Received September 5, 2021, accepted September 18, 2021, date of publication September 22, 2021, date of current version October 5, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3114495

An AMBTC Authentication Scheme With Recoverability Using Matrix Encoding and Side Match

WIEN HONG[1], JIE WU[2], DER-CHYUAN LOU[3,4], XIAOYU ZHOU[5], AND JEANNE CHEN[1]
[1]Department of Computer Science and Information Engineering, National Taichung University of Science and Technology, Taichung City 404, Taiwan
[2]School of Electrical and Computer Engineering, Nanfang College of Guangzhou, Guangzhou 501970, China
[3]Department of Computer Science and Information Engineering, Chang Gung University, Taoyuan City 333, Taiwan
[4]Stroke Center, Department of Neurology, Linkou Chang Gung Memorial Hospital, Taoyuan City 333, Taiwan
[5]School of Information Engineering, Jimei University, Xiamen 361021, China

Corresponding author: Der-Chyuan Lou (dclouprof@gmail.com)

This work was supported in part by the Ministry of Science and Technology of Taiwan under Grant MOST 109-2221-E-182-041, and in part by Chang Gung Memorial Hospital under Grant BMRPB30.

ABSTRACT This paper presents an authentication scheme with recoverability based on Absolute moment block truncation coding (AMBTC). Previous works do not take into account characteristics of the AMBTC compressed codes. Therefore, the qualities of marked and recovered images are not satisfactory. In this paper, we generate the recovery codes according to block similarities. The recovery codes are scrambled and stored in the bitmaps of smooth blocks using Matrix encoding. The bits in the bitmap are flipped to obtain a set of authentication codes. The code that causes the least error is embedded into the quantization values using the adaptive pixel pair matching technique (APPM). For those complex blocks, the authentication codes are generated from the bitmaps, and are embedded into the quantization values. After authenticating, the tampered blocks are recovered by untampered ones with the same cluster information. Some blocks that cannot be recovered in the previous stage are recovered by using the Side match technique. The experimental results show that our method provides better image quality and recoverability than prior works.

INDEX TERMS AMBTC, side match, APPM, matrix encoding.

I. INTRODUCTION

Digital images are one of the most popular media on the Internet. However, with the advancement of retouching software and computing performance of computers, images are susceptible to tampering during the transmission process, causing the authenticity of images to be questioned. Therefore, authenticating the integrity of images becomes a very important issue. The objective of our work is to propose an efficient authentication method to detect image tampering with the capability to recover the tampered parts.

Image authentication can be divided into two major types depending on the detection method. The first type uses the statistical properties to detect the image authenticity [1]–[3]. Because the image itself has some natural statistical features, they will be changed if the image is tampered with. Methods of this type use the change of such features to detect whether the image has been tampered with. These methods do not require to alter pixels of images and thus no damage to the image quality. However, their detection results are less accurate and they are not possible to repair tampered regions.

The second type is termed as the watermarking method, which protects the image by embedding authentication data into pixels. Methods of this type can be categorized into robust, fragile and semi-fragile watermarking. Robust watermarking should be capable of resisting intentional or unintentional image editing, so it is often used for copyright protection [4], [5]. Semi-fragile watermarking is not sensitive to minor image modifications (e.g., JPEG compression), but is sensitive to malicious tampering [6]. In contrast, fragile watermarking is sensitive to any tampering [7]–[25], which protects the image to the largest extent. For the fragile watermarking method, since the embedded authentication code is fragile, tampering with the image will also destroy the authentication code. This will cause the extracted code to be different from the original embedded one. Therefore, it is

The associate editor coordinating the review of this manuscript and approving it for publication was Abdullah Iliyasu.

possible to determine whether the image has been tampered with or not. This technique requires embedding the authentication code into the image, and the quality of the image will be degraded inevitably. Nevertheless, it has a better detection effect and some of methods [9]–[12] can also recover the tampered regions roughly.

Fragile watermarking methods can be divided into two categories according to their domain of application: spatial domain [7]–[11] and compressed domain [12]–[25]. Spatial domain techniques embed authentication codes into pixel values by data embedding techniques [26]–[30]. Because spatial domain images provide a larger amount of redundancy, they have a higher payload and can be embedded with a larger amount of information. Therefore, many spatial domain image authentication techniques also provide a recovery capability [10], [11] to repair the tampered regions. However, most of images on the Internet are transmitted in their compressed format. Therefore, image authentication techniques in the compressed domain have attracted a lot of research, such as those for JPEG [13]–[14], VQ [15] and AMBTC [16]–[25]. The AMBTC compression method is proposed by Lema and Mitchell [32], which compresses a block of images into two quantization values and a bitmap. Since AMBTC requires less computation yet provides relatively acceptable image quality, some studies [16]–[25] have started to develop AMBTC-based image authentication techniques in recent years.

Li *et al.* [18] propose an AMBTC image authentication method in which they use a reference table to fill authentication codes into the bitmaps. The length of authentication codes can be adjusted so that a balance between image quality and detection accuracy can be achieved. Lin *et al.* [19] propose a hybrid AMBTC image authentication method by dividing the image into smooth and complex blocks, and use them to generate authentication codes. Different techniques are applied to embed authentication codes into bitmap or quantization values respectively to improve the image quality. Chen and Chang [20] improve the security of [18] so that the bitmaps of [18] can be protected without loss of image quality. In Hong *et al.*'s method [21], the most significant bit (MSB) of quantization values is adjusted to obtain different authentication codes, and embed them into the least significant bit (LSB) of quantization values. The distortions of the image block are evaluated, and the code that causes the least distortion will be selected and embedded in the LSB. Su *et al.*'s method [23] uses Matrix encoding [31] technique to embed the authentication codes into bitmaps. Instead of flipping a bit in bitmap, their method embeds the position of to-be-flipped bit into the quantization values and yields a good image quality. Chen *et al.* [22] propose an AMBTC image authentication technique based on the Turtle shell reference table. In their method, authentication codes are embedded into the quantization values using an iterative method to provide high image quality and detectability.

Although the above mentioned methods [18]–[23] achieve the purpose of tampering detection, they are unable to recover the tampered regions. This is because authentication methods with recovery capabilities require additional space to embed the recovery codes, which is a challenge task for compressed images that have limited embedding space. Hu *et al.* [24] first propose a recoverable AMBTC image authentication technique. They use two quantization values to carry the authentication codes, and use bitmaps of smooth blocks to embed the recovery codes. An adjustment mechanism is applied to reduce the embedding error caused by the authentication code embedment. The recovery code of a block is the average value of pixels in the block. Multiple copies of the recovery code are embedded to reduce the number of unrecoverable blocks. Based on [24], Hong *et al.* [25] propose a recoverable AMBTC image authentication technique. Unlike [24], Hong *et al.* use the cluster information of the image blocks as the recovery codes. The recovery codes are embedded into the bitmap of smooth blocks, while the authentication codes are embedded in the LSB of the quantized values using MSB perturbation. The detected tampered blocks are restored with the average of the same group of untampered blocks. Although the image quality of [25] is higher than that of [24], the recovery code of [25] is embedded using a method similar to the LSB replacement. Therefore, the increase in image quality is limited. Furthermore, the methods of [24] and [25] do not effectively use the neighboring blocks to recover a tampered one that could not be repaired in the primary recovery. Therefore, some tampered blocks cannot be recovered or recovered blocks have a less natural appearance.

In this paper, the proposed method embeds recovery codes into bitmaps using Matrix encoding. To embed 6-bit recovery code, only two out of 14 bits in the bitmap are required to be modified. By flipping bits in the bitmap, the authentication code that causes the least distortion is embedded into quantization values using the APPM. After authenticating, most tampered blocks can be detected and recovered. The Side match technique is then applied to find the most suitable blocks to recover those unrecoverable ones at the primary recovery stage. The experiments show that not only the marked image quality is higher, but also the visual effect of recovered regions is better than prior works.

The rest of the paper is organized as follows. Section II introduces the AMBTC compression, Matrix encoding and APPM embedding methods. Section III introduces the proposed method in this paper, while Sections IV and V present the experimental results and conclusions, respectively.

II. RELATED WORKS

In this section, we introduce three related techniques, including AMBTC compression, Matrix encoding, and APPM embedding methods. We will use the Matrix encoding and APPM methods to embed the recovery codes and authentication codes into the bitmaps and quantization values respectively.

A. AMBTC COMPRESSION METHOD

The AMBTC compression method compresses an image block of size $n \times n$ into two quantization values and a bitmap. Let O be the original image, and O is partitioned into N blocks of size $n \times n$, i.e., $O = \{O_i\}_{i=0}^{N-1}$. Let m_i be the mean value of the block O_i, and $O_{i,j}$ be the $j-$th pixel of O_i, i.e., $O_i = \{O_{i,j}\}_{j=0}^{n\times n-1}$. Averaging the pixels in O_i with values less than or equal to m_i, we obtain the lower quantization value a_i. The upper quantization value b_i is obtained by averaging the pixels with values greater than m_i. To obtain a bitmap, we first prepare an empty bitmap $B_i = \{B_{i,j}\}_{j=0}^{n\times n-1}$, and visit each pixel in O_i in raster scanning order. If $O_{i,j} \le m_i$, then $B_{i,j} = 0$. Otherwise, $B_{i,j} = 1$. All blocks are processed in the same way and finally we can get the AMBTC compressed codes $\{a_i, b_i, B_i\}_{i=0}^{N-1}$ for O. When decompressing, prepare a matrix I of the same size as O, and partition I into N blocks of size $n \times n$, i.e., $I = \{I_i\}_{i=0}^{N-1}$. For each $\{a_i, b_i, B_i\}$, $I_{i,j} = a_i$ if $B_{i,j} = 0$, and $I_{i,j} = b_i$ if $B_{i,j} = 1$. Each compressed code is processed in the same way and the final decompressed image I is obtained. Here is a simple example. Let O_i be a block of size 4×4 with values [35, 34, 56, 12; 76, 85, 21, 75; 32, 97, 13, 67; 88, 70, 99, 80]. Then $m_i = 58.75$, $a_i = 29$, $b_i = 82$, and $B_i = [0000; 1101; 0101; 1111]$. The decompressed block [29, 29, 29, 29; 82, 82, 29, 82; 29, 82, 29, 82; 82, 82, 82, 82] can be easily calculated via the decompression procedures.

B. MATRIX ENCODING

Matrix encoding is an embedding technique developed based on Hamming code [31]. The $(1, \eta, k)$ Matrix encoding is possible to embed $k-$bit message msg into a $\eta-$bit binary cover vector $c_v = \{\beta_i\}_{i=0}^{\eta-1}$ by flipping one bit in c_v at most, where $\eta = 2^k - 1$ and β_i is the $i-$th bit in c_v. We can describe the algorithm of Matrix encoding as a function $m_v = \mathrm{me}(c_v, msg)$, where m_v is the embedded result (marked vector). Matrix encoding embeds messages using a parity check matrix H, and each column of H is composed of binary bits with its decimal value ranging from 1 to η. For example, in the Matrix encoding of $(1, 7, 3)$, H can be expressed as

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \quad (1)$$

To embed msg into c_v, we first calculate $(H^* c_v^T) \oplus msg^T$, where T, $\oplus$ and * represent the transpose, exclusive-or, and mouldo-2 operators, respectively. Let the calculated result be ρ of $1 \times k-$bit and $(\rho)_{10}$ be the decimal value of ρ. If $(\rho)_{10} = 0$, then $m_v = c_v$. Otherwise, flip the $((\rho)_{10} - 1)-$th bit in c_v, and the marked vector m_v can be obtained. The message msg embedded in m_v can be extracted by $msg = H^* m_v^T$. Let's take an example to illustrate the Matrix encoding technique. Suppose the $(1, 7, 3)$ Matrix encoding is employed to embed $msg = [1, 0, 1]$ into $c_v = [1, 1, 1, 0, 1, 1, 0]$. Because $\rho = (H^* c_v^T) \oplus msg^T = [1, 1, 0]$, we flip the $6 - 1 = 5-$th bit of c_v to obtain $m_v = [1, 1, 1, 0, 1, 0, 0]$. The embedded message msg in m_v can be extracted by calculating $msg = H^* m_v^T = [1, 0, 1]$.

