Journal of Network Intelligence
Taiwan Ubiquitous Information
 ISSN 2414-8105 (Online)
Volume **5**, Number **2**, May **2020**

An Adjustable RDH Method for AMBTC-Compressed Images Using Unilateral Pixel Value Modification Strategy

Wenbin Zheng[1,2], Ching-Chun Chang[3], Ji-Hwei Horng[4*],
Juan Lin[5*], Yunqing Shi[6]

[1] School of Information Science and Engineering, Fujian University of Technology, Fuzhou,350118, China
[2] National Demonstration Center for Experimental Electronic Information and Electrical Technology Education, Fujian University of Technology, Fuzhou 350118, China
[3] Department of Electronic Engineering, Tsinghua University, Beijing, 100084, China
[4] Department of Electronic Engineering, National Quemoy University, Kinmen, 89250, Taiwan
[5] School of Electronic and Information Engineering, Fuqing Branch of Fujian Normal University, Fuzhou, 350300, China
[6] Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07103 USA

Received January 2019; revised November 2019
(Corresponding author: Ji-Hwei Horng(horng@email.nqu.edu.tw), Juan Lin(lj2020229@gmail.com)

Abstract. *In view of the global increase in smartphone and social media usages, there is a growing demand for efficient and secure data storage and transmission. Data hiding in compressed domain enjoys the best of both worlds. In other words, one can use compression techniques to realize storage and transmission efficiency and deploy data hiding methods as a means of accomplishing information security. However, this task is unexpectedly challenging because compression has largely reduced and eliminated data redundancy on which most data hiding algorithms depend. In this paper, we proposed a novel reversible data hiding scheme based on absolute moment block truncation coding (AMBTC) with adjustable payload. AMBTC is one of the most widely used lossy image compression techniques. The proposed scheme embeds secret data into AMBTC compressed images by modifying the gray level and is able to recover the original compression image after secret data extraction. The amount of secret information can be configured to embed one to three digits per pixel. The experimental results showed that our scheme achieved the highest embedding rate compared to the related works under the same image quality constraints.*
Keywords: AMBTC, Adjustable, Reversible data hiding, Unilateral, Pixel value modification

1. **Introduction.** With the development of network technology, the amount of data transmitted by people through the network is massive. Data hiding has become an important issue of information security. Different from traditional cryptographic encryption methods, data hiding is to carry out subtle modifications to ordinary carriers to transmit information, but these modifications will not attract people's attention. That is to say, information is hidden in text files, image signals, voice signals or video signals and other digital media without causing people to perceive visual and auditory changes of these carriers [1]. Through concealing the "communication behavior", the security of information transmission can be greatly improved. Digital image is the most common carrier. In

recent decades, researchers have proposed many information hiding schemes based on digital images in spatial domain [2,3,4,5], frequency domain [6,7,8] and compression domain [9,10,11]. In order to improve the transmission efficiency and reduce the storage space, it is usually necessary to compress the transmitted data. Therefore, how to efficiently implement information hiding in compressed images is an important research problem.

In the compression domain, features of different compression techniques are used to embed secret data. A lot of data hiding methods have been proposed for different image compression formats, including VQ [12], SMVQ [9], JPEG [13,14] and AMBTC [15-19]. Compared with the compression schemes of VQ, SMVQ and JPEG, the AMBTC-based methods have the advantages of low computational complexity, efficient encoding/decoding process, high compression ratio and good image quality. In 2006, Chuang and Chang [15] defined a threshold based on the difference between high and low quantization levels, and classified blocks into smooth blocks and complex blocks according to this threshold. For smooth blocks, the bitmap was directly replaced with secret information; while complex blocks were left unembedded. Ou and Sun [20] used Chuang and Chang's method to classify blocks and embed secret information into smooth blocks. In addition, they utilized Chen *et al.*'s method [21] to embed secret data into complex blocks by switching the order of two quantization levels. Tang *et al.* [22] used the interpolation technique to embed secret information, and obtained high capacity and good image quality. However, these solutions are irreversible and cannot be applied to situations such as military, medical and court forensics, because the distortion of the information carrier cannot be accepted in these situations. Therefore, reversible data hiding (RDH) technology for AMBTC compressed images has attracted researchers' attention. The existing RDH AMBTC schemes can be divided into four categories [23], namely histogram shifting (HS) [24-28], bitmap-based transformation [29-31], prediction error expansion (PEE) [32-37] and compressed image-based data hiding [38-40].

HS-based method was proposed by Lo *et al.* [41]. Their method is to calculate the difference between high and low quantization values, then obtain the three sets of high quantitation values, low quantitation values and differences, and then embed secret information into any two sets of them by using the principle of histogram shifting [4]. This method can achieve good image quality, but its payload is low. The bitmap transformation method embeds secret information by replacing bitmap, which can preserve the compression ratio, but the payload and PSNR are rather low. PEE method can obtain better image quality and higher storage capacity than HS-based and bitmap transformation-based methods. However, these methods are based on the AMBTC compression codes, and their payloads are lower than those of the compressed image-based methods. The compressed image-based method was first proposed by Lin *et al.* in 2015 [38]. Lin *et al.*'s method is to combine secret information and bitmap into a cover block, and then for embeddable blocks, four values defined by the block mean μ and standard deviation σ, namely $\mu+\sigma$, $\mu-\sigma$, $\mu+\sigma+1$, $\mu-\sigma-1$ are applied to replace the cover pixel values and result in a stego block. Thus, the embedding capacity of Lin *et al.*'s method is 1 bit per cover pixel. Later on, many RDH schemes based on compressed images have been proposed.

In 2018, Malik *et al.* proposed an AMBTC data hiding scheme based on pixel value adjustment strategy [40]. This scheme converts the secret information into ternary digits, and then modifies each cover pixel value by at most 1 to embed a secret digit. Due to the minimal adjustment, the visual quality of Malik *et al.*'s stego image is very close to the cover AMBTC image. In consideration of reversibility, a cover block is classified as an embeddable block provided that its two quantization values are gapped by at least 2 gray levels. This limitation severely degrades the embedding capacity of a smooth

cover image. Besides, the embedding capacity of an embeddable cover pixel is low and without flexibility. Based on the above discussions, we propose an AMBTC-based RDH scheme with adjustable payload. In our scheme, the modification for the two different quantization values are unilateral toward opposite directions. So that the limitation on the gap width between two quantization values can be reduced and the number of embeddable blocks is greatly improved. By utilizing different amount of maximum modifications, the embedding capacity of our scheme is switchable

The rest of this paper is organized as follows. In Section 2, we briefly introduce the AMBTC encoding and decoding process. In addition, the details of the Malik *et al.*'s scheme and its characteristics are further discussed. The proposed scheme is presented in Section 3. Section 4 gives the experimental results and discussions. Section 5 is the final conclusion.

2. **Related work.** In the section, we briefly introduce the AMBTC encoding and decoding process first. Then, Malik *et al.*'s [40] data hiding scheme is presented and its performance is discussed.

