2]$, $C_i[6,0]$, $C_i[6,1]$, and $C_i[7,0]$, where $1 \leqq i \leqq (N/8 \times N/8)$. Collect those secret bits to regenerate the secret message $\bar{S}$.*Step* 3: Import secret key k to the decryption method to decrypt the secret message $\bar{S}$ and reconstruct the original message M.

4. Discussions and comparisons

We conduct some experiments to show the flexibility of our approach. Our experiments are executed on SUN SPARC 10. Four gray-level images Lena, Baboon, Girl, and Boat, each of 256×256 pixels are used as the cover-images. We employ the pseudo random number generator (PRNG) of GCC to generate the secret message. The peak signal to noise rate (PSNR) is used in this paper to evaluate the image quality. The PSNR of a gray-level image is defined as:

$$\mathrm{PSNR} = 10 \log_{10} \frac{255^2}{\mathrm{MSE}} \mathrm{dB}. \tag{1}$$

The mean square error (MSE) for an $N \times N$ gray-level image is defined as follows:

$$\mathrm{MSE} = \left(\frac{1}{N}\right)^2 \sum_{i=1}^{N} \sum_{j=1}^{N} (x_{ij} - \bar{x}_{ij})^2. \tag{2}$$

Here x_{ij} denotes the original pixel value, and $\bar{x}_{ij}$ denotes the decoded pixel value.

Tables 1 and 2 show the stego-image sizes (KByte) and message capacity (bits) of our proposed method and Jpeg–Jsteg. For the same stego-image size, the message capacity of our method allows more messages into the cover-image than Jpeg–Jsteg.

A block can embed 52 secret bits into it in our method, and thus a cover-image of 256×256 pixels can embed $52 \times (256 \times 256)/(8 \times 8) = 53\,248$ secret bits into it. In the Jpeg–Jsteg method, however, the message capacity can be inferred from the number of the quantized DCT coefficients whose values are not 0, 1, or -1. Because the DCT coefficients after the quantization are almost all zeros, the message capacity of Jpeg–Jsteg is very much limited.

However, the stego-image size of the proposed image is quite restricted. It cannot be adjusted freely based on the choices of quantization tables, like what we can do with JPEG. This is because, in our method, we set the coefficients to be the ones in the middle part of the quantization table. Thus the quantized DCT coefficients in the middle-frequency part will not be zero even if we choose another quantization table to quantize the DCT coefficients. Thus our method can only compress the downright part of the quantized DCT coefficients.

Moverover, based on the same compression rate, the stego-image quality of Jpeg–Jsteg is better than that of our method. This phenomenon is shown in Table 3. This can be intuitively inferred, since Jpeg–Jsteg embeds fewer messages in its stego-image.

Table 1
The stego-image sizes (KBytes) of the proposed method and Jpeg–Jsteg

Methods/Images	Lena	Baboon	Boat	Girl
Original image	64	64	64	64
Proposed method	25	26	27	22
Jpeg–Jsteg	25	26	26	21

Table 2
Capacity (bits) of the proposed method and Jpeg–Jsteg

Methods/Images	Lena	Baboon	Boat	Girl
Proposed method	53 248	53 248	53 248	53 248
Jpeg–Jsteg	17 798	21 142	20 013	14 751

Table 3
The image quality (PSNR) of the proposed method and Jpeg–Jsteg

Methods/Images	Lena	Baboon	Boat	Girl
Proposed method	34.84	27.63	33.29	39.14
Jpeg–Jsteg	39.10	30.38	37.08	41.60