C. ADAPTIVE PIXEL PAIR MATCHING

The APPM embedding method [29] uses two pixels to embed a digit of base λ using a reference table R_λ of size 256×256, which is equivalent to embed $\log_2 \lambda-$bit message. The value of reference table R_λ at position (a, b) can be generated by the following equation:

$$R_\lambda(a, b) = (a + c_\lambda \times b) \bmod \lambda, \quad 0 \le a \le 255, \ 0 \le b \le 255, \quad (2)$$

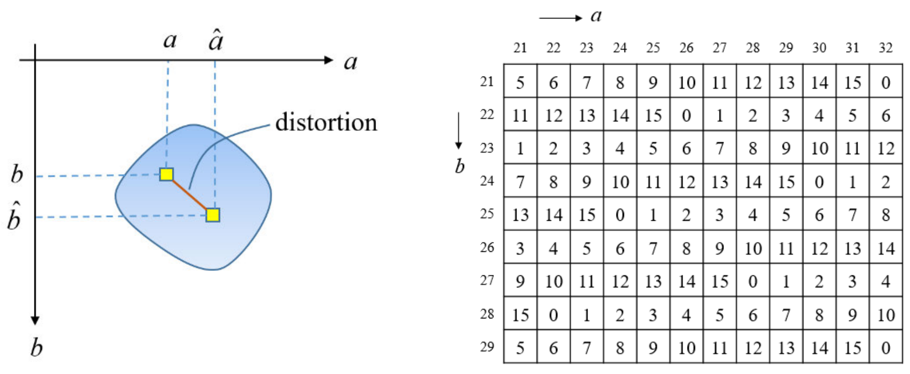
where c_λ is a constant. When $\lambda = 4$, 16, 64 and 256, c_λ are 2, 6, 14 and 60, respectively. To embed a digit d_i of base λ into a pixel pair (a_i, b_i), we first find a location $(\hat{a}_i, \hat{b}_i)$ around $R_\lambda(a_i, b_i)$ such that $R_\lambda(\hat{a}_i, \hat{b}_i) = d_i$, and the distance from $(\hat{a}_i, \hat{b}_i)$ to (a_i, b_i) is the shortest. The found pixel pair $(\hat{a}_i, \hat{b}_i)$ is then used to replace (a_i, b_i), as shown in Fig. 1 (a). Since the reference table R_λ is known at the extraction stage, we can extract d_i by querying the value of reference table R_λ at position $(\hat{a}_i, \hat{b}_i)$.

Here is a simple example. Suppose a pixel pair $(a_i, b_i) = (30, 27)$ is to be embedded a digit $d_i = 12$ with base $\lambda = 16$. Using Eq. (2), we obtain the reference table R_{16}, and part of this table is shown in Fig. 1 (b). Because $R_{16}(30, 26) = 12$, and $(30, 26)$ is the closest to $(30, 27)$, the value of $(30, 27)$ is modified to $(\hat{a}_i, \hat{b}_i) = (30, 26)$ after embedding the message $d_i = 12$. The embedded digit $d_i = 12$ can be extracted by querying the value at $R_{16}(30, 26)$.

III. THE PROPOSED METHOD

In the past two related works, Hu *et al.*'s method [24] uses random numbers to generate the authentication code, which results in some intentional tampering cannot be detected. Furthermore, the embedding technique used in Hu *et al.*'s method is not effective in reducing the damage to the pixel values, and thus the quality of the marked image is relatively low. In addition, their method uses the average value of the image blocks as the recovery code to repair the tampered blocks, which also results in a mosaic-like effect on the recovered regions. Hong *et al.* [25] improve the shortcomings of [24], so that the quality of marked and recovered images is higher than [24] and the malicious tampering can be detected. However, their method uses LSB substitution for embedding the authentication codes and perturbs the MSB value to find the minimum distortion. Besides, it has been shown in [29] that the error caused by the LSB-like embedding method is larger than that of the APPM. Meanwhile, when embedding the recovery code, Hong *et al.*'s method adopts an approach by directly replacing the bitmap with the recovery code, resulting in a larger distortion of the marked images. Finally, for those un-recoverable blocks, Hong *et al.*'s method only uses information of the surrounding blocks to restore them, which will make the recovery of edge or complex blocks ineffective.

In this paper, we propose an AMBTC compression authentication scheme that can approximately recover tampered images with a tampering rate of less than 20%. In our method,



(a) Shortest distance searching

$b \backslash a$	21	22	23	24	25	26	27	28	29	30	31	32
21	5	6	7	8	9	10	11	12	13	14	15	0
22	11	12	13	14	15	0	1	2	3	4	5	6
23	1	2	3	4	5	6	7	8	9	10	11	12
24	7	8	9	10	11	12	13	14	15	0	1	2
25	13	14	15	0	1	2	3	4	5	6	7	8
26	3	4	5	6	7	8	9	10	11	12	13	14
27	9	10	11	12	13	14	15	0	1	2	3	4
28	15	0	1	2	3	4	5	6	7	8	9	10
29	5	6	7	8	9	10	11	12	13	14	15	0

(b) Reference table R_{16}

FIGURE 1. Examples of embedding and extraction of the APPM.

the AMBTC image is divided into blocks of size 4 × 4. Then these blocks are classified into 64 groups according to their similarity using k-means algorithm [33]. Then an index table is used to record these groupings, and each index value in the table is a number between 0 and 63. These numbers are scrambled and then embedded into the bitmap of the smooth blocks using Matrix encoding. After embedding, the two bits that do not participate in Matrix encoding are flipped. A set of authentication codes is generated by hashing the flipped bitmaps, and the one that causes the minimum distortion is embedded using APPM into quantization values of the smooth block. Because the distortion caused by APPM is even smaller than that of the LSB-like embedment, our method can achieve a better image quality.

For complex blocks, since modifying the bitmaps will cause a larger distortion, the bitmaps are not employed to embed the recovery codes. However, they are used to generate the authentication codes, which are then embedded in the quantization values using APPM. After detection, the tampered blocks are known by comparing the regenerated authentication codes with the ones that extracted from the quantization values. The tampered blocks are recovery by averaging those untampered blocks with the same cluster. In some cases, the original cluster information of tampered blocks may be lost. If this happens, the Side match technique is used to estimate the cluster index of these blocks. The detailed description of the proposed method is discussed in the following subsections.

A. GENERATION OF THE RECOVERY CODES

In our method, we use the cluster information of image blocks as their recovery codes. There is a lower chance that image blocks belonging to the same group will be tampered with at the same time when recovering tampered blocks. Therefore, we can use the average of blocks that have not been tampered with and belong to the same group to recover the tampered ones. Let I_O be the original image, and after AMBTC compression, we have the compression codes $\{a_i, b_i, B_i\}_{i=0}^{N-1}$. By decompressing the AMBTC code, we will obtain a decompressed image I. Let I_i be the decompressed block after decompressing the i−th code $\{a_i, b_i, B_i\}$, and $I_{i,j}$ be the j−th pixel of I_i. Therefore, we have $I = \{I_i\}_{i=0}^{N-1}$ and $I_i = \{I_{i,j}\}_{j=0}^{15}$. By employing k-means algorithm [33] to cluster $\{I_i\}_{i=0}^{N-1}$, we obtain the cluster information $\{\varphi_i\}_{i=0}^{N-1}$, which is utilized as the recovery codes.

To perform the k-means algorithm, we randomly select 64 blocks from $\{I_i\}_{i=0}^{N-1}$ as the initial codebook $C = \{C_u\}_{u=0}^{63}$. By comparing the distances between I_i and $\{C_u\}_{u=0}^{63}$, I_i is categorized to group k if I_i has the nearest distance to C_k for $0 \le k \le 63$ and $0 \le i \le N$. Then, blocks of the same group are averaged to generate an updated codebook $C' = \{C'_u\}_{u=0}^{63}$. This process is repeated several times and the final codebook C_f can be obtained. Finally, I_i is coded by φ_i if I_i has the shortest distance to C_{φ_i}. Note that φ_i stands for the image block I_i which is classified into the φ_i−th group, and $0 \le \varphi_i \le 63$. In the next section, we will use the Matrix encoding technique to embed $\{\varphi_i\}_{i=0}^{N-1}$ into the bitmaps of smooth blocks.

B. EMBEDMENT OF AUTHENTICATION AND RECOVERY CODES

The recovery code of a block must be stored in another block to prevent the loss of recovery information due to tampering. This can be done by using a scramble function to randomly assign a block to carry another blocks' recovery information. Using the key κ_S, the scramble function $n = s(i)$ can obtain the next embedding position n of the current position i, where $s(i) \neq s(j)$ if $i \neq j$ and $s(i) \neq i$. When repairing a block, we need to know where the recovery code of the block is hidden. The previous position p of i can also be obtained using the inverse-scramble function $p = s^{-1}(i)$. Therefore, we have $i = s^{-1}(n)$ and $i = s(p)$. We illustrate the scramble function $n = s(i)$ and inverse-scramble function $p = s^{-1}(i)$ in Fig. 2.

Since recovery code of a block is embedded in the bitmap of a smooth block, we must distinguish whether a block is smooth or complex. Let T_0 be a pre-defined threshold. if $b_i - a_i \le T_0$, then the block I_i is considered as a smooth block, otherwise it is a complex one. If $I_{s(i)}$ is a smooth block,

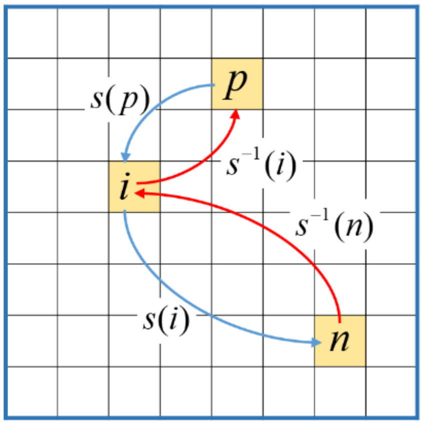


FIGURE 2. Scramble function $n = s(i)$ and inverse-scramble function $p = s^{-1}(i)$.

the cluster information φ_i of block I_i will be embedded into the bitmap $B_{s(i)}$ of block $I_{s(i)}$. Similarly, if I_i is a smooth block, then the cluster information $\varphi_{s^{-1}(i)}$ will be embedded into bitmap B_i. Assume that I_i is a smooth block, and we will use B_i to embed φ_p, where $p = s^{-1}(i)$. Firstly, the bitmap B_i of block I_i is divided into two areas, the first 14 bits are in the embedding area. In this area, we will use Matrix encoding to embed 6-bit recovery code. The other two bits are located in the flipping area, and we will use the bit-flipping technique (BFT) to minimize embedding distortion. We divide the bits $\{B_{i,j}\}_{j=0}^{13}$ in the embedding area into $\{B_{i,j}\}_{j=0}^{6}$ and $\{B_{i,j}\}_{j=7}^{13}$, and convert φ_p into its binary representation $(\varphi_p)_2$ of 6 bits. We use $(\varphi_p^0)_2$ and $(\varphi_p^1)_2$ to represent the first and last 3 bits of $(\varphi_p)_2$, respectively. The Matrix encoding is then applied to embed $(\varphi_p^0)_2$ and $(\varphi_p^1)_2$ into $\{B_{i,j}\}_{j=0}^{6}$ and $\{B_{i,j}\}_{j=7}^{13}$ to obtain $\{\hat{B}_{i,j}\}_{j=0}^{6}$ and $\{\hat{B}_{i,j}\}_{j=7}^{13}$, respectively.