2.1. **AMBTC compression technique.** AMBTC is an improved version of the Block Truncation Coding (BTC) proposed by Lema and Mitchell in 1984. Although it is a lossy compression technique, its algorithm is simple and compression ratio is high. In AMBTC, the original image I is partitioned into N mutually exclusive blocks of size $m \times n$. Then, the following processing is applied to each block. Firstly, the average pixel value P_i of the whole block is calculated by Equation (1).

$$P_i = \frac{1}{T} \sum_{j=1}^{T} p_{ij} \tag{1}$$

where P_{ij} is the j-th pixel value of the i-th block and $T = m \times n$ is the total number of pixels in the i-th block. Then, H_i and L_i are calculated using Equation (2) and Equation (3), respectively.

$$H_i = \left\lfloor \frac{1}{u} \sum_{p_{ij} \geq p_i} p_{ij} \right\rfloor \tag{2}$$

$$L_i = \left\lfloor \frac{1}{T-u} \sum_{p_{ij} < p_i} p_{ij} \right\rfloor \tag{3}$$

where u is the number of pixels with values greater than or equal to P_i and H_i denotes their average value, while L_i denotes the average value of the remainders. Then, the pixels with high values are represented by '1', and the pixels with low values are represented by '0' to constitute the bitmap B_i , as shown in Equation (4).

$$B_{ij} = \begin{cases} 1, & if(p_{ij} \geq P_i) \\ 0, & if(p_{ij} < P_i) \end{cases} \tag{4}$$

Finally, the compressed code of the i-th block (H_i, L_i, B_i) is transmitted. At the receiving end, the image block can be reconstructed from (H_i, L_i, B_i) by simply replacing '1' in Bi by H_i and '0' by L_i.

An example is given in FIGURE 1 to illustrate the AMBTC encoding and decoding process. Assuming that FIGURE 1(a) is a block of the original image with a size of 4×4. First, the average of this block is calculated using Equation (1), which gives $P_i = 89$. According to Equation (4), the bitmap shown in FIGURE 1(b) can be obtained. The

quantization levels H_i and L_i can be calculated using Equation (2) and Equation (3), which give H_i=100, L_i=77. Finally, this block is represented as a compressed code (100,77, [1 1 0 0; 1 1 0 1; 1 0 1 0; 1 0 0 0]) and then transmitted. At the receiving end, '1' in the bitmap is replaced with '100', '0' is replaced with '77', and the result is shown in FIGURE 1(c).

100	89	88	75
125	102	67	101
96	76	103	77
90	68	82	85

(a)

1	1	0	0
1	1	0	1
1	0	1	0
1	0	0	0

(b)

100	100	77	77
100	100	77	100
100	77	100	77
100	77	77	77

(c)

FIGURE 1. AMBTC encoding and decoding example (a)Original block; (b)Bitmap; (c)Decoded block

2.2. **Malik *et al.*'s scheme.** Malik *et al.*'s method [40] is based on the AMBTC compressed image. The original image is compressed using the AMBTC method, and then the secret information is embedded in the reconstructed compressed image with a reversible scheme.

In Malik *et al.*'s method, the original image is partitioned into N mutually exclusive $n \times n$-sized blocks. Then, each block is compressed by AMBTC method to obtain the compressed code (H_i, L_i, B_i). After that the compressed image is reconstructed by AMBTC decoding rules. The reconstructed compressed image consists of N blocks of $n \times n$ pixels, each block has only two different pixel values, namely H_i and L_i. The Malik *et al.*'s scheme uses the redundancy of the pixel values to embed information. At the same time, the stego image is kept reversible after data extraction.

At first, the difference between H_i and L_i for each block is computed. The blocks of $H_i - L_i > 2$ are determined to be embeddable blocks, while the others are non-embeddable. For embeddable blocks, under the consideration of reversibility, the first H_i and the first L_i pixel are non-embeddable and kept unchanged. The binary secret stream is converted to ternary number system representation. By scanning each embeddable pixel in turn, the pixel value is kept unchanged to embed '0', decreased by 1 to embed '1', and increased by 1 to embed '2'. Execute this process until the cover image is fully embedded.

At the receiving end, the pixel value distribution in each block is checked first. If the block is two-valued and the difference between the two values is less than or equal to 2, it is a non-embeddable block; otherwise, it is embeddable. For an embeddable block, the first pixel is one of the quantization values. The second quantization value is the first of the remaining pixel values that is out of its modification range, i.e. ± 1. According to the values of the two non-embeddable pixels, the secret digits can be extracted from the embeddable pixels sequentially and the compressed code can be recovered.

In Malik *et al.*'s scheme, each embeddable block has 2 non-embeddable pixels and 14 embeddable pixels. For each embeddable pixel, a ternary secret digit can be embedded. The total embedding capacity for an embeddable block is $14 \times log_2 3$ bits. Since this scheme utilizes six states, namely $H_i + 1, H_i, H_i - 1, L_i + 1, L_i, L_i - 1$, to embed secret digits, the difference $H_i - L_i$ must be greater than 2 to discriminate between $H_i - 1$ and $L_i + 1$. This limitation severely reduces the number of embeddable blocks in a smooth cover image.

3. **Proposed scheme.** To reduce the limitation on the embeddable decision and increase the embedding capacity, a unilateral embedding scheme is proposed. The modification of pixel value is always addition for H_i and subtraction for L_i. By reserving a guard state for reversible consideration, all blocks of $H_i - L_i > 1$ are embeddable. In addition, by tuning the maximum amount of modification, the embedding capacity of our scheme is adjustable. The proposed embedding, extraction, and compressed code recovering processes are described below.

3.1. **Embedding phase.** Partition the original image O into mutually exclusive blocks $\{O_i\}_{i=1}^{M}$ of size $m \times n$, where M denotes the total number of blocks. The image O is compressed into $\{(H_i, L_i, B_i)\}_{i=1}^{M}$ according to AMBTC encoding, and then the decompressed image R is reconstructed according to AMBTC decoding. In this way, the pixel values in each block $R_i\,(i \in [1, M])$ are either H_i or L_i. The first H_i and the first L_i in the block are denoted as H_{if} and L_{if}, respectively, and are considered as non-embeddable pixels. Before embedding, the embedding capacity p bits/pixel should be determined first and the maximum modification of pixel value is therefore $v(v = 2^p - 1)$. According to p, the binary stream of secret data S is converted into 2^p-ary secret set $S' = \{s'_r\}_{r=1}^{k}$, where s'_r represents the r-th secret digit. The secret digits are consecutively embedded into the embeddable pixels by adding in case of H_i and by subtracting in case of L_i.

$R_{ij}\,(j \in [1, mn])$ is used to represent the j-th element in the block R_i. After embedding secret digits, the first pixel value R_{i1} of an embeddable block can be either H_{if} or L_{if}. The unilateral property of embedding can be utilized to find the other value of H_{if} or L_{if}. In the case of $R_{i1} = H_{if}$, L_{if} can be determined by searching the first pixel value x_1 that is less than R_{i1}. In the contrary case of $R_{i1} = L_{if}$, H_{if} can be determined by searching the first pixel value x_2 that is greater than R_{i1}. Since there is no value between H_{if} and L_{if}, we can discriminate the two cases by checking the remaining pixels. If a value x_3 can be found between R_{i1} and x_1, then $R_{i1} = L_{if}$ and $x_2 = H_{if}$. On the contrary, if a value x_3 can be found between R_{i1} and x_2, then $R_{i1} = H_{if}$ and $x_1 = L_{if}$. An ambiguity situation may occur that there is no pixel value in between for both cases. Under such situation, we further define the last pixel $R_{i(mn)}$ as non-embeddable and assign different values to identify different cases. The pseudo code for embedding algorithm is described below. Its corresponding flowchart is given in FIGURE 2.

