Joint Data Hiding and Compression Scheme Based on Modified BTC and Image Inpainting

Xiaolong Liu[1,2], Chia-Chen Lin[*,3], IET Fellow, Khan Muhammad[*,4], Fadi Al-Turjman[5] and Shyan-Ming Yuan[6]

[1] College of Computer and Information Science, Fujian Agriculture and Forestry University, Fuzhou, 350002, China
[2] Digital Fujian Institute of the Big Data for Agriculture and Forestry, Fujian Agriculture and Forestry University, Fuzhou, 350002, China
[3] Department of Computer Science and Information Management, Providence University, Taiwan; mhlin3@pu.edu.tw
[4] Department of Software, Sejong University, Seoul 143-747, Republic of Korea; khan.muhammad@ieee.org
[5] Antalya Bilim university, Antalya, Turkey
[6] Department of Computer Science, National Chiao Tung University, 300, Taiwan

Corresponding author: Chia-Chen Lin (e-mail: mhlin3@pu.edu.tw), Khan Muhammad (e-mail: khan.muhammad@ieee.org).

This work was supported by the fund of National Natural Science Foundation of China (Grants No. 61702102), Natural Science Foundation of Fujian Province, China (Grant No. 2018J05100), Foundation for Distinguished Young Scholars of Fujian Agriculture and Forestry University (Grant No. xjq201809), and in part by the MOST of Taiwan (Grant No. MOS108-2410-H126-021 and 107-2623-E-009-006-D).

ABSTRACT Seamlessly integrating image compression technique and secret data hiding technique into a single procedure for security and efficient data transmission is a novel research issue in modern, decentralized digital communication environments. The first joint data hiding and compression (JDHC) scheme on block truncation coding (BTC) compression domain is presented in this paper. In the compressing and embedding procedure, for the complex blocks, modified block truncation coding (MBTC) is utilized to embed secret data and compress blocks simultaneously, while further controlling the visual distortion that is caused during data embedding. For the smooth blocks, according to the current embedding bit, either image inpainting or block search order coding (BSOC) is used to embed secret data and compress blocks simultaneously with maintaining acceptable compression performance. According to the image compression codes that are provided as output, image decompression and secret bits extraction procedures can be conducted simultaneously. Experimental results indicate that outstanding compression bit rate can be achieved by the proposed JDHC scheme with satisfactory visual quality and hiding capacity.

INDEX TERMS data hiding, image compression, MBTC, BSOC, image inpainting

I. INTRODUCTION

For efficient data transmission and information communication through the new-media oriented modern digital communication environments, compression techniques have become critical issues in both academia and industry [1]. Nowadays, image compression techniques have achieved a great improvement driven by a growing demand of efficient visual media communication. Over the last three decades, several compression techniques on images, such as BTC, JPEG, and vector quantization (VQ) [2], have been presented.

JPEG is based on frequency domain, which can transform digital images into bit-streams with good compression rate, but with the cost of low compression efficiency [32]. It is because cover images are transformed by a frequency-oriented mechanism discrete cosine transformation (DCT), which would significantly decrease the compression speed. VQ is a block-based image compression technique, compression is achieved by transmitting the generated indices of codebook, instead of pixels in the spatial domain. VQ is more efficient compared with JPEG, however, it requires the help of an auxiliary codebook in compressing procedure [2]. The codebook is generated in pre-training procedure by clustering algorithm, which will cost a lot of time. BTC is also a block-based image compression technique in spatial domain [3]. It is one of the most representative technique that has been used extensively due to its simplicity and high efficiency on block based compression. BTC is considered simpler and more efficient compared with JPEG and VQ. Therefore, we will choose it as the basic compression technique in this study for better compression efficiency.

Delp and Mitchell proposed the original BTC scheme in 1979 [3] with the idea of moment preserving quantization. After coding a bitmap of each block, two quantization levels are obtained by using the standard deviation and mean pixel value. In the following years, diverse variants of conventional BTC were presented to improve the compression performance

[4-9]. Absolute Moment Block Truncation Coding (AMBTC) [4] is the most famous BTC variant, in which absolute moments are preserved for block quantization rather than standard moments. The results showed that AMBTC can provide either better image quality or improved compression performance than conventional BTC. Ramana et al. [5] then proposed an interpolative scheme by transmitting half of the bit plane in encoding phase of BTC compression. The bit rate of Ramana et al.'s scheme is 1.5 bpp, whereas the bit rates of both conventional BTC and AMBTC are 2bpp. To improve visual quality, Mathews et al. [7] proposed the MBTC scheme using a different quantizing method. In their scheme, three extremums of each block were calculated for quantizing instead of mean value and standard deviation. Consequently, compared with conventional BTC, better reconstructed image visual quality could be achieved under the bit rate of 2bpp.

Another critical issue of the information-oriented society is how to maintain the confidentiality of secret information during transmission for security and privacy protection. Traditional cryptographic algorithms encrypt the secret information into cipher text; however, the meaningless cipher text would attract the suspicion of malicious attackers. To reduce this possibility, many data hiding techniques have been presented in the past two decades [10, 11]. However, size expansion could occur after embedding secret data in cover media. To reduce the size of stego-media, many scholars have devoted their attention to studying data hiding algorithms in BTC compression domain [12-18]. Generally, in traditional data hiding schemes designed for compression domain, data hiding procedure is conducted after original image is compressed. Thus, the asynchronous design of compression and hiding modules is inefficient, and may give malicious attackers opportunity to intercept the compressed image and interfere with the data hiding process. Therefore, in order to provide secure and efficient communications, it is crucial to determine how to integrate compression and secret embedding seamlessly into a single module.

With the purpose of jointing data hind and BTC compression simultaneously, in this paper, we tried to design the first JDHC scheme on BTC compression domain. On the sender side, for the complex blocks of an image, MBTC is utilized to embed secret data and compress stego-codestream simultaneously. For the smooth blocks of the image, image inpainting or BSOC is adaptively applied to hide the secret data and perform compression simultaneously. On the other side, the receiver can easily extract secret bits and derive reconstructed image at the same time. Generally, one secret bit could be embedded in each block by the proposed scheme with maintaining very good compression bit rate. In addition, the stego-image would also keep very high visual quality, where the PSNRs of the stego-images are guaranteed to be more than 30 dB. To show the comparison results, the proposed scheme is further compared with the representative BTC compression scheme [8], BTC based data hiding scheme [15], and JDHC scheme [19].