In our method, the authentication code of smooth block I_i is generated by hashing the bitmap of I_i and the location information i. Therefore, modifying the value of the two bits $\{B_{i,j}\}_{j=14}^{15}$ in the flipping area will generate different authentication codes. Embedding these codes into a_i and b_i using the APPM method will also result in different distortions. Therefore, the BFT flips two bits $\{B_{i,j}\}_{j=14}^{15}$ in the flipping area into four different combinations (00_2, 01_2, 10_2, and 11_2) and obtains $\{\hat{B}_{i,j}^k\}_{j=14}^{15}$, where $0 \le k \le 3$. The marked bitmap $\hat{B}_i^k = \{\hat{B}_{i,j}\}_{j=0}^{6}||\{\hat{B}_{i,j}\}_{j=7}^{13}||\{\hat{B}_{i,j}^k\}_{j=14}^{15}$ and the block location i are hashed using MD5 [34], and fold the hashed result into $\alpha-$bit authentication code using the XOR operation, where the symbol $||$ is the concatenation operator. The APPM is then applied to embed the authentication code ac_i into a_i and b_i. The lower and higher quantization values after embedment are denoted by $\{\hat{a}_i^k\}_{k=0}^{3}$ and $\{\hat{b}_i^k\}_{k=0}^{3}$, respectively. With $\{\hat{a}_i^k\}_{k=0}^{3}$, $\{\hat{b}_i^k\}_{k=0}^{3}$ and $\{\hat{B}_i^k\}_{k=0}^{3}$, four AMBTC blocks $\{\hat{I}_i^k\}_{k=0}^{3}$ can be decoded. By comparing $\{\hat{I}_i^k\}_{k=0}^{3}$ and I_i, we can select the best $\{\hat{a}_i, \hat{b}_i, \hat{B}_i\}$ among $\{\hat{a}_i^k, \hat{b}_i^k, \hat{B}_i^k\}_{k=0}^{3}$ with the smallest distortion as the marked AMBTC code. The process of finding the minimum distortion mentioned above can be described as an optimization problem:

$$\text{Minimize:} \quad \sum_{j=0}^{15}(\hat{I}_{i,j}^k - I_{i,j})^2, \tag{3}$$

$$\text{Subject to:} \quad \{\hat{B}_{i,j}\}_{j=0}^{6} = \text{me}(\{B_{i,j}\}_{j=0}^{6}, (\varphi_p^0)_2), \tag{4}$$

$$\{\hat{B}_{i,j}\}_{j=7}^{13} = \text{me}(\{B_{i,j}\}_{j=7}^{13}, (\varphi_p^1)_2), \tag{5}$$

$$\hat{B}_i^k = \{\hat{B}_{i,j}\}_{j=0}^{6}||\{\hat{B}_{i,j}\}_{j=7}^{13}||\{\hat{B}_{i,j}^k\}_{j=14}^{15}, \tag{6}$$

$$ac_i^k = \text{fold}(\text{md5}(\hat{B}_i^k, i), \alpha), \tag{7}$$

$$\hat{a}_i^k, \hat{b}_i^k = \text{appm}(a_i, b_i, ac_i^k), \tag{8}$$

$$\hat{I}_i^k = \text{ambtc}(\hat{a}_i^k, \hat{b}_i^k, \hat{B}_i^k), \tag{9}$$

$$\left| \hat{b}_i^k - \hat{a}_i^k \right| \le T_0, \tag{10}$$

$$0 \le k \le 3. \tag{11}$$

The fold(x, α) in Eq. (7) represents a function that folds a bitstream x into α bits, and the appm() and ambtc() functions denote the APPM and the AMBTC techniques, respectively. By solving the above equations, we can obtain a set of optimal solutions $\hat{a}_i$, $\hat{b}_i$ and $\hat{B}_i$ such that the corresponding decompressed block has the minimum distortion.

When $b_i - a_i > T_0$, I_i is a complex block. The bitmap of a complex block does not carry a recovery code, thus the bitmap keeps unaltered. Hash B_i and location information i using MD5, and fold the hashed result to $\alpha-$bit authentication code ac_i. The APPM method is then applied to embed ac_i into a_i and b_i to obtain $\hat{a}_i$ and $\hat{b}_i$ by ensuring $\left| \hat{b}_i - \hat{a}_i \right| > T_0$. The AMBTC code of the complex block after embedment is denoted as $\{\hat{a}_i, \hat{b}_i, \hat{B}_i\}$, where $\hat{B}_i = B_i$. Image blocks are processed using the aforementioned method based on the block smoothness, and the final marked AMBTC code $\{\hat{a}_i, \hat{b}_i, \hat{B}_i\}_{i=0}^{N-1}$ can be obtained. Followings are the list of algorithms of our embedding method:

Input: Original image I_O
Output: Marked AMBTC codes $\{\hat{a}_i, \hat{b}_i, \hat{B}_i\}_{i=0}^{N-1}$

Step 1: Partition the original image I_O into N blocks of size 4×4.

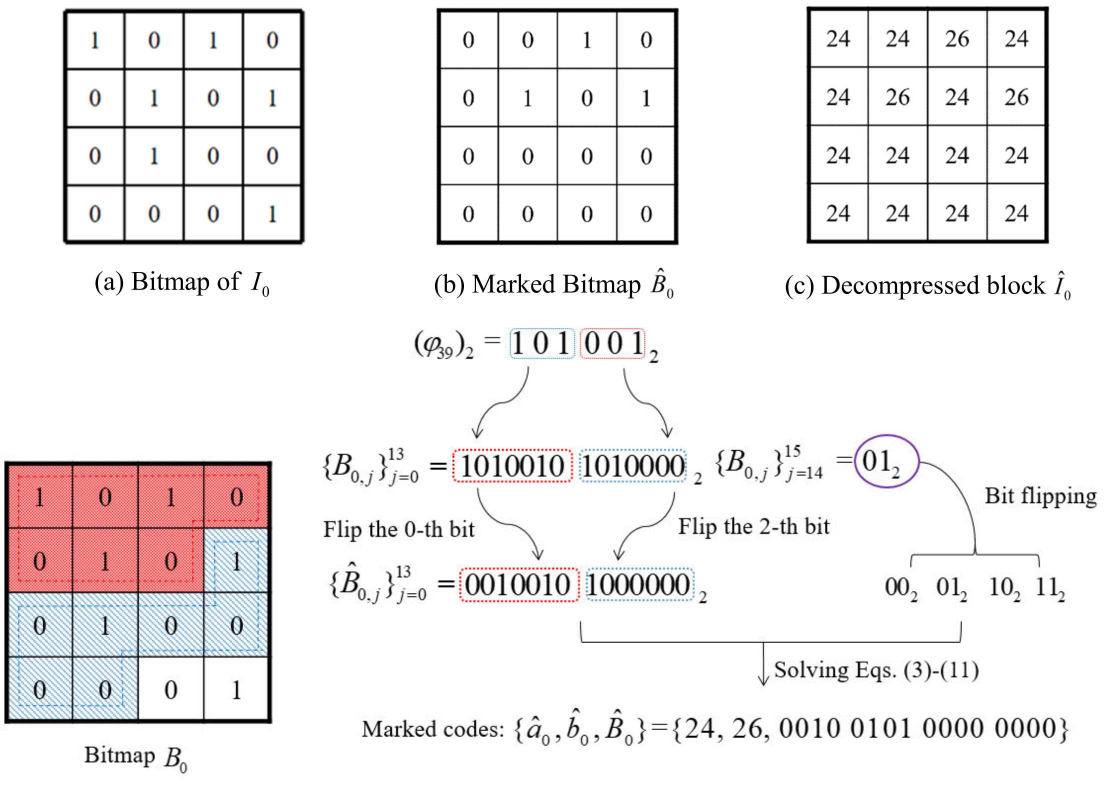
Step 2: Generate the AMBTC codes $\{a_i, b_i, B_i\}_{i=0}^{N-1}$ of I_O, and subsequently we obtain the AMBTC decompressed image $I = \{I_i\}_{i=0}^{N-1}$, as described in Section II.A.

Step 3: Use the k-means algorithm to generate cluster information $\{\varphi_i\}_{i=0}^{N-1}$ from $I = \{I_i\}_{i=0}^{N-1}$, as described in Section III.A.

Step 4: Use key κ_S to scramble $\{\varphi_i\}_{i=0}^{N-1}$, we obtain $\{\varphi_p\}_{p=0}^{N-1}$, $p = s^{-1}(i)$.

Step 5: If $b_i - a_i \le T_0$, use the Matrix encoding to embed recovery codes φ_p into $\{B_{i,j}\}_{j=0}^{6}$ and $\{B_{i,j}\}_{j=7}^{13}$, then we obtain $\{\hat{B}_{i,j}\}_{j=0}^{6}$ and $\{\hat{B}_{i,j}\}_{j=7}^{13}$. If $b_i - a_i > T_0$, we keep B_i unaltered.

Step 6: Generate authentication codes and embed them into a_i and b_i using the APPM, as described in Section III.B.


1 0 1 0 / 0 1 0 1 / 0 1 0 0 / 0 0 0 1
0 0 1 0 / 0 1 0 1 / 0 0 0 0 / 0 0 0 0
24 24 26 24 / 24 26 24 26 / 24 24 24 24 / 24 24 24 24
$(\varphi_{39})_2$ = 1 0 1 0 0 1$_2$
$\{B_{0,j}\}_{j=0}^{13}$ = 1010010 1010000$_2$
$\{B_{0,j}\}_{j=14}^{15}$ = 01$_2$
Bit flipping
Flip the 0-th bit
Flip the 2-th bit
$\{\hat{B}_{0,j}\}_{j=0}^{13}$ = 0010010 1000000$_2$
00$_2$ 01$_2$ 10$_2$ 11$_2$
Solving Eqs. (3)-(11)
Marked codes: $\{\hat{a}_0, \hat{b}_0, \hat{B}_0\}$={24, 26, 0010 0101 0000 0000}
Bitmap B_0

(a) Bitmap of I_0 (b) Marked Bitmap $\hat{B}_0$ (c) Decompressed block $\hat{I}_0$

(d) The procedures of embedment

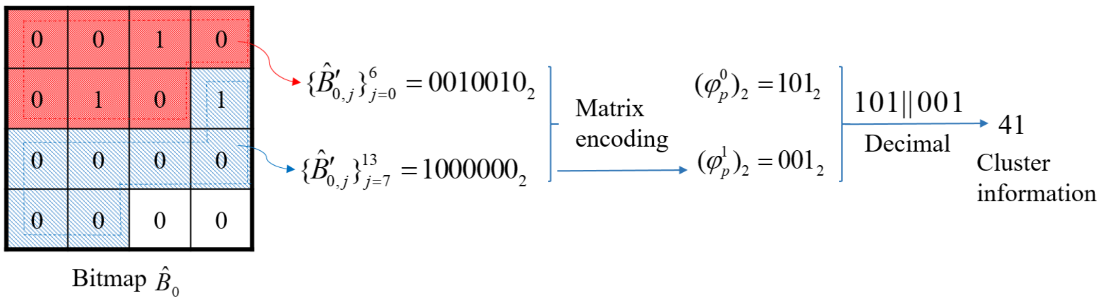
FIGURE 3. Embedment of authentication and recovery codes.

Step 7: Repeat Steps 5-6 until all blocks are processed, and we obtain the marked AMBTC codes $\{\hat{a}_i, \hat{b}_i, \hat{B}_i\}_{i=0}^{N-1}$.