Input: AMBTC-compressed image $\{R_i\}_{i=1}^{M}$, maximum modification $v(v = 2^p - 1)$, and 2^p-ary secret set S'

Output: Stego image $\{R'_i\}_{i=1}^{M}$

Step 1. Read the compression image $\{R_i\}_{i=1}^{M}$ sequentially.

Step 2. Check embeddable block

if $H_i - L_i \leq 1$ or $(H_i + v > 255$ or $L_i - v < 0)$,

the block is non-embeddable;

else

the block is embeddable;

end

Step 3. For an embeddable block, embed secret digits to all R_{ij} except for $R_{i(mn)}$.

For all R_{ij} except for $R_{i(mn)}$.

if $R_{ij} == H_{if}$ or $R_{ij} == L_{if}$,

do nothing;

else

if $R_{ij} == H_i$,

$R'_{ij} = R_{ij} + s'_r$;

else

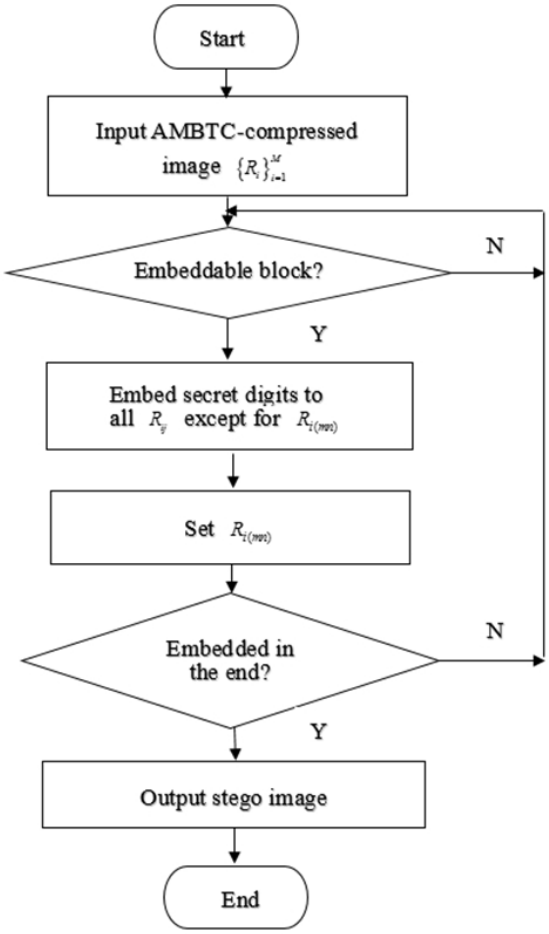


FIGURE 2. Flowchart of the Data Embedding Process

$R'_{ij} = R_{ij} - s'_r$;
end
end
end

Step 4. For an embeddable block, set $R_{i(mn)}$.
Retrieve R_{i1} and check the remaining pixel values from R'_{i2} to $R'_{i(mn-1)}$
If there are x_1, x_2, x_3 and $R_{i1} > x_3 > x_1$ or $R_{i1} < x_3 < x_2$,
$R_{i(mn)}$ is an embeddable pixel and embed secret data according to rules.
else if there is x_1 or x_2,
$R_{i(mn)}$ is an embeddable pixel and embed secret data according to rules.
else
if $R_{i1} \neq R_{i(mn)}$,
$R'_{i(mn)} = R_{i(mn)}$;
else
if $R_{i1} == L_i$,
$R'_{i(mn)} = L_i + 1$;
else
$R'_{i(mn)} = H_i - 1$;
end
end
end

end
Step 5: Repeat Steps 1-4 are until all bits of secret digits are embedded.
Step 6: After data hiding is completed, the stego images are output.

3.2. **Example of embedding phase.** An example of embedding is given in FIGURE 3. A 4×4-sized block of the original image O is shown in FIGURE 3(a). After AMBTC compression, the trio compression code is obtained, which is {54,45, [0 0 1 1;0 0 1 0; 0 0 1 1;0 0 1 1]}, as shown in FIGURE 3(b). The reconstructed version is shown in FIGURE 3(c). Since $H_i - L_i = 54 - 45 = 9 > 1$, the given block is determined as embeddable. The L_{if} and H_{if} are the first pixel R_{i1} and the third pixel R_{i3} is painted blue, respectively. Let $p = 2$, the binary secret stream $S = \{11111000101101001011101100 01\}_2$ is partitioned into doublets $S = \{11, 11, 10, 00, 10, 11, 01, 00, 10, 11, 10, 11, 00, 01\}_2$, where $\{.\}_2$ represents the binary number system. Then, it is converted into a set of 4-based secret digits $S' = \{3, 3, 2, 0, 2, 3, 1, 0, 2, 3, 2, 3, 0, 1\}_4$, where $\{.\}_4$ is used to denote the 4-ary number. After that, secret digits are embedded to embeddable pixels consecutively. Since the first embeddable pixel $R_{i2} = L_i$, the subtraction rule is applied $R'_{i2} = R_{i2} - s'_1 = 45 - 3 = 42$. The next embeddable pixel $R_{i4} = H_i$, the addition rule is applied $R'_{i4} = R_{i4} + s'_2 = 54 + 3 = 57$. In a similar manner, the secret digits are embedded into the remaining embeddable pixels and the resulting stego block is shown in FIGURE 3(d). The first pixel value less than $R_{i1} = 45$ is $x_1 = 42$ and the first pixel greater than R_{i1} is $x_2 = 54$. Because we can find a value $x_3 = 43$ such that $R_{i1} > x_3 > x_1$, the last pixel of the block is an embeddable pixel. Three ambiguity cases of $R_{i1} \neq R_{i(mn)}$, $R_{i1} = L_i$, and $R_{i1} = H_i$ are given in FIGURE 4(a), (b), and (c), respectively.