The remaining contents are organized as following: Section II describes the basic concept of MBTC and image inpainting. The proposed JDHC scheme is illustrated in Section III. In Section IV, we reported the experimental and comparison results of our scheme with several existing schemes. Section V shows the conclusions.

II. Related work

Literature review of related data hiding schemes in compression domain is presented in this section. In addition, MBTC and image inpainting are used in our proposed scheme. Therefore, some concepts of MBTC and image inpainting are described in subsection II.B and II.C, respectively, to provide related background information.

A. Related data hiding schemes in compression domain

Since images conventionally are compressed before transmitting over the Internet to reduce the requirement of transmission bandwidth, scholars have developed diverse data hiding schemes in the compression domain. Mobasseri et al. [30] proposed the first data hiding scheme in JPEG compression image. Since only a part of JPEG code stream was utilized by encoders, they embedded the secret bit by mapping a used variable length code (VLC) to an unused VLC. Qian and Zhang [31] then proposed an improved data hiding scheme by mapping one used VLC to several unused VLCs of the same size to improve the embedding capacity and preserve the size of the file. Liu et al. [32] presented a novel data hiding scheme in JPEG bitstream by using histogram modification. Secret data are embedded by modifying the VLC values slightly according to the peak point and nearest zero point of image histogram.

In terms of data hiding schemes in VQ compression domain, Lee et al. [33] presented a data hiding scheme for VQ indices based on neighboring. A sorted codebook was first utilized to compress the cover image into an index table, and neighboring correlation of four sub-codebooks was generated to further embed secret data. To improve embedding capacity, Lin et al. [34] proposed a scheme with the combination of search order coding and state codebook mapping to conduct the data hiding in VQ index table. The size of output code stream was reduced by their scheme, and a large amount of space was available for embedding secret data.

With respect to data hiding schemes in BTC compression domain, Lin and Chang [12] presented the first data hiding algorithm based on BTC. They performed substitution algorithm on quantization levels and introduced minimum distortion operation on bitmaps to reduce the distortion caused by data embedding. Later, Chang et al. [14] proposed a reversible algorithm with arranging the order of two quantizers and using the properties of side matching. Qin et al. [15] proposed a scheme based on optimal iterative BTC with embedding secret bits in LSB layers. Lin et al. [17] then presented a high-payload scheme by exploring the redundancy of each BTC-compressed block to improve hiding

performance. They created four disjointed sets to hide data in embeddable blocks with different combinations of standard deviation and mean value. Later, Lin et al.'s scheme [17] was improved by Hong et al. [18]. Their scheme could offer higher hiding capacity and reduce the transmission size of stego-media with the utilization of interpolation technique [18].

Generally, in traditional data hiding schemes in compression mentioned above, data hiding procedure is conducted after original image is compressed. Thus, the asynchronous design of compression and hiding modules is inefficient, and may give malicious attackers opportunity to intercept the compressed image and interfere with the data hiding process. In order to provide secure and efficient communications, Qin et al. [19] presented the first joint data hiding and compression (JDHC) technique, which was based on Side-Match Vector Quantization (SMVQ) compression. In their scheme, image blocks are compressed and secret data are adaptively embedded by conducting image inpainting or SMVQ. Recently, several JDHC schemes [20-22] which can embed secret data and compress the image simultaneously have been developed. However, all of the proposed JDHC schemes were only focused on the VQ compression domain. Since BTC is considered simpler and more efficient compared with VQ, we will choose it as the basic compression technique to propose the first jointing data hind and BTC compression scheme in this paper.

B. Modified Block Truncation Coding (MBTC)

Mathews et al. [7] proposed MBTC compression scheme, which is a variant of BTC by using Max-min quantizers to improve the visual quality of restored image. At the beginning, original image is separated into several $k \times k$-sized blocks without overlapping. Assuming p_{ij} represents the pixel value of position *(i, j)* in each block. The mean pixels value *m* of each block can be defined as follows：

$$m = \frac{1}{k \times k} \sum_{i=1}^{k} \sum_{j=1}^{k} p_{ij} \ . \quad (1)$$

Then, find the block truncation value *T* with the following equation.

$$T = \frac{max + m + min}{3} \ , \quad (2)$$

where *min* represents the minimum value of each block, and *max* represents the maximum value. After that, each block's binary bitmap *b* will be obtained by denoting the position where pixel value is higher than *T* as "1", and the rest as "0". Denoting low mean of each block as *l* and high mean as *h*, they will be calculated as:

$$h = \frac{1}{n} \sum_{p_{ij} > m} p_{ij}, \quad (3)$$

$$l = \frac{1}{k \times k - n} \sum_{p_{ij} \le m} p_{ij} \ , (if \ n = k \times k, l = h = m) \quad (4)$$

where *n* is the number of "1" in bitmap *b*, i.e. the number of pixels whose value is higher than *T*. The compression code of this block will consist of *(l, h, b)*. The block can be easily restored by replacing '0' in bitmap *b* with *l*, and replacing '1' with *h* during decoding procedure.

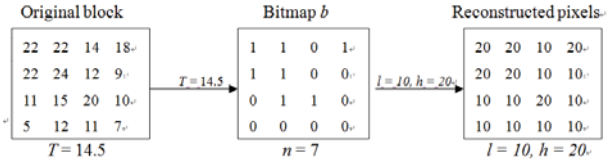


Fig. 1. MBTC compression Example

Fig.1 shows a basic compression and reconstruction example of MBTC. The block size is 4 × 4 pixels in this example. Block truncation value *T* of this block can be calculated as 14.5 according to Eq. (1). Since there are seven pixels values greater than 14.5, n equals to 7. Bitmap b will be obtained by denoting the position where pixel value is greater than 14.5 as "1", and the rest as "0". According to Eq. (3) and (4), *l* and *h* will be calculated as 20 and 10, respectively. The compression code will be (10, 20, 1101110001100000). During reconstruction, the image block can be easily restored by replacing '0' in bitmap b with 10, and replacing '1' with 20.

C. Image inpainting

Image inpainting [26, 27] is a restoring technology to repair missing data of an image. The partial differential equation (PDE) [23-25] is one the most extensively used image inpainting technology. PDE-based image inpainting schemes can achieve very good recovering performance small flaws and thin structures of an image. The PDE-based scheme was first proposed by Bertalmio et al. [23] for digital image inpainting. They used a fluid dynamics model for recovering missing regions. The PDE-based inpainting scheme was designed to smoothly diffuse pixels information from their surrounding areas along the isophote directions. The fluid dynamics model has been found to be a successful PDE-based inpainting scheme, and it has been adopted frequently in various applications. Qin et al. [19] found it suitable to apply image inpainting in the compression of the relatively smooth blocks of an image, and proposed the first JDHC scheme in SMVQ compression domain. The detailed fluid dynamics model is described in the following paragraphs.