The following is an example to illustrate how to embed authentication and recovery codes for smooth and complex blocks. Let I_0 be an AMBTC image block with the code $\{a_0, b_0, B_0\} = \{23, 26, 1010\ 0101\ 0100\ 0001\}$, and the bitmap B_0 is shown in Fig. 3 (a). Suppose $\alpha = 4$ and $T_0 = 4$. Because $b_0 - a_0 \leq T_0$, I_0 is a smooth block. Assume the cluster information $\varphi_{39} = 41$ of block I_{39} is to be embedded into the bitmap B_0 of block I_0. To perform the embedment, bitmap B_0 is divided into $\{B_{0,j}\}_{j=0}^{6} = 1010010_2$ and $\{B_{0,j}\}_{j=7}^{13} = 1010000_2$. Convert φ_{39} into its binary representation, we have $(\varphi_{39})_2 = 101001_2$. The first 3 bits $(\varphi_{39}^0)_2 = 101_2$ and the last 3 bits $(\varphi_{39}^1)_2 = 001_2$ of φ_{39} are respectively embedded into $\{B_{0,j}\}_{j=0}^{6}$ and $\{B_{0,j}\}_{j=7}^{13}$ using the Matrix encoding. Since $((H^*\{B_{0,j}\}_{j=0}^{6}) \oplus (\varphi_{39}^0)_2)_{10} - 1 = 0$ and $((H^*\{B_{0,j}\}_{j=7}^{13}) \oplus (\varphi_{39}^1)_2)_{10} - 1 = 2$, we flip the zero-th bit in $\{B_{0,j}\}_{j=0}^{6}$ and the second bit in $\{B_{0,j}\}_{j=7}^{13}$. Finally, we obtain $\{\hat{B}_{0,j}\}_{j=0}^{6} = 0010010_2$ and $\{\hat{B}_{0,j}\}_{j=7}^{13} = 1000000_2$.

By flipping the bits in $\{B_{0,j}\}_{j=14}^{15}$, four flipped cases $00_2, 01_2, 10_2$ and 11_2 are obtained. Then, we utilize the marked bitmaps 0010010||1000000||00, 0010010||1000000||01, 0010010||1000000||10, and 0010010||1000000||11 to generate four sets of authentication codes using Eq. (7). Suppose the generated authentication codes with these four cases are 0110_2, 1001_2, 0010_2 and 1101_2. These four generated codes are to be embedded into quantization values using APPM, and the one that causes the least distortion will be selected as the final authentication code. To embed $0110_2 = 6$, because $R_{16}(24, 26) = 6$ and $(24, 26)$ has the shortest distance to $(23, 26)$ (see Fig. 1(b)), we obtain $\hat{a}_0 = 24$ and $\hat{b}_0 = 26$. Other authentication codes can be embedded in the same way, and finally we obtain four sets of different quantization values and subsequently four decompressed blocks are obtained. Assuming that the MSE of the decompressed block with I_0 is minimal when the code 0110_2 is embedded. Therefore, we choose $\{\hat{B}_{0,j}\}_{j=14}^{15} = 00_2$($\hat{B}_0$ is shown in Fig. 3 (b)), and the marked AMBTC codes are $\{\hat{a}_0, \hat{b}_0, \hat{B}_0\} = \{24, 26, 0010\ 0101\ 0000\ 0000\}$, as shown in Fig. 3 (c). The process of embedding the authentication and recovery codes can be found in Fig. 3 (d).

Assume that $\{a_1, b_1, B_1\} = \{22, 28, 1001\ 0010\ 0011\ 0011\}$ is the compressed code of another AMBTC image block I_1. Because $b_1 - a_1 > T_0$, I_1 is a complex one. By applying MD5 to hash the position information $i = 1$ and bitmap B_1 and folding the result, we assume that four bits authentication code $1001_2 = 9$ is obtained. Because $R_{16}(21, 27) = 9$, $(21, 27)$ has the shortest distance to $(22, 28)$, and $27 - 21 > T_0$, we obtain $\hat{a}_1 = 21$ and $\hat{b}_1 = 27$. After applying APPM



0 0 1 0
0 1 0 1
0 0 0 0
0 0 0 0
Bitmap $\hat{B}_0$
$\{\hat{B}'_{0,j}\}_{j=0}^{6} = 0010010_2$
$\{\hat{B}'_{0,j}\}_{j=7}^{13} = 1000000_2$
Matrix encoding
$(\varphi_p^0)_2 = 101_2$
$(\varphi_p^1)_2 = 001_2$
$101||001$
Decimal
41
Cluster information

FIGURE 4. The process of extracting the recovery code from a smooth block.

to embed the authentication code, the final marked code $\{\hat{a}_1, \hat{b}_1, \hat{B}_1\} = \{21,\ 27,\ 1001\ 0010\ 0011\ 0011\}$ of bock I_1 is obtained.

C. IMAGE AUTHENTICATION

To detect whether the marked AMBTC code $\{\hat{a}'_i, \hat{b}'_i, \hat{B}'_i\}$ has been tampered with, we first recalculate the α−bit authentication code $\hat{ac}_i = \text{flod}(\text{md5}(i, \hat{B}'_i), \alpha)$, and use the APPM to extract the embedded authentication code $\hat{eac}_i$ from $\hat{a}'_i$ and $\hat{b}'_i$. If $\hat{eac}_i = \hat{ac}_i$, it means that AMBTC code $\{\hat{a}'_i, \hat{b}'_i, \hat{B}'_i\}$ has not been tampered with. Otherwise, the code has been tampered with. All AMBTC codes $\{\hat{a}'_i, \hat{b}'_i, \hat{B}'_i\}_{i=0}^{N-1}$ are processed in the same way, and eventually we can detect which blocks have been tampered with. The above detection process is termed as the primary detection. In our method, a secondary detection is also utilized to refine the detection result. In general, most of the tampered regions are clustered. Hence, in the secondary detection, if an un-tampered block is surrounded by two or more tampered ones, the block is more likely to have been tampered with. Therefore, the block will be re-judged to be tampered.

D. RECOVERY OF TAMPERED BLOCKS

After authenticating the marked AMBTC codes $\{\hat{a}'_i, \hat{b}'_i, \hat{B}'_i\}_{i=0}^{N-1}$, we can tell which blocks have been tampered with or not. Because we will use the cluster information to recover tampered ones, we must first extract the cluster information embedded in untampered blocks.

Let $\{\hat{a}'_u, \hat{b}'_u, \hat{B}'_u\}_{u=0}^{N_u-1}$ be all untampered AMBTC codes and $\{\hat{I'}_u\}_{u=0}^{N_u-1}$ are the decompressed blocks of these codes, where N_u is the total number of untampered blocks. The cluster information φ_p of block $\hat{I}'_p$ is stored in bitmap $\hat{B}'_u$, where $p = s^{-1}(u)$. Extract $\{\hat{B'}_{u,j}\}_{j=0}^{6}$ and $\{\hat{B'}_{u,j}\}_{j=7}^{13}$ from $\hat{B}'_u$ and decode them using Matrix encoding, we obtain $(\varphi_p^0)_2$ and $(\varphi_p^1)_2$, respectively. Convert $(\varphi_p^0)_2||(\varphi_p^1)_2$ to its decimal representation, we obtain the cluster information φ_p embedded in $\hat{B}'_u$, which is the cluster information of block $\hat{I}'_p$.

We continue with the above example to illustrate the recovery code extraction process. Assuming that the authentication code $\hat{eac}_0$ extracted from $\{\hat{a}'_0, \hat{b}'_0, \hat{B}'_0\}$ is the same as the regenerated one $\hat{ac}_0$, it means that block I'_0 has not been tampered with. In this case, extract $\{\hat{B}'_{0,j}\}_{j=0}^{6} = 0010010_2$ and $\{\hat{B}'_{0,j}\}_{j=7}^{13} = 1000000_2$ from $\hat{B}'_0$, and apply Matrix encoding to decode them, we obtain $(\varphi_p^0)_2 = 101_2$ and $(\varphi_p^1)_2 = 001_2$. Convert $(\varphi_p^0)_2\,||\,(\varphi_p^1)_2$ into its decimal representation, we know that the cluster information embedded in $\hat{B}'_0$ is 41. By applying the inverse-scramble function, we obtain $p = 39$ and the cluster information $\varphi_{39} = 41$ of block I'_{39} is known. Finally, since I'_1 is a complex block, no recovery codes are embedded and need not to be extracted. The process of extracting the recovery code from an untampered smooth one is shown in Fig. 4.

By visiting blocks $\{\hat{B}'_u\}_{u=0}^{N_u-1}$, we can extract N_u−th cluster information $\{\varphi_{s^{-1}(u)}\}_{u=0}^{N_u-1}$ of blocks $\{\hat{I'}_{s^{-1}(u)}\}_{u=0}^{N_u-1}$. Because blocks with the same cluster information are more similar to each other, we can average those blocks that have not been tampered with and have the same cluster information. Therefore, we obtain 64 blocks $\{\bar{I}_r\}_{r=0}^{63}$ of size 4×4, where $\bar{I}_r$ represents the average of all blocks with cluster information r.

To recover the tampered block $\hat{I}_t$, we know that the cluster information φ_t of $\hat{I}_t$ is embedded in bitmap $\hat{B}_{s(t)}$ using key κ_S. Extract $\{\hat{B}_{s(t),j}\}_{j=0}^{6}$ and $\{\hat{B}_{s(t),j}\}_{j=7}^{13}$ from $\hat{B}_{s(t)}$, then use Matrix encoding to decode them and obtain $(\varphi_{s(t)}^0)_2$ and $(\varphi_{s(t)}^1)_2$, respectively. Convert $(\varphi_{s(t)}^0)_2||(\varphi_{s(t)}^1)_2$ to decimal representation, the cluster information φ_t of $\hat{I}_t$ is obtained, and finally the tampered block $\hat{I}_t$ is recovered by $\bar{I}_{\varphi_t}$. By this way, most of the tampered blocks can be recovered using the same process.

However, some blocks in $\{\bar{I}_r\}_{r=0}^{63}$ cannot be generated due to the loss of cluster information. Fortunately, unrecoverable blocks are often sparsely distributed in the tampered regions. Therefore, we can use the Side match technique to estimate the unrecoverable blocks with their surrounding ones that have been recovered previously. Let $\hat{I}_\tau$ be the block need to be recovered by using the Side match technique, and $\hat{I}_N$, $\hat{I}_W$, $\hat{I}_S$, and $\hat{I}_E$ are the north, west, south, and east blocks of $\hat{I}_\tau$, respectively, as shown in Fig. 5.

For most of natural images, the difference between of adjacent pixel values is small. Therefore, the border pixels in $\hat{I}_\tau$ can be estimated by $\hat{I}_{\tau,0} = (\hat{I}_{N,12} + \hat{I}_{W,3})/2, \hat{I}_{\tau,1} = \hat{I}_{N,13}, \hat{I}_{\tau,2} = \hat{I}_{N,14}, \hat{I}_{\tau,3} = (\hat{I}_{N,15} + \hat{I}_{E,0})/2, \hat{I}_{\tau,4} = \hat{I}_{W,7}, \hat{I}_{\tau,7} = \hat{I}_{E,4}, \hat{I}_{\tau,8} = \hat{I}_{W,11}, \hat{I}_{\tau,11} = \hat{I}_{E,8}, \hat{I}_{\tau,12} = (\hat{I}_{W,15} + \hat{I}_{S,0})/2, \hat{I}_{\tau,13} = \hat{I}_{S,1}, \hat{I}_{\tau,14} = \hat{I}_{S,2}$ and $\hat{I}_{\tau,15} = (\hat{I}_{E,12} + \hat{I}_{S,3})/2$.

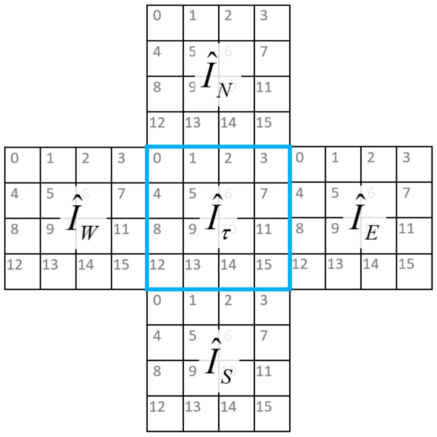


FIGURE 5. The relationship between $\hat{I}_\tau$ and neighboring blocks.

