applied sciences

MDPI

Article

An Authentication Method for AMBTC Compressed Images Using Dual Embedding Strategies

Xiaoyu Zhou [1], Jeanne Chen [2,*], Guangsong Yang [1], Zheng-Feng Lin [2] and Wien Hong [2,*]

[1] School of Ocean Information Engineering, Jimei University, Xiamen 361021, China
[2] Department Computer Science and Information Engineering, National Taichung University of Science and Technology, Taichung City 404, Taiwan
* Correspondence: jeanne@nutc.edu.tw (J.C.); wienhong@nutc.edu.tw (W.H.)

Abstract: In this paper, we proposed an efficient authentication method with dual embedment strategies for absolute moment block truncation coding (AMBTC) compressed images. Prior authentication works did not take the smoothness of blocks into account and only used single embedding strategies for embedment, thereby limiting image quality. In the proposed method, blocks were classified as either smooth or complex ones, and dual embedding strategies used for embedment. Respectively, bitmaps and quantized values were embedded with authentication codes, while recognizing that embedment in bitmaps of complex blocks resulted in higher distortion than in smooth blocks. Therefore, authentication codes were embedded into bitmaps of smooth blocks and quantized values of complex blocks. In addition, our method exploited to-be-protected contents to generate authentication codes, thereby providing satisfactory detection results. Experimental results showed that some special tampering, undetected by prior works, were detected by the proposed method and the averaged image quality was significantly improved by at least 1.39 dB.

Keywords: AMBTC; authentication; matrix encoding; APPM

Citation: Zhou, X.; Chen, J.; Yang, G.; Lin, Z.-F.; Hong, W. An Authentication Method for AMBTC Compressed Images Using Dual Embedding Strategies. *Appl. Sci.* **2023**, *13*, 1402. https://doi.org/10.3390/app13031402

Academic Editor: Andrea Prati

Received: 24 October 2022
Revised: 6 January 2023
Accepted: 11 January 2023
Published: 20 January 2023

1. Introduction

With the progress and development of Internet, digital images are increasingly easy to be transmitted over the network. However, these images are likely to be tampered with either with malicious intent or otherwise, resulting in unauthentic images being received. Therefore, authentication of images has become an important issue. Fragile watermarking [1–5] is a common technique used in image authentication by embedding watermark data into images. If pixels were tampered with, the embedded data could be retrieved to authenticate the image.

The fragile watermarking technique can be applied to images in spatial [6–10] or compressed [11–13] domains. In the spatial domain, data are embedded directly by modifying pixel values. Since images of spatial domain have many redundancies for embedding, higher payloads are expected. However, in recent years, most images are stored and transmitted in compressed formats. This is because compressed images have lower storage space and transmission bandwidth requirements. At present, vector quantization (VQ) [14–16], joint photographic expert group (JPEG) [17–19], and absolute moment block truncation coding (AMBTC) [20,21] are some commonly used compression techniques. Amongst these techniques, AMBTC requires the least computational cost. Therefore, several authentication techniques based on AMBTC images [22–27] have been proposed.

AMBTC was proposed by Lema and Mitchell [28], where blocks were compressed into quantized values and bitmaps. In 2013, ref. [22] proposed an AMBTC image authentication method with joint image coding. This method used pseudo random sequence to generate authentication codes which were embedded into the bitmaps. The embedded bitmaps together with the quantized values were, then, losslessly compressed to obtain the final

bitstream. The advantage of their method was that it required less storage and achieved good detection accuracy. To improve image quality, ref. [23] in 2016 proposed a novel image authentication method based on a reference matrix. The matrix was employed to embed authentication codes into quantized values. The length of authentication codes was determined by the size of the matrix. Ref. [23] achieved a higher image quality compare to [22]. However, this method could not detect tampering in the bitmaps which were independently generated from the authentication codes. To improve security, ref. [24] also proposed an authentication method that used bitmaps to generate authentication codes. Furthermore, the resulting image had equivalent quality and higher detection accuracy to [23].

In 2018, ref. [25] proposed a tamper detection method for AMBTC images. The most significant bits (MSBs) of quantized values and bitmaps were hashed to generate the authentication codes. The codes were embedded into the least significant bits (LSBs) of the quantized values using the LSB embedding method. To obtain higher image quality, the MSBs of quantized values were perturbed within a small range and used to generate a set of authentication codes. The code with the smallest embedding error was selected for embedment to provide a higher image quality than [22–24].