We denoted B_x as the current processing block that needs inpainting, and *RB* as the region that includes B_x and the available neighboring regions of B_x. Assuming *B(i, j)* is the pixel's value in position *(i, j)* of *RB*. Laplacian $\Delta B(i, j)$ is used as a smoothness measure of *RB*. By using the fluid dynamics inpainting model, the details in B_x may be restored by propagating the foregone information of the neighboring regions into B_x along the isophote directions. Since, the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2935907, IEEE Access

gradient vector $\nabla B(i,j)$ gives the direction of largest change for given pixel $B(i, j)$ in RB. The 90-degree rotation $\nabla^{\perp}B(i,j)$ gives the direction of smallest change, which is the isophote directions. The isophote directions field is defined in Eq. (5):

$$\nabla^{\perp}B(i,j) = \left(-\frac{\partial}{\partial i}\boldsymbol{e_1} + \frac{\partial}{\partial j}\boldsymbol{e_2}\right) B(i,j), \tag{5}$$

where e_1 and e_2 represent unit directional vectors. After the inpainting process, $\nabla^{\perp}B(i,j)$ should be normal to the gradient of the smoothness $\Delta B(i, j)$, shows in Eq. (6):

$$\nabla[\Delta B(i,j)] \cdot \nabla^{\perp}B(i,j) = 0. \tag{6}$$

The scalar product in Eq. (6) represents projection of the smoothness changes onto the isophote directions field. By letting the change of image's pixel values of time t be equaled to the projection value in Eq. (6), the following PDE can be acquired:

$$\frac{\partial}{\partial t}B(i,j) = \nabla[\Delta B(i,j)] \cdot \nabla^{\perp}(i,j), \quad \forall(i,j) \in B_{\boldsymbol{x}}. \tag{7}$$

When the pixel values of B_x is reaching a stable state, the information diffusion of this inpainting model will be terminated. As a result, the gray values of current block B_x can be restored effectively without resulting serious blurring phenomena at the blocks.

III. The proposed scheme

A BTC-based JDHC scheme that seamlessly integrates compression and secret data hiding into a single module is presented in this section. In general, blocks in a cover image can be categorized as complex or smooth, according to the correlation among the pixels in the block. A smooth block is a block that has pixels those are highly correlated with neighboring pixels, so we say that this block is highly correlated with its neighboring blocks. However, a complex block is a block that has pixels that are highly separated from neighboring pixels, which means this block is highly separated with its neighboring regions. For the smooth blocks, image inpainting or BSOC is designed to hide secret data and simultaneously compress data by utilizing the correlation of adjoining blocks. MBTC can achieve a better decompressed image quality compared with image inpainting and BSOC, but with the cost of a higher compression ratio. Thus, with the purpose of controlling visual distortion, MBTC is utilized to hide secret bits and simultaneously compress complex blocks. The details of the proposed JDHC scheme are illustrated in the following sections.

A. Procedure for image compression and secret data embedding

A parameter R ($R=2^i$, where $i \in N$,) and a Euclidean distance threshold TH are first predetermined for block search order coding and distortion control, respectively. First, the original $M \times N$-sized original image is separated into blocks with $k \times k$ size without overlapping. Fig. 2 presents the general procedure for image compression and secret data embedding of each processed block.

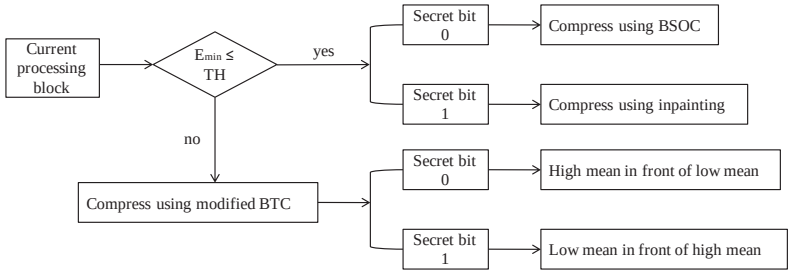


Fig. 2. Block compression and secret data embedding flowchart

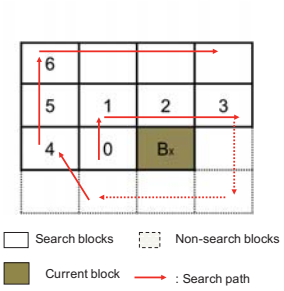


Fig. 3. Searching order of the current processing block B_x

Notably, the current processing block is denoted as B_x in the following paragraphs, and R-1 neighboring blocks of B_x are chosen according to the searching order illustrated in Fig. 3. In the example of Fig. 3, the parameter R is chosen as 8, which means that 7 neighboring blocks of B_x are selected to conduct search-order coding. The numbers ranged from 0 to 7 presented in Fig. 3 are the search order of neighboring blocks B_n (n = 0, 1…(R-2)). For each neighboring block B_n the Euclidean distance E_n is calculated using Eq. (8):

$$E_n = \sqrt{\sum_{i=1}^{k}\sum_{j=1}^{k}\left(B_n(i,j) - B_x(i,j)\right)^2}. \tag{8}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2935907, IEEE Access

Smallest Euclidean distance in E_n is selected as E_{min} of block B_x, and the corresponding neighboring block B_n with value E_{min} is selected as the most similar neighboring block of block B_x.

If E_{min} is greater than the threshold *TH*, it implies B_x has lower correlation with its neighboring area and is located in a relatively complex region. Thus, in order to control visual distortion, MBTC is utilized to process block B_x, and an indicator bit "0" is added at the front of the compression code of B_x. The secret data is embedded by arranging the order of *l* and *h* in compression code: if secret bit is "1", *l* is arranged in front of *h*; conversely, *h* is arranged in front of *l* when secret bit is "0". Thus, if secret bit is "1," B_x is encoded into *0|| l|| h|| bitmap*; otherwise, it is encoded into *0|| h|| l|| bitmap*.