Therefore, we can extract the pixel values of blocks $\{\bar{I}_r\}_{r=0}^{63}$ at positions 0, 1, 2, 3, 4, 7, 8, 11, 12, 13, 14 and 15, and compare the squared difference between the extracted pixels and border pixels of $\hat{I}_\tau$. The block in $\{\bar{I}_r\}_{r=0}^{63}$ with the least difference is selected to recover the unrecoverable block $\hat{I}_\tau$. Finally, we obtain the recovery image $I^R = \{I_i^R\}_{i=0}^{N-1}$. Followings are the list of our decoding and recovery algorithms:

Input: To-be-authenticated codes $\{\hat{a}'_i, \hat{b}'_i, \hat{B}'_i\}_{i=0}^{N-1}$.

Output: Recovered image I^R.

Step 1: Scan $\{\hat{a}'_i, \hat{b}'_i, \hat{B}'_i\}_{i=0}^{N-1}$ and compute the authentication code of $\{\hat{a}'_i, \hat{b}'_i, \hat{B}'_i\}$ using the equation $\hat{ac}_i = \text{fold}(\text{md5}(i, \hat{B}'_i), \alpha)$.

Step 2: Extract $\hat{eac}_i$ from $\hat{a}'_i$ and $\hat{b}'_i$ using the APPM, and compare $\hat{eac}_i$ with $\hat{ac}_i$. If they are the same, $\{\hat{a}'_i, \hat{b}'_i, \hat{B}'_i\}$ is judged to be untampered with. Otherwise, it is judged to have been tampered with.

Step 3: Repeat Steps 1-2 until all blocks are processed, and re-scan the image blocks. If an untampered block surrounds by two or more tampered blocks, then the block is re-judged as a tampered one. After performing this step, we obtain un-tampered AMBTC codes $\{\hat{a}'_u, \hat{b}'_u, \hat{B}'_u\}_{i=0}^{N_u-1}$.

Step 4: Use the inverse-scramble function $p = s^{-1}(u)$ to obtain the cluster information of $\hat{I}'_p$ stored in $\hat{B}'_u$. Then, extract $\{\hat{B}'_{u,j}\}_{j=0}^{6}$ and $\{\hat{B}'_{u,j}\}_{j=7}^{13}$ from $\hat{B}'_u$ and obtain $(\varphi_p^0)_2$ and $(\varphi_p^1)_2$ using Matrix encoding. Finally, convert $(\varphi_p^0) \,||\, (\varphi_p^1)$ into its decimal representation and obtain the cluster information φ_p of block $\hat{I}'_p$.

Step 5: Repeat Step 4 until all $\{\hat{a}'_u, \hat{b}'_u, \hat{B}'_u\}_{i=0}^{N_u-1}$ are processed, and we obtain the cluster information $\{\varphi_{s^{-1}(u)}\}_{u=0}^{N_u-1}$ of $\{\hat{I}'_{s^{-1}(u)}\}_{u=0}^{N_u-1}$. Then, average the blocks with the same cluster information and obtain $\{\bar{I}_r\}_{r=0}^{63}$.

Step 6: To recover a tampered block $\hat{I}_t$, we extract its cluster information from $\{\hat{I}'_{s^{-1}(u)}\}_{u=0}^{N_u-1}$, and find a block $\bar{I}_{\varphi_t}$ from $\{\bar{I}_r\}_{r=0}^{63}$ to recover $\hat{I}_t$.

Step 7: Repeat Step 6 until all tampered blocks are visited and processed.

Step 8: For those blocks that cannot recovered in Steps 6-7, we perform the secondary recovery using the Side match technique, as described in Section III.D. Finally, the recovered image I^R can be obtained.

Here we use a simple example to illustrate the secondary recovery process of the proposed method. Let $\hat{I}_\tau$ be the block that cannot be repaired after performing the primary recovery, as shown in Fig. 6 (a). In the figure, the numbers in the upper left corner of each square denote the pixel positions. Since the neighboring blocks of $\hat{I}_\tau$ are known, we can use the equations shown above to estimate the pixel values of $\hat{I}_\tau$ at positions 0, 1, 2, 3, 4, 7, 8, 11, 12, 13, 14 and 15. Therefore, we have $\hat{I}_{\tau,0} = 57, \hat{I}_{\tau,1} = 59, \hat{I}_{\tau,2} = 63, \hat{I}_{\tau,3} = 63, \hat{I}_{\tau,4} = 61, \hat{I}_{\tau,7} = 66, \hat{I}_{\tau,8} = 63, \hat{I}_{\tau,11} = 59, \hat{I}_{\tau,12} = 63, \hat{I}_{\tau,13} = 55, \hat{I}_{\tau,14} = 57$ and $\hat{I}_{\tau,15} = 59$, as shown in Fig. 6 (b). Then, we compare the estimated values of $\hat{I}_\tau$ with the corresponding pixels of $\{\bar{I}_r\}_{r=0}^{63}$, and select a block from $\{\bar{I}_r\}_{r=0}^{63}$ with the smallest distance to recover the block $\hat{I}_\tau$, as shown in Fig. 6 (c).

IV. EXPERIMENTAL RESULTS

In this section, we will conduct several experiments to demonstrate the effectiveness of the proposed method. In our experiments, ten grayscale images of size 512×512 are used as test images. Among them, Lena, Baboon, House, Stream, Tiffany, Splash, Airplane, and Boat are obtained from the USC-SIPI image database [35]. The Building and Cloud are natural images obtained from a modern digital camera. These images are shown in Fig. 7. In our experiments, a block size of 4×4 is used. The size of the image has a small impact on the detectability of our method, but the detectability will be significantly influenced by the block size. For each block embedded with equal-length authentication codes, the larger blocks yield rougher results, while the smaller ones yield more accurate results.

We use the peak signal-to-noise ratio PSNR to measure the quality of the marked images. Higher PSNR means that the marked images are closer to the original ones. The metric PSNR used in this paper is defined as:

$$\text{PSNR} = 10\log_{10}\frac{255^2}{\frac{1}{n\times n\times N}\sum_{i=0}^{n\times n\times N-1}(x_i - x'_i)^2}, \quad (12)$$

where x_i and x'_i are the pixel values of the AMBTC compressed images and the marked images, respectively.

A. IMAGE QUALITY EVALUATION OF THE PROPOSED METHOD

In this section, we examine the image quality at different thresholds with and without using the BFT, as described in Section III.B. The comparison results are shown in Table 1, where 'w/BFT' and 'w/o BFT' denote with and without applying the BFT, respectively. In this experiment, the quantization values of both smooth and complex blocks are

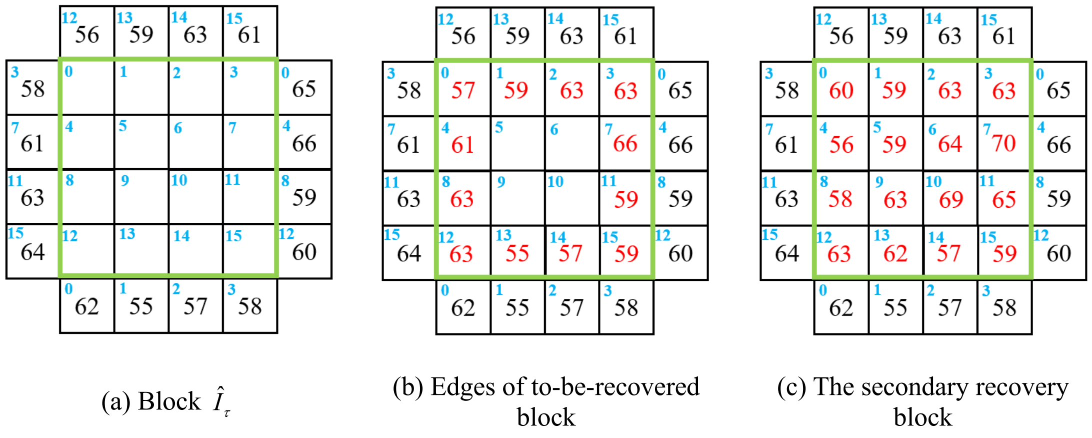

(a) Block $\hat{I}_\tau$
(b) Edges of to-be-recovered block
(c) The secondary recovery block

FIGURE 6. **The secondary recovery procedures using the side match technique.**

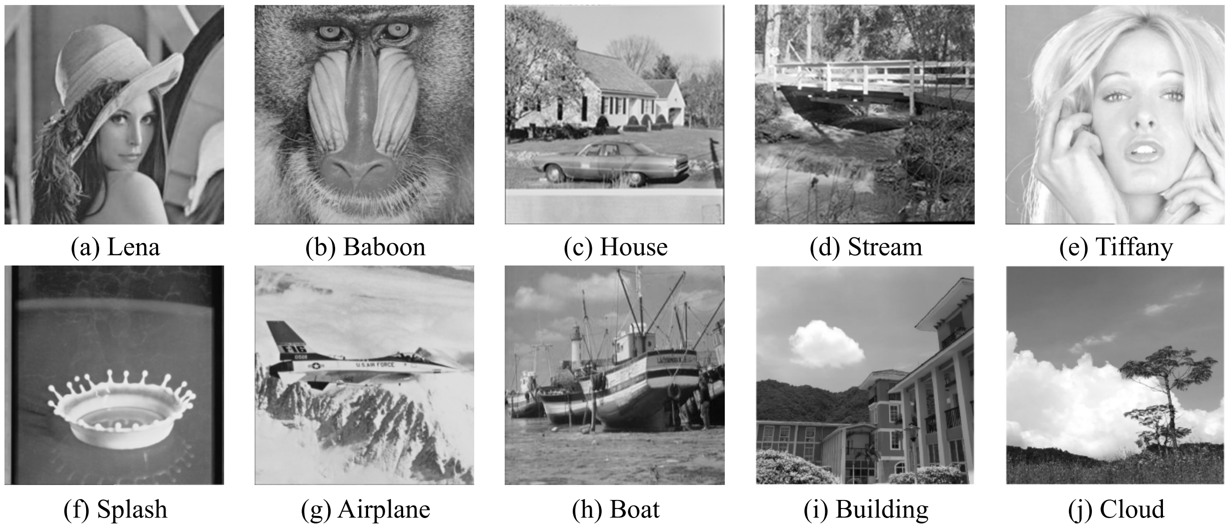


FIGURE 7. **Ten test images.**

embedded with 4-bit authentication codes, while the bitmaps of smooth blocks are embedded with 6-bit recovery codes.

As can be observed from Table 1, the image quality decreases with increasing T_0. This is because a larger T_0 represents a larger number bitmaps of image blocks being used to embed the recovery codes, and thus it causes a larger image distortion. For example, when $T_0 = 5$, the Lena image has an embedding capacity of 93,406 bits, while the embedding capacity reaches 152,260 bits when $T_0 = 30$. In addition, we can also observe that the use of BFT has a significant improvement on the image quality when the threshold value is small. However, with the increase of T_0, the improvement of image quality by BFT is not significant. We take the House image as an example. When $T_0 = 5$, the use of BFT improves $46.65 - 46.07 = 0.58$ dB than without using BFT, while when $T_0 = 30$, only $36.55 - 36.50 = 0.05$ dB is improved. This is because the larger the threshold, the greater the difference between the upper and lower quantization values. In real applications, the selection of the threshold mainly depends on the actual needs. If the quality of the marked image is more desirable than the recovery capability, we can choose a smaller threshold; otherwise, a larger threshold is selected.

However, not every smooth image block can reduce the embedding error using the BFT. The reason is that flipping bits using the BFT operation also causes some errors. Fig. 8 shows the distribution of reduced or unaltered errors when applying the BFT. In this figure, black and white blocks are spread over the image. The black blocks indicate that the use of the BFT reduces embedding distortions, while the white ones indicate that the technique is not effective. When $T_0 = 5$, an equal number of white and black blocks can be found; while when $T_0 = 10$, there are significantly fewer black blocks than white ones. The experimental results are consistent with the analysis given above.

B. DETECTABILITY AND RECOVERABILITY OF THE PROPOSED METHOD

In order to evaluate the detection and recovery performance of the proposed method, this experiment performs on ten test images with different tampering. The tampering rates

TABLE 1. Image quality comparisons with different thresholds.