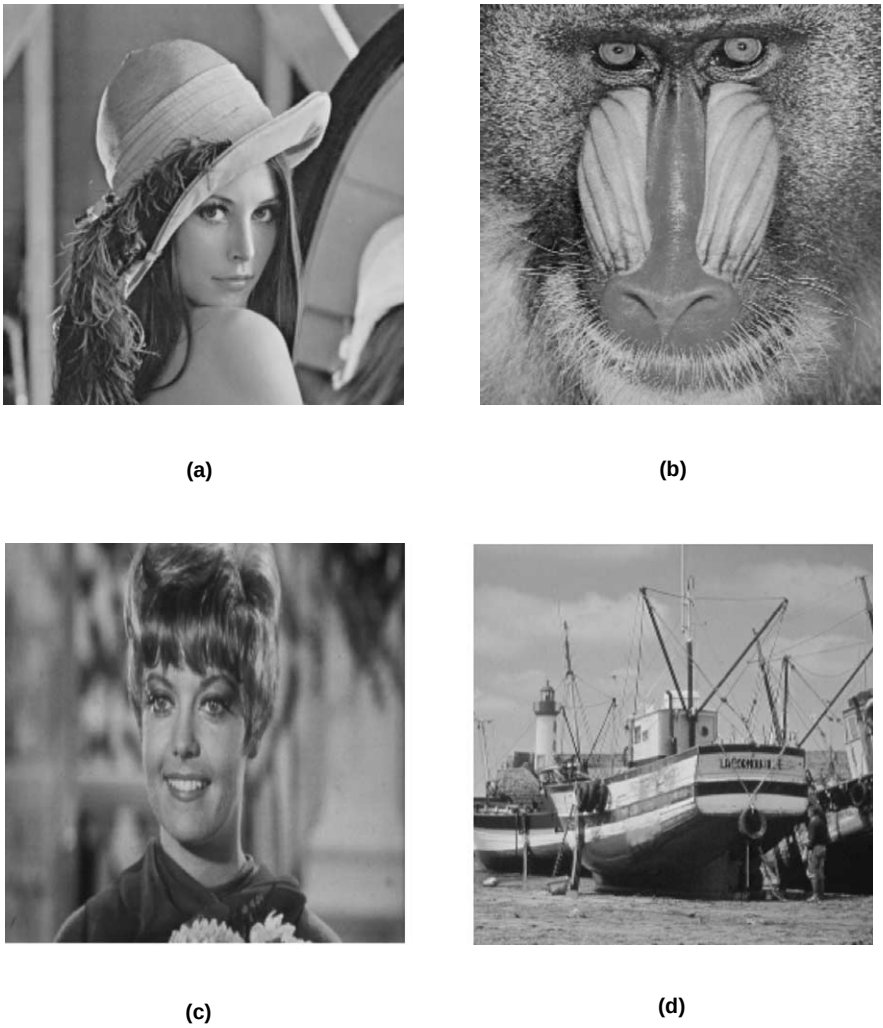


Fig. 10. The original images of (a) Lena, (b) Baboon, (c) Girl and (d) Boat.

(a.1) Stego-image Lena (PSNR=39.10 dB) of Jpeg-Jsteg

(a.2) Stego-image Lena (PSNR=34.84 dB) of the proposed method

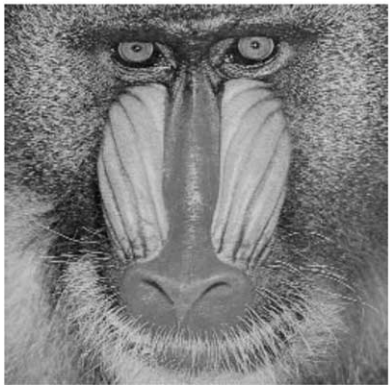
(b.1) Stego-image Boat (PSNR=37.08 dB) of Jpeg-Jsteg

(b.2) Stego-image Lena (PSNR=33.29 dB) of the proposed method

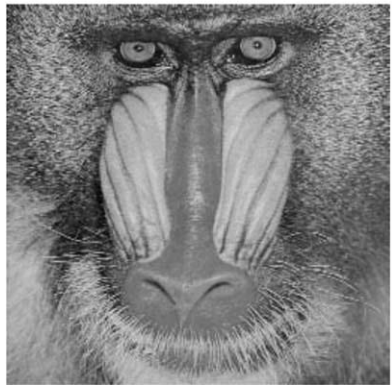
Fig. 11. The results of stego-images of our scheme and Jpeg–Jsteg for comparison.