If E_{min} is lower than the threshold *TH*, it implies that B_x is similar with its neighboring area and is located in a smooth region. Hence, under such condition, either inpainting or BSOC is adaptively utilized to compress and embed secret data simultaneously. Notably, in order to differentiate from blocks that are compressed by MBTC, an indicator bit "1" is added in front of the compression code of block B_x when inpainting or BSOC is used. If secret bit is "0", BSOC is utilized to conduct block compression. In other words, the binary representation of index value n of most similar neighboring block B_n will be used to represent B_x in the compression code. For example, if *R* is set as 8, the most similar neighboring block for B_x is 3, and secret bit is "0." Then, the B_x will be encoded into 1||011. However, if secret bit is "1", B_x is processed by image inpainting. The binary representation of value *R-1* will be used to represent the compressed code of B_x. For example, if *R* is set as 8 and secret bit is "1", B_x will be encoded into 1||111 according to our designed data hiding strategy and compression policy.

In general, a 4×4-sized block that is compressed using MBTC requires 33 bits compression code, i.e., 1 bit for the indicator "0", 8 bits for *h*, 8 bits for *l*, and 16 bits for bitmap *b*. A 4×4-sized block compressed using BSOC requires $1+\lfloor \log_2 R \rfloor$ bits to be encoded, i.e., 1 bit for the indicator "1" and $\lfloor \log_2 R \rfloor$ bits for the index value *n*. A 4×4-sized block compressed using image inpainting requires $1+\lfloor \log_2 R \rfloor$ bits to be encoded, i.e., 1 bit for the indicator "1" and $\lfloor \log_2 R \rfloor$ bits for the value *R-1*. Furthermore, one secret bit can be simultaneously hidden during block compression. The pseudo code of block compression and secret data embedding procedure is presented in Algorithm 1.

Algorithm 1: Compression

```
1.  procedure Compression(Bx, R, TH, s)
2.    min = 10000
3.    similarblock = R-1
4.    for n = 0 to R-2 do
5.      e = EuclideanDistance(Bx, B[n])
6.      if e < min
7.        min = e
8.        similarblock = n
9.      end if
10.   end for
11.   if Emin <= TH
12.     if s = 0
13.       output -> 1|| binary(n)
14.     else if s = 1
15.       output -> 1|| binary(R-1)
16.     end if
17.   else if Emin > TH
18.     if s = 0
19.       output -> 0 || highmean || lowmean || bitmap
20.     else if s = 1
21.       output -> 0 || lowmean || highmean || bitmap
22.     end if
23.   end if
24. end procedure
```

B. Image decompression and secret data extracting procedure

According to the received compression codes, the proposed image decompression and secret data extracting procedure can help the receiver to perform decompression and data extracting at the same time. Fig. 4 shows the general decompression and secret data extracting procedure.

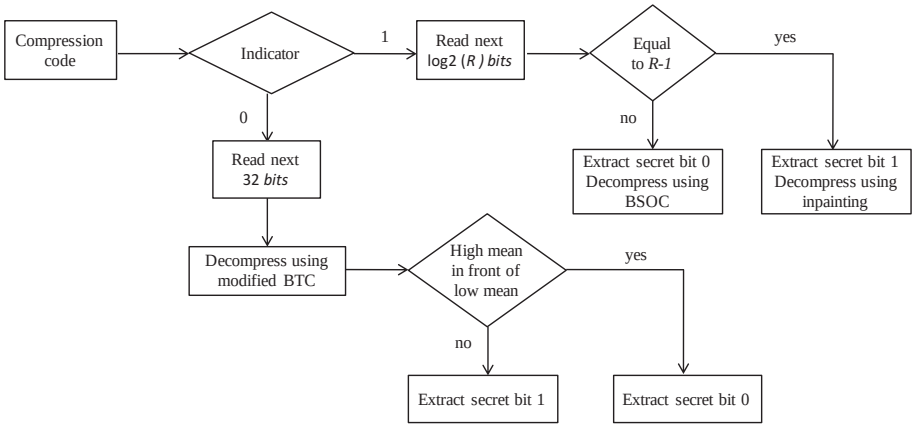


Fig. 4. Image decompression and secret data extracting flowchart

If indicator bit appears in the compression code is "0", it corresponds to a block compressed by MBTC with one secret bit. The next 32 bits that correspond to $l \parallel h \parallel bitmap$ or $h \parallel l \parallel bitmap$ are used to reconstruct the MBTC-compressed block. If the decimal value of the first 8 bits is greater than the decimal value of the next 8 bits, i.e., the high mean h appears before the low mean l, the secret bit in this section is extracted as "0." If the decimal value of the first 8 bits is smaller than the next 8 bits (i.e., l appears before h), the secret bit in this section is extracted as "1."

If indicator bit appears in the compression code is "1," it corresponds to a block compressed by image inpainting or BSOC compressed block. If the decimal value of the next $\log_2 R$ bits equals *R-1*, it indicates the secret bit to be extracted is "1", and this must be decompressed by image inpainting. Otherwise, if the decimal value of the next $\log_2 R$ bits is not *R-1*, it indicates that the secret bit is "0", and this block was encoded by BSOC during compression. In this case, the decimal value of the next $\log_2 R$ bits is the index of the most similar neighboring block of the current recovering block. The most similar neighboring block's pixel values will be used to reconstruct the current block directly according to the search-order in Fig. 3. The pseudo code of image decompression and secret data extracting procedure is presented in Algorithm 2.

Algorithm 2: Decompression

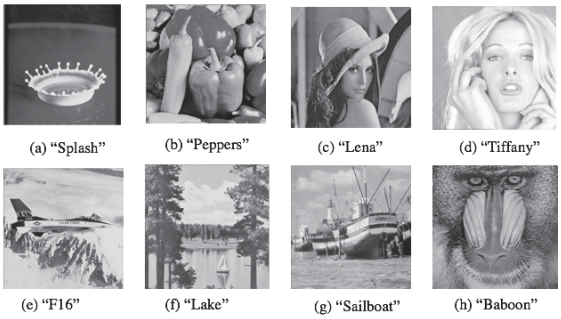
```
1.  procedure Decompression(code, R)
2.    for i = 0 to length(code)-1 do
3.      if code[i] = 0
4.        mean1 = read(code[i+1] to code[i+8])
5.        mean2 = read(code[i+9] to code[i+16])
6.        bitmap = read(code[i+17] to code[i+32])
7.        i = i + 33
8.        if mean1 > mean 2
9.          output -> secret(0)
10.         highmean = mean1
11.         lowmean = mean0
12.       else
13.         output -> secret(1)
14.         highmean = mean0
15.         lowmean = mean1
16.       end if
17.       MBTC(highmean, lowmean, bitmap)
18.     else if code[i] = 1
19.       similarblock = read(code[i+1] to code[i+log2(R)])
20.       i = i + log2(R) + 1
21.       if similarblock=R-1
22.         output -> secret(1)
23.         inpainting()
24.       else
25.         output -> secret(0)
26.         BSOC(similarblock)
27.       end if
28.     end if
29.   end for
30. end procedure
```

IV. Experimental and comparison results

Eight 512×512 grayscale images selected from USC-SIPI [35] are used in the experiments, i.e., "Splash," "Peppers," "Lena," "Tiffany," "F16," "Lake," "Sailboat," and "Baboon," shown in Fig. 5. All of the images are standard test images in compression and data hiding research areas. The test images are with different complexity, where Fig. 5(a)-(d) are images with more homogeneous areas and Fig. 5(e)-(h) are images with more high fidelity areas, respectively. Among them, "Splash" is consider to be the smoothest image and "Baboon" is the most complex image. The different kind of images are selected to evaluate the performance of the proposed scheme, and show whether the proposed scheme works well in general or not.