Image		$T_0 = 5$	$T_0 = 10$	$T_0 = 15$	$T_0 = 20$	$T_0 = 25$	$T_0 = 30$
Lena	w/BFT	45.89	43.19	41.17	39.67	38.49	37.46
	w/o BFT	45.50	43.00	41.04	39.58	38.42	37.40
Baboon	w/BFT	46.77	44.99	41.92	39.57	37.64	35.91
	w/o BFT	46.76	44.98	41.92	39.56	37.64	35.91
House	w/BFT	46.65	44.81	42.43	40.07	38.13	36.55
	w/o BFT	46.07	44.44	42.21	39.95	38.05	36.50
Stream	w/BFT	46.72	45.20	42.11	39.00	36.50	34.58
	w/o BFT	46.65	45.15	42.08	38.99	36.50	34.57
Tiffany	w/BFT	45.84	43.27	41.22	39.68	38.59	37.71
	w/o BFT	45.33	43.01	41.05	39.57	38.50	37.64
Splash	w/BFT	45.67	42.90	41.58	40.73	40.22	39.80
	w/o BFT	45.11	42.61	41.36	40.55	40.06	39.66
Airplane	w/BFT	46.25	44.36	42.55	40.89	39.64	38.61
	w/o BFT	45.53	43.91	42.25	40.68	39.48	38.49
Boat	w/BFT	46.43	43.03	40.33	38.46	37.12	36.11
	w/o BFT	46.30	42.97	40.29	38.44	37.11	36.10
Building	w/BFT	47.70	46.31	44.19	41.77	40.11	38.87
	w/o BFT	46.31	45.27	43.52	41.38	39.84	38.66
Cloud	w/BFT	47.92	47.04	46.24	45.28	44.03	42.12
	w/o BFT	46.08	45.49	44.92	44.19	43.18	41.56
Average	w/BFT	46.58	44.51	42.37	40.51	39.05	37.77
	w/o BFT	46.01	43.99	41.81	39.92	38.47	37.27

of the images are 10.01%, 7.97%, 6.62%, 10.42%, 9.30%, 9.22%, 8.83%, 5.21%, 8.12% and 10.56%, respectively, and the tampered images can be seen in Figs. 9 (a)-(j). In the experiments of this section, we set $T_0 = 20$ to achieve a balance between the detectability and recoverability. The lengths of the authentication and recovery codes are set to 4 and 6 bits, respectively.

Figure 10 shows the secondary detection results of the ten tampered images. At the first detection, the probability of a hash collision is $1/2^4 = 0.0625$ since 4 bits of authentication code are embedded. As a result, about 6.25% of tampering per image goes undetected. In secondary detection, if a tampered block is within the tampered region, it has a higher probability of being redetected as a tampered one than if it is at the edge of the tampered region. Therefore, the more the tampered regions is clustered, the better the final detection result will be. For example, images with less clustered tampered regions, such as Stream, have a detection rate of only 94.50 %. However, for the Lena, Tiffany and Splash images, the tampered regions are clustered, and they all achieve a detection result of more than 99%. As can be seen from Fig. 10, the tampered regions can be detected with a good result using our method regardless of whether they are clustered or not.

To further present the detection performance of the proposed method, Table 2 shows the true positive rate (TPR), false negative rate (FNR), true negative rate (TNR) and false positive rate (FPR) of Tiffany images at $T_0 = 20$. TPR is the rate of recognizing tampered blocks as tampered, while FNR is the rate of recognizing tampered blocks as untampered. TNR is the rate of recognizing untampered blocks as untampered, and FPR is the rate of recognizing untampered blocks as tampered. In the experiment, the length of authentication code is set to 4. Theoretically, the hash collision rate is $1/2^4 = 0.0625$. From the table, we can see that the TPR of the first detection is 93.23%, which is well matched with the theoretical value. Since FNR $= 100\% - $ TPR, the FNR is also consistent with the theoretical value. In the primary detection, the value of TNR is 100% and FPR is 0%. This is

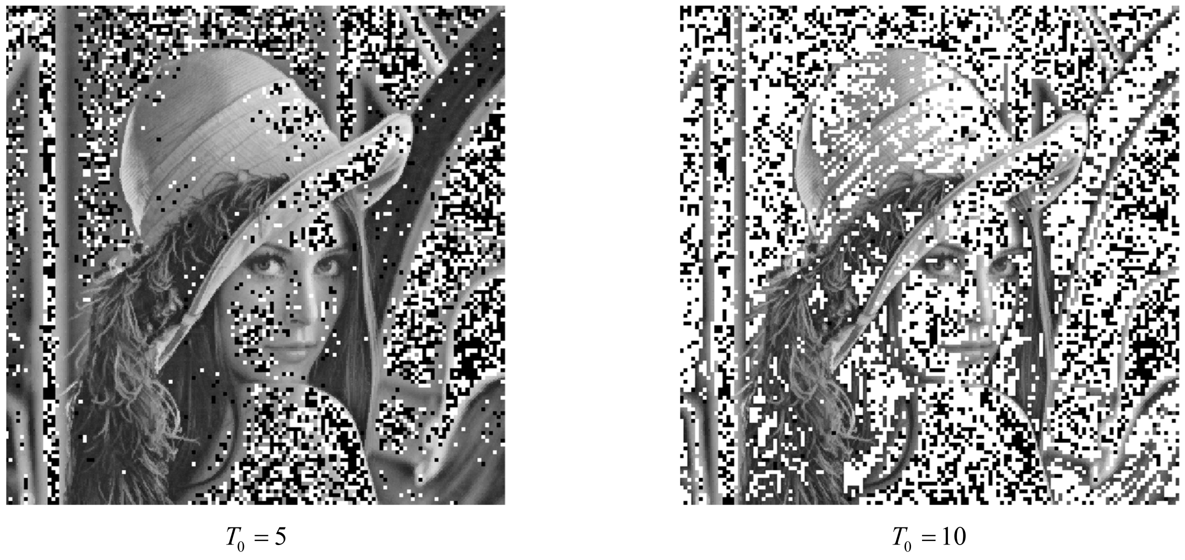


FIGURE 8. Effectiveness comparison of the BFT.

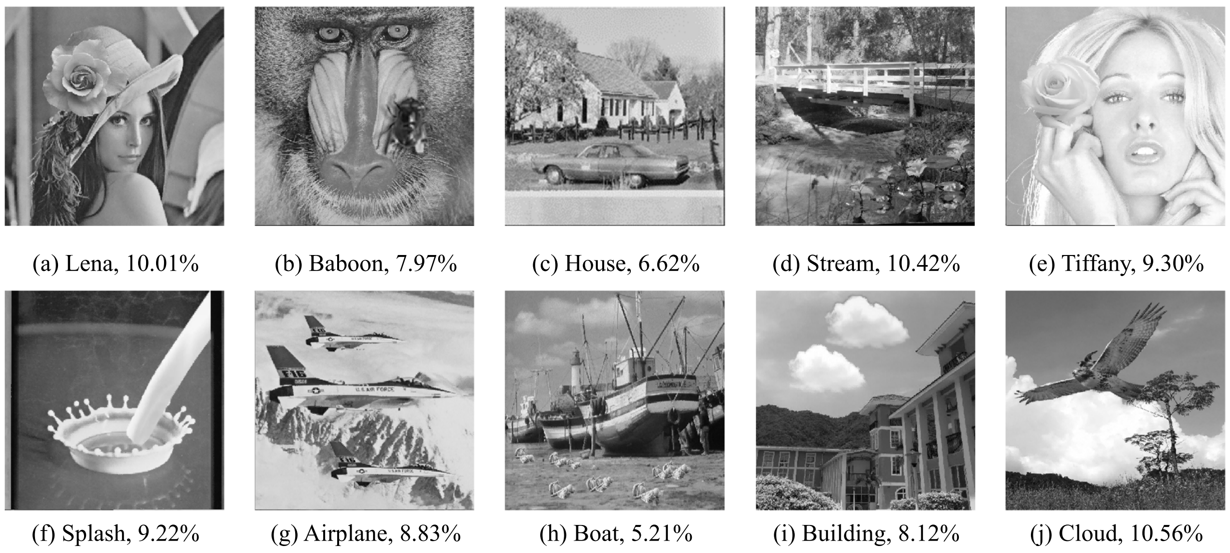


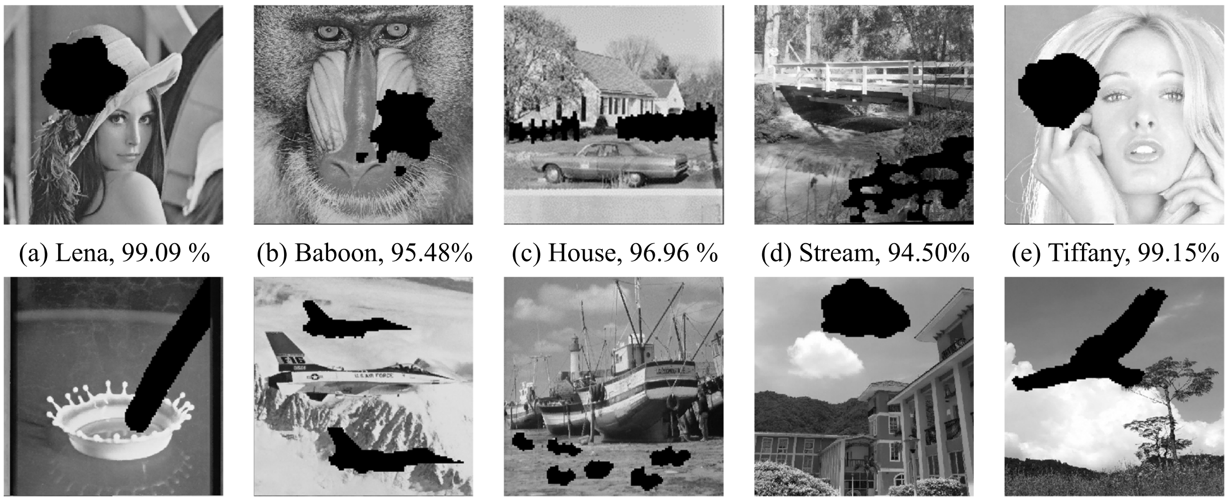
FIGURE 9. The ten tampered images.

because the primary detection does not misidentify untampered blocks as tampered. However, after the secondary detection, some untampered blocks are misclassified as tampered, and the TNR is reduced to 99.97%. Nevertheless, the TPR is increased to 99.21%, and the FNR is reduced to 0.79%, revealing the effectiveness of the secondary detection. Although the experiment is based on the Tiffany image, other images show very similar results. Therefore, the test results for the other images are not shown in Table 2.

Figure 11 shows the secondary recovery results for the ten tampered images. Note that the PSNRs shown in the figure only measures the quality of recovered regions (not the whole image), making it easier to compare the recovery effect. As can be seen from the figure, the quality of the recovered image is related to the smoothness of the original image and the size of tampered regions. For example, Lena and Stream have roughly the same tampering rate. However, since Lena is smoother than Stream, the quality of recovery is higher for Lena than Stream. This is because with the same threshold, a smooth image will have more blocks for

TABLE 2. The primary and secondary detection results.