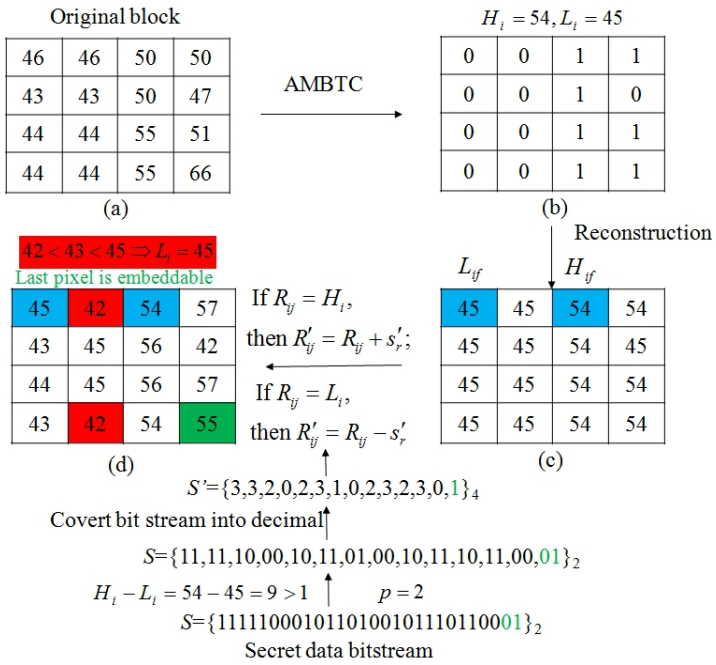


FIGURE 3. Example of embedding phase (a)Original block; (b)Bitmap; (c)Compressed block; (d)Stego block

3.3. **Extracting and recovering phase.** In the extracting and recovering phase, the secret binary stream is extracted from the stego image and the AMBTC compressed image

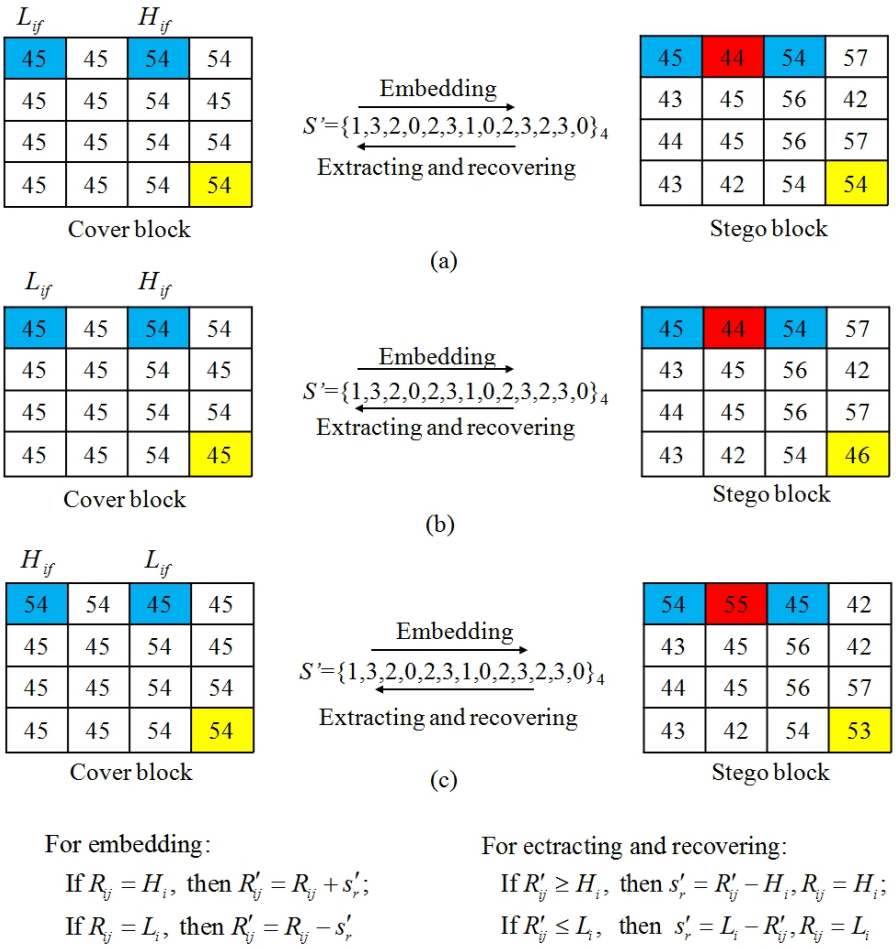


FIGURE 4. Example of R_{imn} as a non-embeddable pixel (a)Case I; (b)Case I; (c)Case III

can be recovered. The detail process is shown as below. Its corresponding flowchart is given in FIGURE 5.

Input: A stego image R'.

Output: A recovered AMBTC compressed image $\{R_i\}_{i=1}^{M}$ and extracted secret data S.

Step 1. Divide stego image R' into $m \times n$-sized non-overlapped pixel blocks $\{R'_i\}_{i=1}^{M}$.

Step 2. Scan $\{R'_i\}_{i=1}^{M}$ according to the raster scan order.

Step 3. Identify H_{if} and L_{if}. For R'_i, first find x_1, x_2 and x_3, then analyze their relationship with R'_{i1} to identify H_{if} and L_{if}. According to the search results of x_1, x_2 and x_3, the process of identifying H_{if} and L_{if} can be divided into the following three cases.

Case 1: Missing either x_1 or x_2.

(a)Both x_1 and x_2 cannot be found. The block is of a single value and non-embeddable.

(b)x_2 cannot be found.

If $R'_{i1} - x_1 \leq 1$, The block is non-embeddable.

else $H_{if} = R'_{i1}$, $L_{if} = x_1$.

(c)x_1 cannot be found.

If $x_2 - R'_{i1} \leq 1$, The block is non-embeddable.
else $H_{if} = x_2,\ L_{if} = R'_{i1}$.
Case 2: x_1, x_2 and x_3 all exist.
If $x_2 > x_3 > R'_{i1} > x_1$, then $H_{if} = R'_{i1},\ L_{if} = x_1$;
If $x_2 > R'_{i1} > x_3 > x_1$, then $H_{if} = x_2,\ L_{if} = R'_{i1}$.
Case 3: Missing x_3.
If $R'_{i1} > R'_{i(mn)}$, then $H_{if} = R'_{i1},\ L_{if} = x_1$ and $R'_{i(mn)}$ is non-embeddable pixel.
If $R'_{i1} < R'_{i(mn)}$, then $H_{if} = x_2,\ L_{if} = R'_{i1}$ and $R'_{i(mn)}$ is non-embeddable pixel.
Step 4. Extract secret digit and restore original pixel value.
If $R'_{ij} \leq L_i$, then $s'_r = L_{if} - R'_{ij}$ and $R_{ij} = L_{if}$.
else $s'_r = R'_{ij} - H_{if}$ and $R_{ij} = H_{if}$.
Step 5. Repeat Steps 1-4 until all secret digits are extracted.
Step 6. Convert S' into S.
Step 7. Once data extracting is completed, the original AMBTC image is also recovered.

3.4. **Example of extracting and recovering phase.** An example of secret data extraction and compressed image recovery in shown in FIGURE 6.

Since the stego block in FIGURE 6(a) has more than 3 different valued pixels, it is an embeddable block and thus H_{if} and L_{if} should be identified first. By applying our rules, $R'_{i1} = 45$, $x_1 = 42$, $x_2 = 54$ and $x_3 = R'_{i5} = 43$ satisfies $x_1 < x_3 < R'_{i1} < x_2$. Therefore, R'_{i1} is L_{if} and $R'_{i3} = H_{if}$, i.e., $H_{if} = 54$, $L_{if} = 45$. It can also be determined that R'_{i16} is an embeddable pixel and contains secret information.