Since BTC is the basic compression domain in the proposed scheme, and it can be conducted with different block sizes. As we know, the larger block size would result in a higher compression rate but worse visual quality of BTC compression image [3]. In the experiments, the block size during images compression are set as 4×4 pixels, meaning that the block size $k = 4$. It is because that block size set as 4 can achieve a better tradeoff between the performance of compression rate and visual quality in BTC compression [3]. The simulation environment is equipped with Windows 7 operating system, 2.66-GHz Core2 memory, 8-GB RAM Quad system, and MATLAB 7 programming environment.

The bit rate and peak signal-to-noise ratio (PSNR) are utilized for estimating experimental performance. A lower bit rate value implies higher compression efficiency, and a higher PSNR implies higher image visual quality. The equations utilized for calculating bit rate and PSNR are Eq. (9) and Eq. (10), respectively.



(a) "Splash" (b) "Peppers" (c) "Lena" (d) "Tiffany"

(e) "F16" (f) "Lake" (g) "Sailboat" (h) "Baboon"

Fig. 5. Test images

$$bit\ rate = \frac{size\ of\ (compression\ code)}{size\ of\ (original\ image)}\ bpp, \qquad (9)$$

$$PSNR = 10 \times log_{10} \frac{(255)^2}{\frac{1}{(512 \times 512)} \sum_{i=1}^{512} \sum_{j=1}^{512} \left(x(i,j) - x'(i,j)\right)^2}\ dB, \qquad (10)$$

A. Experimental results of our proposed scheme

Obviously, since one secret bit can be embedded simultaneously in each block during compression, the hiding capacity of our scheme is a fixed number, i.e., 16,384 bits, for all test images with 512×512 size. In addition, in the proposed scheme, parameter R ($R=2^i$, where $i \in N$) for the block search order coding and the threshold TH for the distortion control are closely related to the performance results. Since the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2935907, IEEE Access

different values of *R* and *TH* affect the PSNRs of the decompressed images as well as the compression bit rate, it is necessary to determine the combination of *R* and *TH* that produces the best results for the test images.

Tables I-VIII show the compression performance of our scheme for the test images with different R and TH values. In general, the performance of the proposed scheme for smooth images is better than complex images if the same parameter is used. It is also apparent that both the PSNRs and bit rate decrease when R is fixed and TH is increasing and that both decrease when TH is fixed and R is increasing. This occurs because the number of BTC blocks that are modified is decreased when either R or TH is increasing. As described in subsection III.A, the compression codes of the blocks compressed by modified BTC are much longer than those that are compressed by BSOC or image inpainting. Thus, if the number of modified BTC blocks decreases, the bit rate is decreased. Considering the PSNRs, since the visual quality of decompressed images reconstructed by modified BTC is better than that of images reconstructed by BSOC or image inpainting, the PSNR of the decompressed image is decreased if the number of modified BTC blocks decreases. According to the observations mentioned above, users can adaptively coordinate the values of R and TH for different purposes. If users prefer for the decompressed images having better visual quality rather than higher compression efficiency, they can choose smaller values of R or TH; otherwise, larger values of R or TH should be chosen.

TABLE I
COMPRESSION RESULTS WITH DIFFERENT PARAMETERS FOR "SPLASH"

R \ TH	Factors	10	15	20	25	30	35
2	PSNR	36.34	35.01	33.59	31.47	29.96	27.67
	Bit rate	1.932	1.489	1.292	0.912	0.812	0.674
4	PSNR	36.21	34.83	32.75	30.46	28.32	27.24
	Bit rate	1.812	1.475	0.976	0.864	0.753	0.643
8	PSNR	36.18	34.76	31.89	**30.13**	27.93	26.58
	Bit rate	1.621	1.431	0.945	**0.810**	0.711	0.612
16	PSNR	36.09	34.51	31.13	29.84	27.82	26.24
	Bit rate	1.523	1.412	0.923	0.802	0.683	0.588

TABLE II
COMPRESSION RESULTS WITH DIFFERENT PARAMETERS FOR "PEPPERS"

R \ TH	Factors	10	15	20	25	30	35
2	PSNR	33.96	33.76	32.94	31.61	30.02	28.21
	Bit rate	2.037	1.839	1.572	1.374	1.215	1.076
4	PSNR	33.95	33.53	32.16	30.37	27.82	26.18
	Bit rate	2.010	1.708	1.391	1.139	0.937	0.781
8	PSNR	33.93	33.32	31.88	**30.20**	28.64	27.18
	Bit rate	1.984	1.636	1.328	**1.082**	0.888	0.741
16	PSNR	33.92	33.26	31.79	29.59	28.02	26.22
	Bit rate	1.967	1.609	1.311	1.070	0.883	0.748

TABLE III
COMPRESSION RESULTS WITH DIFFERENT PARAMETERS FOR "LENA"

R \ TH	Factors	10	15	20	25	30	35
2	PSNR	34.22	33.85	32.11	29.19	26.95	25.91
	Bit rate	1.995	1.663	1.400	1.249	1.126	1.033
4	PSNR	34.17	32.94	30.46	27.84	26.45	25.80
	Bit rate	1.875	1.282	1.024	0.894	0.791	0.716
8	PSNR	34.12	32.47	29.69	26.83	25.85	25.24
	Bit rate	1.811	1.216	0.979	0.847	0.744	0.676
16	PSNR	34.06	32.30	**30.05**	26.77	25.55	24.83
	Bit rate	1.746	1.177	**0.965**	0.842	0.749	0.684

TABLE IV
COMPRESSION RESULTS WITH DIFFERENT PARAMETERS FOR "TIFFANY"