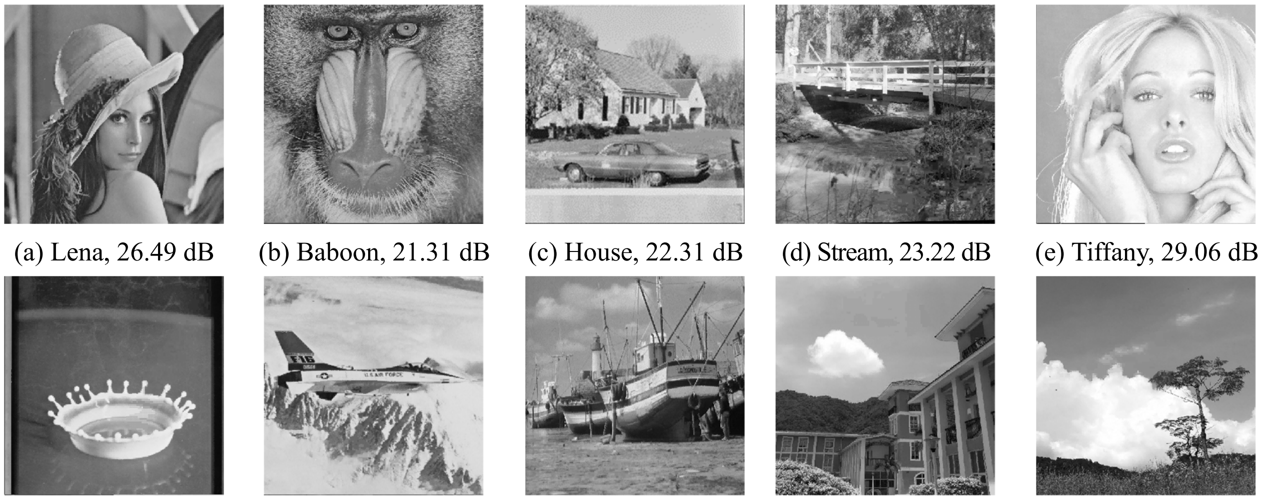
	TPR	FNR	TNR	FPR
Primary detection	93.23%	6.77%	100%	0%
Secondary detection	99.21%	0.79%	99.97%	0.03%



(a) Lena, 99.09 % (b) Baboon, 95.48% (c) House, 96.96 % (d) Stream, 94.50% (e) Tiffany, 99.15%

(f) Splash, 99.01 % (g) Airplane, 97.92% (h) Boat, 96.72% (i) Building, 99.02% (j) Cloud, 99.19%

FIGURE 10. The secondary detection results of ten test images.



(a) Lena, 26.49 dB (b) Baboon, 21.31 dB (c) House, 22.31 dB (d) Stream, 23.22 dB (e) Tiffany, 29.06 dB

(f) Splash, 31.00 dB (g) Airplane, 26.72 dB (h) Boat, 28.50 dB (i) Building, 41.77dB (j) Cloud, 28.46 dB

FIGURE 11. The recovery of tampered regions.

embedding the recovery code and thus a better recoverability can be achieved. In fact, the primary recovery rates of the ten images are 77.26%, 33.88%, 53.69%, 34.31%, 77.61%, 83.45%, 72.13%, 67.41%, 65.41%, and 65.55%, respectively. Although the recovery rates of some images in the first phase are somewhat unsatisfactory due to the limited space for embedding the recovery codes, the recovery results in the second phase are very impressive, as shown in Fig. 11. The experimental results reveal that our method can roughly recover the tampered area with a tampering rate of less than 20%.

C. PERFORMANCE COMPARISONS WITH OTHER WORKS

In this section, we compare the proposed method with other works in terms of image quality, detectability, and recoverability. For a fair comparison, each method will embed a 4-bit authentication code with a 6-bit recovery code. Table 3 shows comparisons of the marked image qualities of [24], [25] and the proposed method under different thresholds. The last row of the table represents the quality average of the ten test images.

From Table 3, it can be observed that the quality of the marked images obtained by the proposed method is the highest when the same number of bits are embedded. This is because the APPM method is more effective in embedding data than other related ones. The methods of [24] and [25] embed the recovery code by using LSB-like substitution, while the proposed method uses Matrix encoding to embed the recovery codes. Since the use of Matrix encoding reduces the modification of bitmaps, it effectively reduces the damage

TABLE 3. Image quality comparisons with other works under different thresholds.

Image	$T_0 = 10$			$T_0 = 20$			$T_0 = 30$		
	Proposed	[25]	[24]	Proposed	[25]	[24]	Proposed	[25]	[24]
Lena	43.19	41.46	35.45	39.67	38.29	32.29	37.46	36.07	30.44
Baboon	44.99	43.17	34.27	39.57	38.05	32.31	35.91	34.32	29.61
House	44.81	42.95	35.02	40.07	38.41	32.53	36.55	34.94	30.10
Stream	45.20	43.46	34.56	39.00	37.27	32.07	34.58	32.94	28.80
Tiffany	43.27	41.31	35.51	39.68	38.31	32.31	37.71	36.29	30.59
Splash	42.90	41.18	35.77	40.73	39.34	32.59	39.80	38.46	31.53
Airplane	44.36	42.47	35.44	40.89	39.32	32.78	38.61	37.11	31.25
Boat	43.03	41.16	35.00	38.46	37.02	31.77	36.11	34.79	29.53
Building	46.31	43.72	35.10	41.77	39.83	33.23	38.87	37.16	31.56
Cloud	47.04	43.33	35.15	45.28	42.22	34.01	42.12	39.55	32.98
Average	44.51	42.42	35.13	40.51	38.81	32.59	37.77	36.16	30.64

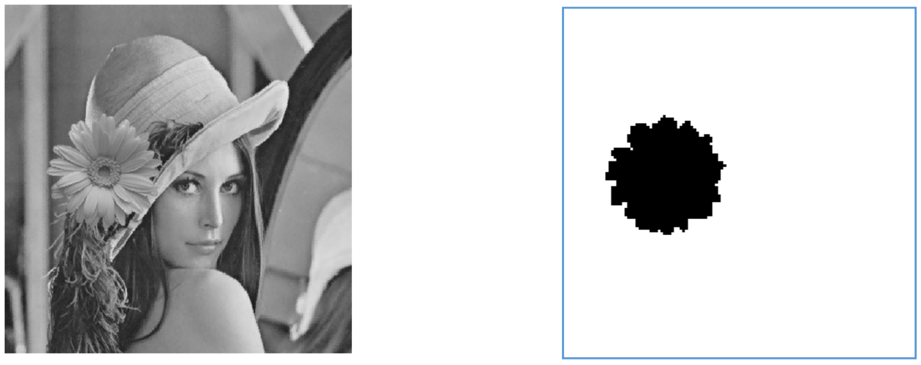
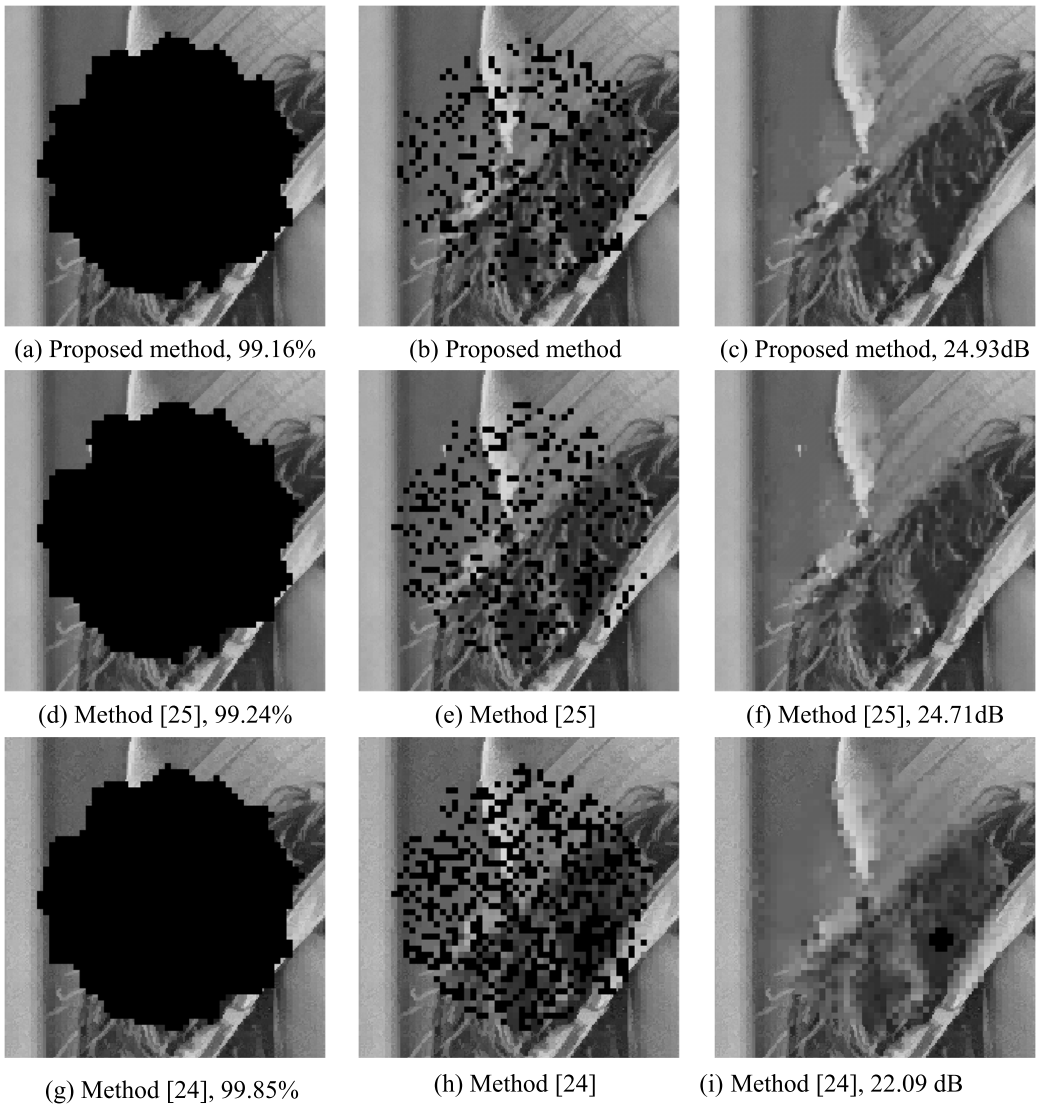


FIGURE 12. The tampered Lena image.

to bitmaps. When $T_0 = 30$, the average marked image quality obtained by the proposed method is $37.77 - 36.16 = 1.61$ dB higher than that of [25] and $37.77 - 30.64 = 7.13$ dB higher than that of [24].

To compare the detection and recovery performance with different methods, we add a daisy to Lena's hat, as shown in Fig. 12 (a), whereas the region of tampering is shown in Fig. 12 (b).

Figure 13 shows the comparisons of detection and recovery results of [24], [25], and the proposed method. From the secondary detection results (See Figs. 13 (a), (d) and (g)), it can be seen that the proposed method and other works can achieve a good detection performance of more than 99%. However, compared to [24], the proposed method and [25] have better primary recovery results, as seen in Figs. 13 (b), (e) and (h). This is because [24] embeds the average value of each block directly into the bitmap of smooth block. If the bitmap used to store the average value of a block is damaged, the block will not be recovered in the primary recovery process. Reference [25] and the proposed method both use the cluster information to embed the recovery code. If the bitmap used to embed the recovery code of a block is tampered with,



(a) Proposed method, 99.16% (b) Proposed method (c) Proposed method, 24.93dB

(d) Method [25], 99.24% (e) Method [25] (f) Method [25], 24.71dB

(g) Method [24], 99.85% (h) Method [24] (i) Method [24], 22.09 dB

FIGURE 13. **Comparisons of the detection and recovery results.**

TABLE 4. **Comparisons with other works.**

Methods	[24]	[25]	Proposed
AC generation	Random number generator	MD5	MD5
Special tampering	Undetectable	Detectable	Detectable
RC generation	Mean value of blocks	Cluster information	Cluster information
RC embedment	LSB substitution	LSB substitution	Matrix encoding
AC embedment	LSB substitution	LSB-like	APPM
Secondary recovery	Adjacent blocks	Adjacent blocks	Side match

the block can be recovered with blocks of the same group. Figs. 13 (i), (f) and (c) are the secondary recovery results of [24], [25] and the proposed method, respectively. From Fig. 13 (i), it is observed that the block recovered by [24] has an apparent mosaic-like pattern, which is due to the fact that all pixels in a tampered block are recovered by an identical average value. Compared with the secondary recovery result of [25] (Fig. 13 (f)), the proposed method uses the Side match technique to more effectively utilize the pixels at the block borders, and obtains better recovery results. Note that the secondary recovery qualities obtained by [24], [25] and the proposed method are 22.09, 24.71 and 24.93 dB, respectively, and the proposed method also has the best visual quality.