Based on H_{if} and L_{if}, the secret digits can be extracted and the original compressed image block can be recovered by the following rule. If $R'_{ij} \leq L_{if}$, then $s'_r = L_{if} - R'_{ij}$ and $R_{ij} = L_{if}$, else $s'_r = R'_{ij} - H_{if}$ and $R_{ij} = H_{if}$. For example, $R'_{i2} = 42 < 45$, then $s'_r = 45 - 42 = 3$, $R_{i2} = 45$. In the similar way, we can extract all the secret information $S' = \{3, 3, 2, 0, 2, 3, 1, 0, 2, 3, 2, 3, 0, 1\}_4$ and reconstruct the original compressed image {45 45 54 54; 45 45 54 45; 45 45 54 54; 45 45 54 54} as shown in FIGURE 6(b). Finally according to $p = 2$, we convert S' back to the binary bitstream $S = \{11,11,10,00,10,11,01,00,10,11,10,11,00,01\}_2$.

For the more complicated cases of ambiguity situation, we can refer back to FIGURE 4. There are three different cases that R'_{i16} is a non-embeddable pixel. For Case I, $R'_{i16} = 54$, $R'_{i1} = 45$, $R'_{i1} < R'_{i16}$. Therefore, R'_{i1} is L_{if} and R'_{i3} is H_{if}, i.e., $H_{if} = 54$, $L_{if} = 45$. For Case II, $R'_{i16} = 46$, $R'_{i1} = 45$, $R'_{i1} < R'_{i16}$. Therefore, R'_{i1} is L_{if} and R'_{i3} is H_{if}, i.e., $H_{if} = 54$, $L_{if} = 45$. For Case III, $R'_{i16} = 53$ $R'_{i1} = 54$, $R'_{i1} > R'_{i16}$. So, R'_{i1} is H_{if} and R'_{i3} is L_{if}, i.e. $H_{if} = 54$, $L_{if} = 45$.

4. **Experimental results.** In this section, we will show the performance of the proposed scheme through several sets of experiments. The ten classic grayscale images of 512×512 pixels, i.e., Lena, F16, Boat, Baboon, Peppers, House, Couple, Elaine, Sailboat and Zelda as shown in FIGURE 7 are used as test images. All test images are divided into 4×4 blocks, and then processed by AMBTC compression method. The secret data for the simulation is generated by a random number generator.

The two indices ER and PSNR are applied to measure the performance of the proposed data hiding scheme. The embedding capacity ER is defined by Equation (5), where N_s represents the total number of embedded secret bits, and N_p represents the total number of pixels in the cover image.

$$ER = \frac{N_s}{N_p} \tag{5}$$

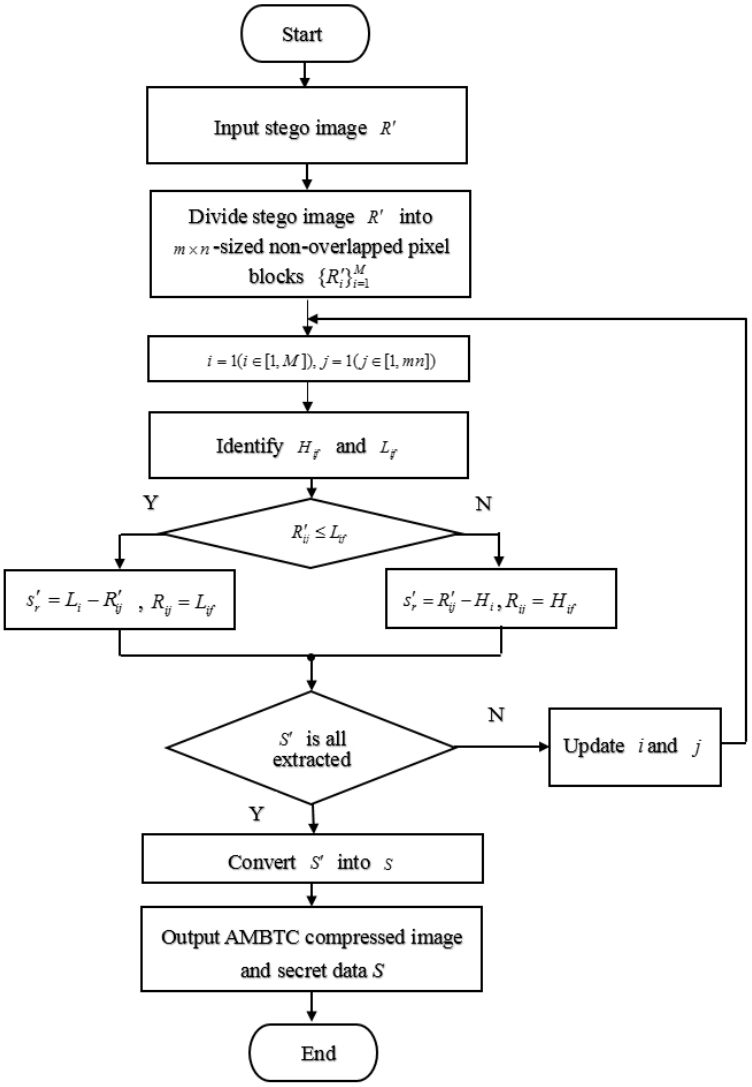


FIGURE 5. Flowchart of the data extraction and image recovery

The visual quality is measured by peak signal to noise ratio (PSNR), which is defined in Equation (6). where the mean-square error (MSE) is given in Equation (7).

$$\text{PSNR} = 10\log_{10}(\frac{255^2}{MSE})(dB) \tag{6}$$

$$MSE = \sum_{i=1}^{W}\sum_{j=1}^{H}(O_{ij} - R'_{ij})^2 /(W \times H) \tag{7}$$

where O_{ij} and R'_{ij} denote the pixel value at the (i, j) coordinates of the original and stego image, respectively; H and W represent the height and width of the image, respectively. The larger the PSNR, the better the visual quality of the image. However, it is a tradeoff between the PSNR and the ER. The higher the hiding capacity, the more the original image is modified, the greater the difference between the stego image and the