R \ TH	Factors	10	15	20	25	30	35
2	PSNR	33.39	33.33	32.97	32.24	30.94	29.27
	Bit rate	2.022	1.940	1.705	1.490	1.346	1.230
4	PSNR	33.37	33.23	32.54	31.27	29.25	27.71
	Bit rate	2.002	1.839	1.499	1.263	1.101	0.972
8	PSNR	33.37	33.16	32.17	**30.82**	28.47	27.49
	Bit rate	1.995	1.773	1.425	**1.210**	1.047	0.921
16	PSNR	33.37	33.10	31.75	29.44	26.65	24.26
	Bit rate	1.993	1.733	1.398	1.191	1.031	0.912

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2019.2935907, IEEE Access

TABLE V
COMPRESSION RESULTS WITH DIFFERENT PARAMETERS FOR "F16"

R \ TH	Factors	10	15	20	25	30	35
2	PSNR	31.61	31.29	30.48	29.59	28.13	27.03
	Bit rate	1.838	1.548	1.283	1.100	0.976	0.899
4	PSNR	31.58	31.06	**30.15**	28.13	27.86	26.85
	Bit rate	1.746	1.430	**1.160**	0.996	0.896	0.822
8	PSNR	31.55	30.97	29.25	28.09	27.27	26.53
	Bit rate	1.702	1.371	1.108	0.968	0.878	0.806
16	PSNR	31.53	30.93	29.44	27.67	26.10	25.01
	Bit rate	1.690	1.365	1.114	0.987	0.898	0.827

TABLE VI
COMPRESSION RESULTS WITH DIFFERENT PARAMETERS FOR "LAKE"

R \ TH	Factors	10	15	20	25	30	35
2	PSNR	31.59	31.51	31.20	30.60	30.06	29.42
	Bit rate	1.987	1.875	1.727	1.568	1.407	1.255
4	PSNR	31.57	31.36	30.95	**30.05**	29.51	28.67
	Bit rate	1.925	1.740	1.568	**1.357**	1.188	1.071
8	PSNR	31.55	31.18	30.62	29.83	28.92	28.37
	Bit rate	1.861	1.690	1.505	1.294	1.142	1.036
16	PSNR	31.55	31.16	30.50	29.65	28.89	28.46

TABLE VII
COMPRESSION RESULTS WITH DIFFERENT PARAMETERS "SAILBOAT"

R \ TH	Factors	10	15	20	25	30	35
2	PSNR	31.58	31.53	31.32	30.77	**30.08**	29.06
	Bit rate	2.032	1.946	1.780	1.577	**1.395**	1.253
4	PSNR	31.57	31.50	31.09	30.18	29.17	28.13
	Bit rate	2.019	1.890	1.665	1.434	1.241	1.090
8	PSNR	31.57	31.46	30.88	29.64	28.01	26.24
	Bit rate	2.011	1.854	1.606	1.377	1.183	1.033
16	PSNR	31.56	31.43	30.68	29.57	26.86	25.08
	Bit rate	2.005	1.837	1.589	1.360	1.172	1.025

TABLE VIII
COMPRESSION RESULTS WITH DIFFERENT PARAMETERS FOR "BABOON"

R \ TH	Factors	10	15	20	25	30	35
2	PSNR	29.63	29.24	28.81	28.03	27.24	26.14
	Bit rate	2.067	1.903	1.767	1.654	1.479	1.3223
4	PSNR	29.61	**29.01**	28.62	27.23	26.35	25.33
	Bit rate	2.025	**1.801**	1.593	1.525	1.423	1.279
8	PSNR	29.61	28.95	28.53	26.93	25.26	24.23
	Bit rate	2.013	1.793	1.534	1.481	1.382	1.212
16	PSNR	29.60	28.89	28.26	26.15	24.775	22.18
	Bit rate	2.009	1.774	1.511	1.446	1.315	1.136

The threshold of a good reconstructed image's PSNR is considered to be 30 dB, since above this threshold the distortion cannot be perceived by the human eye [29]. Tables I-VII indicate that the PSNRs of test images can be 30 dB if suitable values of R and TH are chosen. The experimental results also show that the proposed scheme works well if the values of R and TH are adaptively coordinate, the PSNR can large than 30 dB with bit rate nearly 1 bpp in general. The only exception in the experiments is the results of image "Baboon," as shown in Table VIII. Since "Baboon" is consider to be the most complex image with more high fidelity areas in the standard test images, the visual quality of it is less than 30 dB. But it is worth mentioning that the acceptable PSNR with 29 dB of "Baboon" can be achieved if appropriate parameters are used.

The results of acceptable PSNR values with the lowest bit rate of each image are marked in bold in Tables I-VIII. The R and TH values that correspond to the bold numbers are chosen as the suggested (R, TH) for each test image. The suggested values are (8, 25), (8, 25), (16, 20), and (8, 25) for "Splash," "Peppers," "Lena," and "Tiffany," as show in Fig. 5(a)-(d), respectively. The suggested values are (4, 20), (4, 25), (2, 30), (4, 15) for "F16," "Lake," "Sailboat," and "Baboon," as show in Fig. 5(e)-(h), respectively. Since Fig. 5(a)-(d) are comparatively smooth images with more homogeneous areas, and Fig. 5(e)-(h) are comparatively complex images with more high fidelity areas. The experimental results indicate that the optimum parameter R would be 8 or 16 for smooth images, and 2 or 4 for complex images, respectively. In addition, there is barely noticeable difference between the optimum parameter TH for smooth and complex images, it is suggested to choose the values of TH from 20 to 30.

B. Comparison results

The results conducted by the suggested *(R, TH)* in the proposed scheme are used to compare with Xiang et al.'s scheme [8], Qin et al.'s scheme [15], and Qin et al.'s JDHC

scheme [19] in this subsection. Scheme [8] is a pure BTC based compression scheme with outstanding compression rate and visual quality, but their scheme could not hide secret data. Scheme [15] is a representative data hiding scheme in BTC compression domain. However, it is not a JDHC scheme, which means their scheme must be conducted twice to achieve image compression and secret data hiding. Scheme [19] is the representative JDHC scheme based on VQ compression domain. Even if the basis compression domain of scheme [19] is different from our scheme, we will also compare with it to show the superiority. The detailed comparison results are shown in Table VIIII.

By considering the comparison results with scheme [8], our scheme achieves better bit rate with the cost of slight visual distortion of the decompressed images. It worth to note that the PSNRs of our scheme for most test images are still higher than 30dB. The bit rates of the proposed scheme are nearly 30% less than those of scheme [8]. Furthermore, the proposed scheme is able to embed 16,348 bits along with compression in each test image, while scheme [8] is just a pure compression scheme. Thus, in comparison to the representative BTC compression scheme, the proposed scheme achieved a better bit rate and also embedded a large amount of secret in the test images. In other words, our proposed scheme has the potential for use in a greater number of potential applications than the BTC compression scheme.