Table 4 shows the comparisons of the proposed method with other works in various aspects, where AC and RC denote the abbreviations of authentication code and recovery code, respectively. Reference [24] uses the parity of bits in bitmap to generate the authentication code, causing failure in detecting special tampering, such as two bits in bitmaps are flipped simultaneously. However, since [25] and our method hash the bitmap to generate authentication code, the aforementioned tampering can be successfully detected. In addition, recovered images of [24] will exhibit mosaic-like patterns as their method uses a mean value to repair a tampered block. Both [25] and our method do not have this phenomenon because the tampered block is recovered by a similar one. Besides, both [24] and [25] use LSB substitution to embed the recovery codes, while our method utilizes the Matrix encoding to effectively reduce the embedding error. Moreover, [24], [25] and our method utilize different approaches to embed the authentication codes. As can be seen from the experiments, the APPM used in our method achieves the highest image quality. Finally, the Side match technique is utilized to recover tampered blocks in the proposed method, which has the best recovery performance than [24] and [25]. Therefore, our method is better than the other works in terms of detectability and recoverability.

V. CONCLUSION

In this paper, we propose an image authentication method with recovery capability based on the AMBTC compressed images. The proposed method employs Matrix encoding to embed the recovery code into the bitmap of a smooth block. Therefore, only two out fourteen bits are required to be modified at most for embedding a 6-bit recovery code, which reduces the image distortion in a great extent. Besides, our method generates the authentication code by flipping bits in the bitmap, and selecting the code that causes the least embedding error. Finally, the authentication code is embedded in the quantization values using APPM. The experimental results show that the quality of the marked and the recovered images obtained by our method are better than prior works.

However, there are some limitations to our approach. Since each block of AMBTC is represented by only two quantization values and one bitmap, the quality of AMBTC decompressed image is not high. Therefore, our approach is not suitable to be applied to mark cards, certificates, cheques, and handwritten images. Nevertheless, this paper still uses common test images obtained from the USC-SIPI image database and two natural images obtained from a digital camera to demonstrate the effectiveness of our method. Future developments can be implemented by improving the quality of compressed images for use in a wider range of applications. Besides, when the tampering rate is higher than 20%, some blocks located in the tampered area may not be successfully recovered in our method. Therefore, in the future work, we will improve the recoverability of our method so that the tampering can be recovered with a higher tampering rate.

REFERENCES

[1] T. Qiao, A. Zhu, and F. Retraint, "Exposing image resampling forgery by using linear parametric model," *Multimedia Tools Appl.*, vol. 77, no. 2, pp. 1501–1523, 2018.

[2] T. Mahmood, Z. Mehmood, M. Shah, and T. Saba, "A robust technique for copy-move forgery detection and localization in digital images via stationary wavelet and discrete cosine transform," *J. Vis. Commun. Image Represent.*, vol. 53, pp. 202–214, May 2018.

[3] Z. He, W. Sun, W. Lu, and H. Lu, "Digital image splicing detection based on approximate run length," *Pattern Recognit. Lett.*, vol. 32, no. 12, pp. 1591–1597, Sep. 2011.

[4] R. O. Preda and D. N. Vizireanu, "Robust wavelet-based video watermarking scheme for copyright protection using the human visual system," *J. Electron. Imag.*, vol. 20, no. 1, pp. 146–152, 2011.

[5] T.-M. Thanh and G.-N. Dan, "Robust watermarking method using QIM with VSS for image copyright protection," *J. Res. Develop. Inf. Commun. Technol.*, vol. 2021, no. 1, pp. 28–40, May 2021.

[6] R. O. Preda, "Semi-fragile watermarking for image authentication with sensitive tamper localization in the wavelet domain," *Measurement*, vol. 46, no. 1, pp. 367–373, Jan. 2013.

[7] Y. Peng, X. Niu, L. Fu, and Z. Yin, "Image authentication scheme based on reversible fragile watermarking with two images," *J. Inf. Secur. Appl.*, vol. 40, pp. 236–246, Jun. 2018.

[8] Z. Yin, X. Niu, Z. Zhou, J. Tang, and B. Luo, "Improved reversible image authentication scheme," *Cognit. Comput.*, vol. 8, no. 5, pp. 890–899, Oct. 2016.

[9] X. Zhou, W. Hong, S. Weng, T.-S. Chen, and J. Chen, "Reversible and recoverable authentication method for demosaiced images using adaptive coding technique," *J. Inf. Secur. Appl.*, vol. 55, Dec. 2020, Art. no. 102629.

[10] D. Singh and S. K. Singh, "Effective self-embedding watermarking scheme for image tampered detection and localization with recovery capability," *J. Vis. Commun. Image Represent.*, vol. 38, pp. 775–789, Jul. 2016.

[11] C. Qin, P. Ji, X. Zhang, J. Dong, and J. Wang, "Fragile image watermarking with pixel-wise recovery based on overlapping embedding strategy," *Signal Process.*, vol. 138, pp. 280–293, 2017.

[12] C. Qin, P. Ji, C.-C. Chang, J. Dong, and X. Sun, "Non-uniform watermark sharing based on optimal iterative BTC for image tampering recovery," *IEEE MultiMedia*, vol. 25, no. 3, pp. 36–48, Jul./Sep. 2018.

[13] E. Kee, M. K. Johnson, and H. Farid, "Digital image authentication from JPEG headers," *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 3, pp. 1066–1075, Sep. 2011.

[14] T. H. Thai, R. Cogranne, F. Retraint, and T.-N.-C. Doan, "JPEG quantization step estimation and its applications to digital image forensics," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 1, pp. 123–133, Jan. 2017.

[15] A. Tiwari and M. Sharma, "Novel watermarking scheme for image authentication using vector quantization approach," *Radioelectron. Commun. Syst.*, vol. 60, no. 4, pp. 161–172, Apr. 2017.

[16] S. Nguyen, C.-C. Chang, and T.-F. Chung, "A tamper-detection scheme for BTC-compressed images with high-quality images," *KSII Trans. Internet Inf. Syst.*, vol. 8, no. 6, pp. 2005–2021, 2014.

[17] C. C. Lin, Y. Huang, and W. L. Tai, "A high-quality image authentication scheme for AMBTC-compressed images," *KSII Trans. Internet Inf. Syst.*, vol. 8, no. 12, pp. 4588–4603, 2014.

[18] W. Li, C.-C. Lin, and J.-S. Pan, "Novel image authentication scheme with fine image quality for BTC-based compressed images," *Multimedia Tools Appl.*, vol. 75, no. 8, pp. 4771–4793, 2016.

[19] C.-C. Lin, Y. Huang, and W.-L. Tai, "A novel hybrid image authentication scheme based on absolute moment block truncation coding," *Multimedia Tools Appl.*, vol. 76, no. 1, pp. 463–488, 2017.

[20] T.-H. Chen and T.-C. Chang, "On the security of a BTC-based-compression image authentication scheme," *Multimedia Tools Appl.*, vol. 77, no. 10, pp. 12979–12989, 2018.

[21] W. Hong, X. Zhou, D.-C. Lou, X. Huang, and C. Peng, "Detectability improved tamper detection scheme for absolute moment block truncation coding compressed images," *Symmetry*, vol. 10, no. 8, p. 318, Aug. 2018.

[22] C.-C. Chen, C.-C. Chang, C.-C. Lin, and G.-D. Su, "TSIA: A novel image authentication scheme for AMBTC-based compressed images using turtle shell based reference matrix," *IEEE Access*, vol. 7, pp. 149515–149526, 2019.

[23] G. Su, C. C. Chang, and C. C. Lin, "High-precision authentication scheme based on matrix encoding for AMBTC-compressed images," *Symmetry*, vol. 11, no. 8, p. 996, Aug. 2019.

[24] Y.-C. Hu, K.-K.-R. Choo, and W.-L. Chen, "Tamper detection and image recovery for BTC-compressed images," *Multimedia Tools Appl.*, vol. 76, no. 14, pp. 15435–15463, Jul. 2017.

[25] W. Hong, X. Zhou, and D.-C. Lou, "A recoverable AMBTC authentication scheme using similarity embedding strategy," *PLoS ONE*, vol. 14, no. 2, Feb. 2019, Art. no. e0212802.

[26] C.-K. Chan and L. M. Cheng, "Hiding data in images by simple LSB substitution," *Pattern Recognit.*, vol. 37, no. 3, pp. 469–474, Mar. 2004.

[27] J. Mielikainen, "LSB matching revisited," *IEEE Signal Process. Lett.*, vol. 13, no. 5, pp. 285–287, May 2006.

[28] W. Hong, T.-S. Chen, and J. Chen, "Reversible data hiding using Delaunay triangulation and selective embedment," *Inf. Sci.*, vol. 308, pp. 140–154, Jul. 2015.

[29] W. Hong and T.-S. Chen, "A novel data embedding method using adaptive pixel pair matching," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 1, pp. 176–184, Feb. 2012.

[30] H. Chen, J. Ni, W. Hong, and T.-S. Chen, "High-fidelity reversible data hiding using directionally enclosed prediction," *IEEE Signal Process. Lett.*, vol. 24, no. 5, pp. 574–578, May 2017.

[31] S. Liu, Z. Fu, and B. Yu, "Rich QR codes with three-layer information using Hamming code," *IEEE Access*, vol. 7, pp. 78640–78651, 2019.

[32] M. Lema and O. Mitchell, "Absolute moment block truncation coding and its application to color images," *IEEE Trans. Commun.*, vol. COM-32, no. 10, pp. 1148–1157, Oct. 1984.

[33] R. M. Gray, "Vector quantization," *IEEE ASSP Mag.*, vol. AM-1, no. 2, pp. 4–29, Apr. 1984.

[34] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. Boca Raton, FL, USA: CRC Press, 1996.

[35] *The USC-SIPI Image Database*. [Online]. Available: http://sipi.usc.edu/database/

WIEN HONG received the M.S. and Ph.D. degrees from The State University of New York at Buffalo, USA, in 1994 and 1997, respectively. He is currently a Professor with the Department of Computer Science and Information Technology, National Taichung Institute of Science and Technology. His research interests include steganography, watermarking, and image compression.

JIE WU has been a Senior Researcher and an Engineer with the School of Electrical and Computer Engineering, Nanfang College of Guangzhou, since 2019. She has incorporated several major projects about digital signal processing (DSP) and applications of embedded systems. Her research interests include development of single-chip microcomputer, image compression, data hiding, and image authentication.

DER-CHYUAN LOU received the B.S. degree from Chung Cheng Institute of Technology (CCIT), National Defense University, Taiwan, in 1987, the M.S. degree from the National Sun Yat-sen University, Taiwan, in 1991, both in electrical engineering, and the Ph.D. degree from the Department of Computer Science and Information Engineering, National Chung Cheng University, Taiwan, in 1997. He was a Lecturer, an Associate Professor, and a Professor with the Department of Electrical Engineering, CCIT, from 1987 to 2009. He had served as the Director of the Computer Center, CIT, from 2004 to 2006. Currently, he is a Professor with the Department of Computer Science and Information Engineering, Chang Gung University. His research interests include multimedia security, steganography, cryptography, and computer arithmetic.

XIAOYU ZHOU is currently pursuing the degree with the School of Information Engineering, Jimei University, Xiamen, China. She has published several SCI papers in the field of image and signal processing. Her research interests include image analysis, data hiding, and image authentication.

JEANNE CHEN was on the Faculty of the Department of Computer Science Information Management, Hungkuang University, Taiwan, from 1992 to 2006, and had also held a concurrent position as the Head of the Computer Center for the first four years. Currently, she is a Professor with the Department of Computer Science and Information Engineering, National Taichung University of Science and Technology, Taiwan. Her research interests include steganography, image compression, image cryptosystems, and bioinformatics.