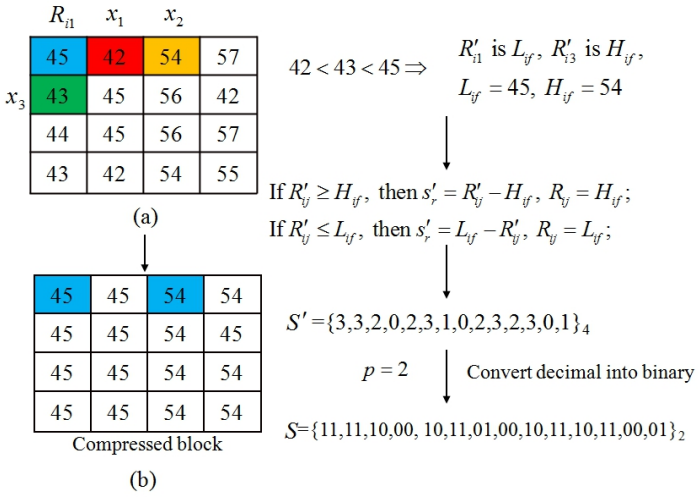


FIGURE 6. An example of data extracting and image recovery

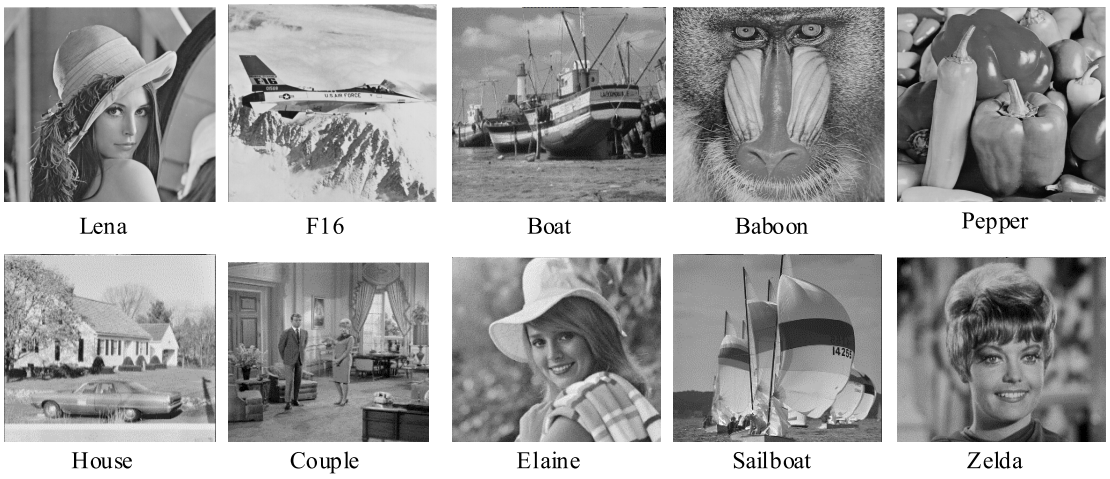


FIGURE 7. Ten test images

original image, and the lower the PSNR. As a result, we often increase the hiding capacity within acceptable visual quality.

4.1. **Performance of the proposed method.** TABLE 1 shows the capacity and visual quality of 10 test images under different embedding bits p, where PSNR_C represents the PSNR of the original image O and the AMBTC compressed image R, while the PSNR_S represents the PSNR of the original image and the stego AMBTC compressed image R'. Since the pixel values are modified to embed the secret digits, the AMBTC compressed image will be degraded. However, it can be clearly seen from the TABLE 1 that at $p = 1$, the PSNR_S and PSNR_C are very close, and the average difference is only 32.11-32.03=0.08dB. It means that our scheme can have good image quality. On the other hand, by switching the p value, the proposed method can obtain different capacity within an acceptable range of image quality. When $p = 2$, the average capacity of 454800bits

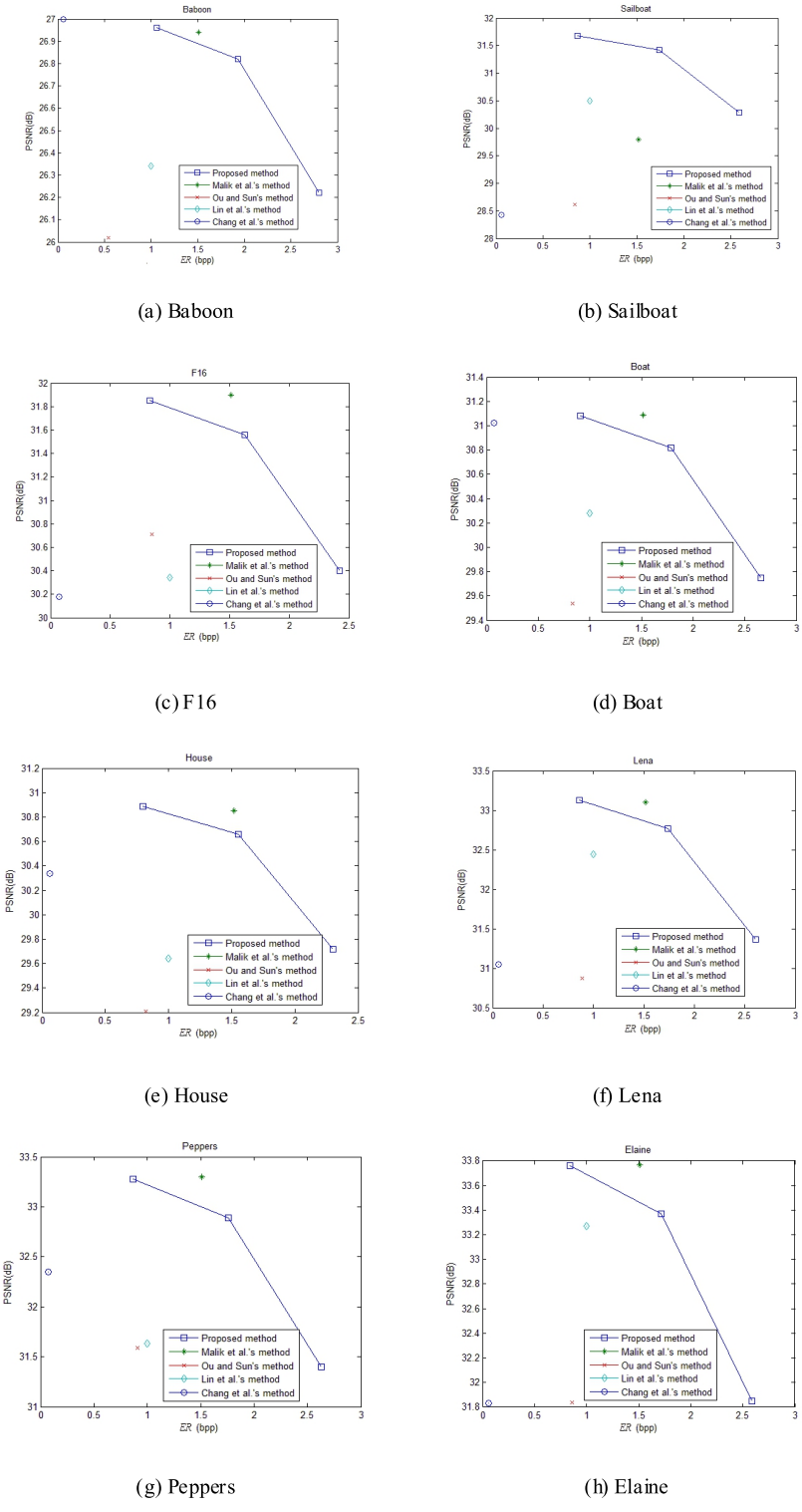
TABLE 1. Performance of our proposed scheme

Test Images	$PSNR_c$ (dB)	$p = 1$		$p = 2$		$p = 3$	
		$PSNR_s$ (dB)	N_s (bits)	$PSNR_s$ (dB)	N_s (bits)	$PSNR_s$ (dB)	N_s (bits)
Lena	33.2	33.13	225260	32.77	455475	31.36	685061
F16	31.95	31.85	217606	31.56	427190	30.4	635044
Boat	31.15	31.08	237555	30.81	468137	29.75	697775
Baboon	26.98	26.96	277375	26.82	506069	26.22	734029
Peppers	33.39	33.28	228773	32.89	460220	31.4	688312
House	30.97	30.89	209086	30.66	407574	29.72	601935
Couple	31.25	31.16	239259	30.9	469841	29.83	695809
Elaine	33.88	33.76	219650	33.37	449682	31.85	678691
Sailboat	31.72	31.68	228878	31.42	455318	30.29	679792
Zelda	36.65	36.48	216898	35.82	448502	33.52	676856
Average	32.11	32.03	230034	31.7	454800	30.43	677330

can be obtained, and the average PSNRs can be preserved above 30dB. When $p = 3$, the average capacity is 677330 bits, and the more complex the image, the higher the storage, and the less quality change. For example, the capacity of Baboon is 734029 bits, and the PSNRs is decreased by only 0.76dB with respect to $PSNR_C$.