TABLE VIIII
COMPARISON RESULTS OF OUR PROPOSED SCHEME, SCHEME [8], SCHEME [15], AND SCHEME [19]

Schemes	Factors	Splash	Peppers	Tiffany	Lena	F16	Lake	Sailboat	Baboon
Scheme [8]	Bit rate	1.154	1.219	1.613	1.353	1.343	1.604	1.871	1.974
	PSNR	38.11	36.52	36.76	36.61	37.35	36.56	35.00	33.21
	Capacity	0	0	0	0	0	0	0	0
Scheme [15]	Bit rate	2	2	2	2	2	2	2	2
	PSNR	36.26	34.52	34.10	34.79	32.03	32.11	32.17	29.86
	Capacity	65536	65536	65536	65536	65536	65536	65536	65536
Our scheme	Bit rate	0.810	1.082	1.210	0.965	1.160	1.357	1.395	1.801
	PSNR	30.13	30.20	30.82	30.05	30.15	30.05	30.08	29.01
	Capacity	16384	16384	16384	16384	16384	16384	16384	16384
Scheme [19]	Bit rate	0.351	0.377	0.368	0.387	0.374	0.425	0.398	0.489
	PSNR	31.84	30.35	30.54	29.85	29.31	26.67	28.42	24.56
	Capacity	11903	11295	11566	10292	11182	7247	9301	6462

In comparisons with other data hiding schemes in BTC domain [12-18], data hiding procedure in schemes [12-18] is conducted after original image is compressed, whereas the proposed scheme only requires one synchronous round to obtain both data hiding and compression objectives. The asynchronous design of compression and hiding modules in schemes [12-18] is inefficient, and may give malicious attackers opportunity to intercept the compressed image and interfere with the data hiding process. Scheme [15] is chosen as the representative BTC-based data hiding scheme to take a comprehensive comparison with our scheme. In scheme [15], secret bits are embedded by LSB algorithm and the original image is compressed by optimal iterative BTC compression. Their scheme can embed 65536 bits for a test image with good visual quality. However, scheme [15] need at least two separated procedures to compress the original image and embed secret data. Compared with it, our scheme is much efficient by joining data hiding and compression simultaneously. As shown in Table VIIII, the hiding capacity and visual quality of our proposed scheme are less than scheme [15]. It is because that scheme [15] mainly focused on data hiding performance, whereas further compression on BTC is not considered. With respect to the results of bit rate, all of our scheme's bit rate results are lower that scheme [15]. Especially for smooth test images "Splash" and "Lena", our scheme's bit rates are lower than 1bpp, whereas the result of scheme [15] is 2bpp. It indicates that the performance of our scheme is better while compressing the cover image on BTC compression domain.

As a representative scheme of JDHC applications, Qin et al.'s scheme [19] focused on VQ techniques. They designed a VQ-based JDHC scheme rather than using the BTC compression technique. It is known to all that VQ-based compression scheme [2] must have a codebook as extra data so that the decompression of the received VQ compression codes can be guaranteed. The experimental data of PSNR and hiding capacity obtained from comparing our proposed BTC-based JDHC scheme with VQ-based JDHC scheme [19] are also provided in Table VIIII. Since VQ-based compression techniques basically provide better compression performance than BTC-based compression techniques, the bit rate results of scheme [19] are better than our proposed scheme. With respect to hiding capacity, Table VIIII indicates our scheme achieves better hiding capacity than scheme [19]. Our scheme could

embed one secret bit into each block, but scheme [19] could just embed one secret bit in some specific blocks. Therefore, the number of embeddable blocks with our scheme is much more than that with scheme [19]. Moreover, the hiding capacity is fixed for all test images with our scheme. By contrast, the lowest payload is only 6462 bits, i.e., "Baboon", with scheme [19]. With respect to the image quality (PSNRs) results listed in Table VIIII, it is obvious that most image qualities of the decompressed test images are more than 30 dB for our scheme, while most of the PSNRs of the decompressed test images with scheme [19] are lower than 30dB. The worst case of scheme [19] is image "Baboon," the PSNR value was only 24.56dB. Thus, it can be concluded that our scheme could achieve a better image quality of decompressed images than scheme [19]. The comparison results of our proposed scheme, the representative BTC-based compression scheme [8], traditional BTC-based data hiding schemes [15], and VQ-based JDHC scheme [19] clearly demonstrate that our proposed scheme provided the superior performance.

V. Conclusions

This paper presents the first JDHC scheme designed for block truncation coding (BTC) compression domain. The proposed scheme seamlessly integrates the functions of image compression technique and secret data hiding technique into a single procedure. In the image decompression and secret data extraction procedure, a flexible option is provided, i.e., receivers can choose to extract only the hidden secret data without performing the decompression procedure, or they can conduct secret data extraction and block decompression synchronously. Experimental results showed that the size of output code-stream for our proposed scheme could be reduced efficiently after the embedding and compression procedures, and it could provide a larger embedding capacity compared with VQ-based JDHC scheme [19]. Moreover, the bit rate provided by the proposed scheme could be reduced to nearly 1 bpp, and 1 secret bit can be hidden into each compression block. The experimental data confirmed that our scheme achieves outstanding bit rate, visual quality, and hiding capacity for different test images, exceeding the performances of representative BTC compression scheme [8], BTC based data hiding scheme [15], and JDHC scheme [19].

References

[1] O. Mohammed, Y. Salah, "Image compression based on mapping image fractals to rational numbers," *IEEE Access*, Vol. 6, pp. 47062 - 47074

[2] R. M. Gray, "Vector quantization," *IEEE ASSP Magazine*, 1984, Vol. 1, No. 2, pp. 4-29.

[3] E. J. Delp, O. R. Mitchell, "Image compression using block truncation coding," *IEEE Transactions on Communications*, 1979, Vol. 27, No. 9, pp. 1335-1341.

[4] M. D. Lema, O. R. Mitchell, "Absolute moment block truncation coding and its application to color images," *IEEE Transactions on Communications*, 1984, Vol. 32, No. 10, pp. 1148-1157.

[5] Y. V. RAMANA, and C. ESWARAN, "A new algorithm for BTC bit plane coding," *IEEE Transactions on Communications*, 1995, Vol. 43, No. 6, pp. 2010-2011.