To compare the visual quality between our method and Jpeg–Jsteg, we use the four original cover-images listed in Fig. 10. In Fig. 11, we show the stego-images of our method and Jpeg–Jsteg. We observe that the stego-images of our method are almost identical with those of Jpeg–Jsteg and, moreover, that they are close to the original images.

There are two important criteria for steganography. They are the image quality and message capacity of the stego-image. The message capacity of our method is larger than that of Jpeg–Jsteg, and the image quality of the proposed method is acceptable.



(c.1) Stego-image Baboon (PSNR=30.38dB) of Jpeg-Jsteg



(c.2) Stego-image Baboon (PSNR=27.63dB) of the proposed method

(d.1) Stego-image Girl (PSNR=41.60 dB) of Jpeg-Jsteg

(d.2) Stego-image Girl (PSNR=39.14 dB) of the proposed method

Fig. 11. (*Continued*).

4.1. Security analyses

In steganography, image quality and message capacity are two important issues. How to get a larger message capacity and better stego-image quality are therefore important topics. Note that, the better quality the stego-image has the more secure the steganography system will be.

In our data hiding method, we modify the quantization table to help with message embedding. Our quantization table is not easily detected by anyone.

Even if the illegal user fingers out that the quantization table has been modified and extracts the message embedded in the image, she/he still cannot decrypt the secret message, because the secret message has been encrypted by the encryption method. Without the key the secret message cannot be recovered.

If the illegal user detects and knows the quantization table has been modified and decides to destroy the message embedded in the image, she/he can remove the lowest two order bits, and thus the message will be destroyed. However, Jpeg–Jsteg method does not prevent this attack. Therefore, how to protect and correct the messages is another issue, which still remains open.

In a JPEG image, the quantization table is hidden. That means the quantization table is in the JPEG file. And it is not easy for anyone to detect it.

5. Conclusions

The goal of data hiding is to avoid peeper from discovering the secret messages embedded in the cover-images. In Jpeg–Jsteg, only few messages can be embedded in the cover-image. To improve the capacity of hidden message, we propose a new steganographic method to increase the message load in every block of the stego-image while keeping the stego-image quality acceptable. In our method, the secret message is embedded in the middle-frequency part of the quantized DCT coefficients. Our experimental results show the proposed method provides acceptable image quality and a large message capacity. Moreover, based on our security analysis, we observe that the proposed method has the same camouflage and thus has the same security level as Jpeg–Jsteg. Overall, the proposed method matches the requirement of steganography with a larger message capacity than that of Jpeg–Jsteg.

References

[1] R. Anderson, F. Petitcolas, On the limits of steganography, IEEE Journal on Selected Areas in Communications 16 (4) (1998) 471–478.
[2] T.S. Chen, C.C. Chang, M.S. Hwang, A virtual image cryptosystem based upon vector quantization, IEEE Transactions on Image Processing 7 (10) (1998) 1485–1488.
[3] E. Cole, Steganography, Information System Security Paper, George Mason University.
[4] C.T. Hsu, J.L. Wu, Hidden digital watermarks in images, IEEE Transactions on Image Processing 8 (1) (1999) 58–68.
[5] N.F. Johnson, S. Jajodia, Steganalysis: the investigation of hidden information, IEEE Information Technology Conference (1998) 113–116.
[6] N.F. Johnson, S. Jajodia, Exploring steganography: seeing the unsee, IEEE Computer 31 (2) (1998) 26–34.
[7] N.F. Johnson, S. Jajodia, Steganalysis of images created using current steganography software, in: Proceedings of the Information Hiding Workshop, Portland, Oregon, USA, April, 1998.

[8] W.B. Pennebaker, J.L. Mitchell, JPEG: Still Image Data Compression Standard, Van Nostrand Reinhold, New York, 1993.
[9] B. Pfitzmann, Information hiding terminology, in: First Workshop of Information Hiding Proceedings, May 30–June 1, 1996, Cambridge, UK, Lecture Notes Computer Science, vol. 1174, Springer, Berlin, 1996.
[10] D. Upham, Jpeg–Jsteg, http://www.tiac.net/users/korejwa.