4.2. **Performance comparison.** To further illustrate the performance of the proposed scheme, in this section we will compare it with four other methods, including Malik *et al.*'s method [40], Ou and Sun's method [20], Lin *et al.*'s method [38] and Li *et al.*'s method [24]. TABLE 2 shows the comparison of the capacity of the five schemes in the case of approximate $PSNR_S$. For the sake of simplicity, we take the capacity and $PSNR_S$ of the proposed scheme with $p = 2$ for comparison. From the data in the TABLE 2, it can be seen that the capacity of proposed scheme is much larger than the other four schemes, and the $PSNR_S$ is also higher than Ou and Sun's method [20], Lin *et al.*'s method [38] and Li *et al.*'s method [24]. Among all the schemes, Malik *et al.*'s Method [40] has the highest $PSNR_S$, because it only modifies the pixel value at most by 1 when embedding secret data, which makes it has high image quality. However, the proposed scheme has much higher embedding capacity than Malik *et al.*'s scheme, and the PSNR is only slightly lower than Malik *et al.*'s scheme. As can be seen from the mean values in the TABLE 2, the average of $PSNR_S$ of the proposed scheme is 31.29 dB and the average of capacity is 453708 bits. Accordingly, the average of $PSNR_S$ of Malik *et al.*'s method is 31.34 and the average of capacity is 397224bits. The difference between the average of $PSNR_S$ of the two methods is only 0.05dB, which decreases by 0.15%. However, the capacity of proposed scheme is 56484bits, which is higher than that of Malik *et al.*'s method, with an increase of 14.22%.

As shown in FIGURE 8, the blue line indicates the changes of the capacity and $PSNR_S$ of the proposed scheme under different embedding bits, p. The three points from left to right correspond to the three cases of p=1,2,3. With the increase of p, the embedding capacity increases, the $PSNR_S$ decreases, but they are all in the acceptable range of more than 30dB. Baboon is a special case, its $PSNR_C$ is 26.98dB. Even so, its $PSNR_S$ is preserved at 26.22dB when the ER of it reaches 2.8 bpp (bit per pixel). However, the other four methods can only obtain fixed capacity and $PSNR_S$. Compared with Malik *et al.*'s scheme, the proposed scheme has a higher embedding capacity with a slightly degrade of image quality.



(a) Baboon (b) Sailboat

(c) F16 (d) Boat

(e) House (f) Lena

(g) Peppers (h) Elaine

FIGURE 8. Performance comparison between our scheme and four compared schemes for eight test images

TABLE 2. Performance of the proposed scheme with the $PSNR_S$ and the N_s by comparing with other four schemes for 8 test images

Method	Factor	Lena	F16	Boat	Baboon	Peppers	House	Elaine	Sailboat	Average
Proposed Method	N_s (bits)	455475	427190	468137	506069	460220	407574	449682	455318	453708
	$PSNR_s$ (dB)	32.77	31.56	30.81	26.82	32.89	30.66	33.37	31.42	31.29
Malik *et al.*'s Method[40]	N_s (bits)	397348	397147	397380	397105	397057	397187	397098	397466	397224
	$PSNR_s$ (dB)	33.1	31.9	31.09	26.94	33.3	30.85	33.77	29.8	31.34
Ou and Sun's Mehtod[20]	N_s (bits)	234004	223039	217264	141919	238969	214609	226549	219169	214440
	$PSNR_s$ (dB)	30.87	30.71	29.54	26.02	31.59	29.21	31.84	28.61	29.8
Lin *et al.*'s Method[38]	N_s (bits)	262096	262144	262112	262144	262144	261760	261408	262144	261994
	$PSNR_s$ (dB)	32.44	30.34	30.28	26.34	31.6311	29.64	33.2674	30.5	30.55
Li *et al.*'s Method[24]	N_s (bits)	16789	17659	17082	16880	17264	16820	16580	16990	17008
	$PSNR_s$ (dB)	31.05	30.18	31.02	27	32.35	30.34	31.83	28.43	30.28

5. **Conclusions.** In this paper, we proposed an adjustable RDH scheme for AMBTC compressed images based on the unilateral pixel value modification strategy. Based on our strategy, the embedding capacity can be adjusted by switching the number of embedding bits per embeddable pixel. In comparing with the existing methods, our scheme can utilize more embeddable blocks and embed more bits per embeddable pixel. The experimental results show that, compared with the other four schemes, the proposed scheme can obtain a higher embedding capacity under the condition of approximate $PSNR_S$.

Acknowledgment. This work was supported in part by the National NSF of China under Grant 61872095, Grant 61571139, Grant 61872128, in part International Scientific and Technological Cooperation of Guangdong Province under Grant 2019A050513012, in part by the Open Project Program of Shenzhen Key Laboratory of Media Security under Grant ML-2018-03, in part by Natural Science Foundation of Fujian Province under Grant 2018J01637.

REFERENCES

[1] W. Hong, Efficient data hiding based on block truncation coding using pixel pair matching technique, *Symmetry*, vol.10, no. 2, pp. 36-54, 2018.

[2] X. Liao, Q. Wen, J. Zhang, Improving the adaptive steganographic methods based on modulus function, *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. 96, no.12, pp.2731-2734, 2013.

[3] X. Liao, S. Guo, J. Yin, H. Wang, X. Li, AK. Sangaiah, New cubic reference table based image steganography, *Multimedia Tools and Applications*, vol.77, no.8, pp. 10033-10050, 2018.

[4] Z. Ni, Y. Q. Shi, N. Ansari, W. Su, Reversible data hiding, *IEEE Transactions on circuits and systems for video technology*, vol.16, no.3, pp. 354-362, 2006.

[5] D. M. Thodi and J. J. Rodriguez, Expansion embedding techniques for reversible watermarking, *IEEE Transactions on Image Processing*, vol. 16, no. 3, pp. 721-730, 2007.

[6] C. C. Chang, C. C. Lin, C. S. Tseng, et al., Reversible hiding in DCT-based compressed images, *Information Sciences*, vol. 177, no. 13, pp. 2768-2786, 2007.

[7] X. Liao, K. Li, J. Yin, Separable data hiding in encrypted image based on compressive sensing and discrete fourier transform, *Multimedia Tools and Applications*, vol.76, no.20, pp.20739-20753, 2016.

[8] B. Yang, M. Schmucker, W. Funk, et al., Integer DCT-based reversible watermarking for images using compounding technique, *Proceedings of the SPIE, Security, Steganography and Watermarking*

of Multimedia Contents, International Society for Optics and Photonics, San Jose, California, United States, 2004, vol. 5306, pp. 405-415.