[6] Y. C. Hu, "Improved moment preserving block truncation coding for image compression," *Electronics Letters*, 2003, Vol. 39, No. 19, pp. 1377-1379.

[7] J. Mathews, M. S. Nair, L. Jo, "Modified BTC algorithm for gray scale images using max-min quantizer," *International Multi-Conference on Automation, Computing, Communication, Control and Compressed Sensing (iMac4s), 2013*, pp. 377-382.

[8] Z. Xiang, Y. C. Hu, Y. H. Yao, C. Qin, C., "Adaptive and dynamic multi-grouping scheme for absolute moment block truncation coding," Multimedia Tools and Applications, 2018, doi: 10.1007/s11042-018-6030-5.

[9] J. M. Guo, M. F. Wu, "Improved block truncation coding based on the void-and-cluster dithering approach," *IEEE Transactions on Image Processing*, 2009, Vol. 18, No. 1, pp. 211-213.

[10] W. Tang, B. Li, S. Tan, M. Barni, J. Huang, "Cnn-based adversarial embedding for image steganography," *IEEE Transactions on Information Forensics and Security*, 2019, doi: 10.1109/TIFS.2019.2891237.

[11] D. Hou, W. Zhang, Y. Yang, N. Yu, "Reversible data hiding under inconsistent distortion metrics," *IEEE Transactions on Image Processing*, 2018, Vol. 27(10), pp. 5087 - 5099.

[12] M. H. Lin, C. C. Chang, "A novel information hiding scheme based on BTC," *The 4th International Conference on Computer and Information Technology*, 2004, pp. 66-71.

[13] J. C. Chuang, C. C. Chang, "Using a simple and fast image compression algorithm to hide secret information," *International journal of computers & applications*, 2006, Vol. 28, No. 4, pp.329-333.

[14] C. C. Chang, C. Y. Lin, Y. H. Fan, "Lossless data hiding for color images based on block truncation coding," *Pattern Recognition*, 2008, Vol. 41, No.7, pp. 2347-2357.

[15] C. Qin, P. Ji, C. C. Chang, J. Dong, X. Sun, X, "Non-uniform watermark sharing based on optimal iterative btc for image tampering recovery," IEEE Multimedia, 2018, 25(3), pp. 36-48.

[16] W. Sun, Z. M. Lu, Y. C. Wen, "High performance reversible data hiding for block truncation coding compressed images," *Signal, Image and Video Processing*, March 2013, Vol. 7, No. 2, pp 297-306.

[17] C. C. Lin, X. L. Liu, W. L. Tai, S. M. Yuan, "A novel reversible data hiding scheme based on AMBTC compression technique," *Multimedia Tools and Applications*, 2015, Vol. 74, No. 11, pp. 3823-3842.

[18] W. Hong, Y.B. Ma, H.C. Wu, "An efficient reversible data hiding method for AMBTC compressed images," Multimedia Tools and Applications 76(4), pp. 5441–5460, 2017.

[19] C. Qin, C. C. Chang, Y. P. Chiu, "A novel joint data-hiding and compression scheme based on SMVQ and image inpainting," *IEEE Transactions on Image Processing*, 2014, Vol. 23, No. 3, pp. 969-978.

[20] S. Badr, P. Noufal, "Integrated data hiding and compression scheme based on smvq and foe inpainting. Procedia Technology, 2016, 24, 1008-1015.

[21] K. S. Wong, H. Kiya, "Reversible data hiding for compression-friendly image encryption method: Asia-pacific Signal & Information Processing Association Summit & Conference, 2018, DOI: 10.1109/APSIPA.2017.8282213.

[22] Y. Y. Satpute, N. R. Pardeshi, "Advanced side-match vector quantization and image inpainting used for data hiding and compression," *International Journal of Engineering Trends and Technology*, 2015, Vol. 2, No. 2, pp. 385-389.

[23] M. Bertalmio, G. Sapiro, V. Caselles, C. Ballester, "Image inpainting," *Proceedings of 27th International Conference on Computer Graphics and Interactive Techniques*, 2000, New Orleans, Louisiana, USA, pp. 417-424.

[24] H. Grossauer, "A combined PDE and texture synthesis approach to inpainting," *8th European Conference on Computer Vision*, 2004, pp. 214–224.

[25] S. D. Rane, G. Sapiro, M. Bertalmio, "Structure and texture filling-in of missing image blocks in wireless transmission and compression applications," *IEEE Transactions on Image Processing*, 2003, Vol. 12, No. 3, pp. 296–303.

[26] C. Qin, F. Cao, X. Zhang, "Efficient image inpainting using adaptive edge-preserving propagation," *The Imaging Science Journal*, 2011, Vol. 59, No. 4, pp. 211-218.
[27] A. Criminisi, P. Perez, K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Transactions on Image Processing*, 2004, Vol. 13, No. 9, pp. 1200-1212.
[28] T. N. Narasimhan, P. A. Witherspoon, "An integrated finite difference method for analyzing fluid flow in porous media," *Water Resources Research*, 1976, Vol. 12, No. 1, pp. 57-64.
[29] K. H. Jung, K. Y. Yoo, "Steganographic method based on interpolation and LSB substitution of digital images," *Multimedia Tools and Applications*, 2015, Vol. 74, No. 6, pp. 2143-2155.
[30] B. G. Mobasseri, R. J. Berger, M. P. Marcinak, Y. J. NaikRaiKar, "Data embedding in JPEG bitstream by code mapping," *IEEE Trans. Image Proc.*, 2010, Vol. 19, No. 4, pp. 958–966.
[31] Z. X. Qian, X. P. Zhang, "Lossless data hiding in JPEG bit-stream," *J. Syst. Softw.*, 2012, Vol. 85, No. 2, pp. 309–313.
[32] X. L. Liu, C. C. Chang, C. C. Lin, S. M. Yuan, "A high-payload, reversible data hiding scheme based on histogram modification in jpeg bitstream." *The Imaging Science Journal*, 2016, Vol. 64, No. 7, pp. 364-373.
[33] J. D. Lee, Y. H. Chiou, J. M. Guo, "Lossless data hiding for VQ indices based on neighboring correlation," *Inform. Sci.* 2014, Vol. 221, pp. 419–438.
[34] C. C. Lin, X. L. Liu, S. M. Yuan, "Reversible data hiding for vq-compressed images based on search-order coding and state-codebook mapping," *Information Sciences*, 2015, Vol, 293, pp. 314-326.
[35] USC-SIPI, [Online], Available: http://sipi.usc.edu/database/.