[9] C. C. Chang, C. Y. Lin, Reversible steganographic method using SMVQ approach based on declustering, *Information Sciences*, vol. 177, no. 8, pp. 1796-1805, 2007.

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

[11] P. F. Shiu, W. L. Tai and J. K. Jan, et al., An interpolative AMBTC-based high-payload RDH scheme for encrypted images, *Signal Process Image Communication*, vol. 74, pp.64-77, 2019.

[12] T. Kim, Side match and overlap match vector quantizers for images, *IEEE Transactions on Image Processing*, vol.1, no.2, pp.170-185, 1992.

[13] S. A. Parah, J. A. Sheikh, N. A. Loan, G. M. Bhat, Robust and blind watermarking technique in DCT domain using inter-block coefficient differencing, *Digital Signal Processing*, vol.53, pp.11-24, 2016.

[14] X. Liao, J. Yin and S. Guo, et al., Medical JPEG image steganography based on preserving inter-block dependencies, *Computers & Electrical Engineering*, vol.67, pp. 320-329, 2018.

[15] J. C. Chuang, C. C. Chang, Using a simple and fast image compression algorithm to hide secret information, *International Journal of Computers and Applications*, vol.28, pp.329-333, 2006.

[16] J. Chen, W. Hong, T. S. Chen, C. W. Shiu, Steganography for BTC compressed images using no distortion technique, *The Imaging Science Journal*, vol.58, pp. 177-185, 2010.

[17] W. Hong, J. Chen, T. S. Chen, C. W. Shiu, Steganography for block truncation coding compressed images using hybrid embedding scheme, *International Journal of Innovative Computing Information and Control*, vol.7, pp. 733-743, 2011.

[18] J. Bai, C. C. Chang, A high payload steganographic scheme for compressed images with Hamming code, *International Journal of Network Security*, vol. 18, no. 6, pp. 1122-1129, 2016 .

[19] Y. H. Huang, C. C. Chang, Y. H. Chen, Hybrid secret hiding schemes based on absolute moment block truncation coding, *Multimedia tools and applications*, vol. 76, pp. 6159-6174, 2017.

[20] D. Ou, W. Sun, High payload image steganography with minimum distortion based on absolute moment block truncation coding, *Multimedia tools and applications*, vol. 74, pp. 9117-9139, 2015.

[21] J. Chen, W. Hong, T. S. Chen, C. W. Shiu, Steganography for BTC compressed images using no distortion technique, *The Imaging Science Journal*, vol. 58, no.4, pp.177-185, 2010.

[22] M. Tang, S. Zeng and X. Chen, et al., An adaptive image steganography using AMBTC compression and interpolation technique, *Optik*, vol.127, no.1, pp. 471-477, 2016.

[23] R. Kumar, D.S. Kim, and K.H. Jung, Enhanced ambtc based data hiding method using hamming distance and pixel value differencing, *Journal of Information Security and Applications*, vol. 47, pp. 94-103, 2019.

[24] C.H. Li, Z.M. Lu, and Y.X. Su, Reversible data hiding for btc-compressed images based on bitplane flipping and histogram shifting of mean tables, *Information technology journal*, vol. 10, no. 7, pp. 1421-1426, 2011.

[25] C. Lin and X. Liu, A reversible data hiding scheme for block truncation compressions based on histogram modification, *2012 Sixth International Conference on Genetic and Evolutionary Computing*, Kitakushu, 2012, pp. 157-160.

[26] C.I. Chang, C.Y. Hu, L. W. Chen, and C. C. Lu, High capacity reversible data hiding scheme based on residual histogram shifting for block trunca-tion coding, *Signal Processing*, vol. 108, pp. 376-388, 2015.

[27] F. Li, K. Bharanitharan, C.C. Chang, and Q. Mao, Bi-stretch reversible data hiding algorithm for absolute moment block truncation coding com-pressed images, *Multimedia Tools and Applications*, vol. 75, no. 23, pp. 16153-16171, 2016.

[28] C. C. Lin, C. C. Chang, and Z. M. Wang, Reversible data hiding scheme using adaptive block truncation coding based on an edge-based quantization approach, *Symmetry*, vol. 11, no. 6, pp. 765-791, 2019.

[29] J. C. Chuang and C. C. Chang, Using a simple and fast image compression algorithm to hide secret information, *International Journal of Computers and Applications*, vol. 28, no. 4, pp. 329-333, 2006.

[30] Y. H. Huang, C. C. Chang, and Y. H. Chen, Hybrid secret hiding schemes based on absolute moment block truncation coding, *Multimedia Tools and Applications*, vol. 76, no.6, pp. 6159-6174, 2017.

[31] Y. Y. Chen and K. Y. Chi, Cloud image watermarking: high quality data hiding and blind decoding scheme based on block truncation coding, *Multimedia Tools and Applications*, vol. 25, no.5, pp. 551-563, 2017.

[32] K. Wang, Y. Hu and Z. Lu, Reversible data hiding for block truncation coding compressed images based on prediction-error expansion, *2012 Eighth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, Piraeus, pp. 317-320, 2012.

[33] W. Sun, Z. M. Lu, Y.C. Wen, F. X. Yu, and R. J. Shen, High performance reversible data hiding for block truncation coding compressed images, *Signal Image Video Processing*, vol. 7, no. 2, pp. 297-306, 2013.

[34] W. Hong, Y. B. Ma, and H. C. Wu, An efficient reversible data hiding method for ambtc compressed images, *Multimedia Tools and Applications*, vol. 76, no. 4, pp. 5441-5460, 2017.

[35] Y. Y. Tsai, C. S. Chan, C. L. Liu, and B. R. Su, A reversible steganograph-ic algorithm for btc-compressed images based on difference expansion and median edge detector, *The Imaging Science Journal*, vol. 62, no. 1, pp. 48-55, 2014.

[36] C. C. Chang, T. S. Chen, Y. K. Wang, and Y. J. Liu, A reversible data hiding scheme based on absolute moment block truncation coding compression using exclusive or operator, *Multimedia Tools and Applications*, vol. 77, no. 7, pp. 9039-9053, 2018.

[37] W. Hong, X. Y. Zhou, and S.W. Weng, Joint adaptive coding and reversible data hiding for ambtc compressed images, *Symmetry*, vol. 10, no.7, pp. 254-267, 2018.

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

[39] J. Pan, W. Li, and C. C. Lin, Novel reversible data hiding scheme for ambtc-compressed images by reference matrix, *in Proc. MISNC*, Berlin, pp. 427-436, 2014.

[40] A. Malik, G. Sikka, and H. K. Verma, An ambtc compression based data hiding scheme using pixel value adjusting strategy, *Multidimensional Systems and Signal Processing*, vol.29, no. 4, pp. 1801-1818, 2018.

[41] C. C. Lo, Y. C. Hu, W. L. Chen, C. M. Wu, Reversible data hiding scheme for BTC-compressed images based on histogram shifting, *International Journal of Security and Its Applications*, vol. 8, no.2, pp.301-314, 2014.