In 2018, ref. [26] also proposed an AMBTC authentication method using adaptive pixel pair matching (APPM). Bitmaps and the position information were employed to generate the authentication codes, which were embedded in the quantized values using APPM. A threshold was used to classify blocks into edge and nonedge ones. If the difference of quantized values was larger than the threshold, the block was considered as an edge one; or otherwise, a nonedge block. Edges in an image were considered more informative than the nonedges; therefore, more bits were embedded in edge blocks than nonedge ones to provide more protection to edges.

In 2019, ref. [27] proposed a high-precision authentication method for AMBTC images using matrix encoding (ME). Bitmaps and position information were used to generate 6-bit authentication codes. The generated codes were divided into two equal parts of 3 bits, and embedded into the bitmaps using matrix encoding. To avoid damages to the bitmaps caused by embedding, the positions of to-be-flipped bits in the bitmap were recorded and embedded into the quantized values. This method resulted in higher image quality as well as higher detection accuracy.

Methods [25–27] could detect most tampering and also provided satisfactory image quality. However, some special tampering might escape detection using these methods. For example, the generation of authentication codes in [25] was independent of block position information. Therefore, tampering could not be detected when two blocks were interchanged. In addition, the embedding techniques of these methods did not take the smoothness of blocks into account, which resulted in low image quality. To improve image quality and security, this paper proposes an authentication method for AMBTC images using dual embedding strategies. Blocks are classified into smooth and complex ones according to a predefined threshold, and appropriate embedding strategies are employed based on their smoothness. Blocks are classified as smooth blocks for quantized values less than the threshold, or otherwise, as complex ones. Both quantized values and bitmaps are used to carry authentication codes. However, the difference of quantized values of complex blocks can be large and flipping bits of bitmaps can result in significant distortion. Therefore, the authentication codes of complex blocks will be embedded into the quantized values. In contrast, smooth blocks have lower distortion from flipping bits of bitmaps, and thus authentication codes will be embedded into bitmaps. Since the dual embedding strategies are based on block smoothness, the aim is to obtain higher image qualities. Moreover, the generation of authentication codes in the proposed method is related to the to-be-protected contents, which will increase the probability of detecting some special tampering.

The rest of this paper is organized as follows. Section 2 describes related works and Section 3 presents the proposed method. Sections 4 and 5 show the experimental results and conclusions, respectively.

2. Related Works

This section briefly introduces the concepts of AMBTC compression technique. The APPM and matrix encoding techniques used in the proposed method are also presented. The specific procedures are described in the following three subsections.

2.1. AMBTC Compression Technique

AMBTC is a lossy compression technique [28] that uses low and high quantized values and a bitmap to represent a block. The proposed method is to embed the authentication codes into quantized values and bitmaps to protect the AMBTC images, and the limitations of AMBTC is that the compression ratio and image quality are both relatively low. Let I be the original image. Divide I into N blocks $I = \{I_i\}_{i=0}^{N-1}$ of size $n \times n$, and $I_i = \{I_{i,j}\}_{j=0}^{n\times n-1}$, where $I_{i,j}$ represents the j-th pixel of I_i. Then, scan each block I_i of $\{I_i\}_{i=0}^{N-1}$. To compress block I_i, calculate the mean value m_i of I_i. Average pixels in I_i that are smaller than m_i, and record the result as the low quantized value a_i. On the other hand, the high quantized value b_i is the average of pixels greater than or equal to m_i. Compare m_i and the pixel $I_{i,j}$ to obtain the bitmap B_i of block I_i. If $I_{i,j} \geq m_i$, then $B_{i,j} = 1$; otherwise, $B_{i,j} = 0$, where $B_{i,j}$ represents the j-th bit of B_i. Thus, the compressed code is $C_i = (a_i,\ b_i,\ B_i)$ of I_i. All blocks are compressed in the same way, and the result is the AMBTC compressed codes of image I, which is denoted by $C = \{a_i,\ b_i,\ B_i\}_{i=0}^{N-1}$. To decompress $C_i = (a_i,\ b_i,\ B_i)$, prepare an empty block D_i with the same size of I_i, and $D_{i,j}$ is the j-th pixel of D_i. If $B_{i,j} = 0$, then $D_{i,j} = a_i$; otherwise, $D_{i,j} = b_i$. All codes $C = \{a_i,\ b_i,\ B_i\}_{i=0}^{N-1}$ are decompressed using the same procedures where the decompressed image is $D = \{D_i\}_{i=0}^{N-1}$.

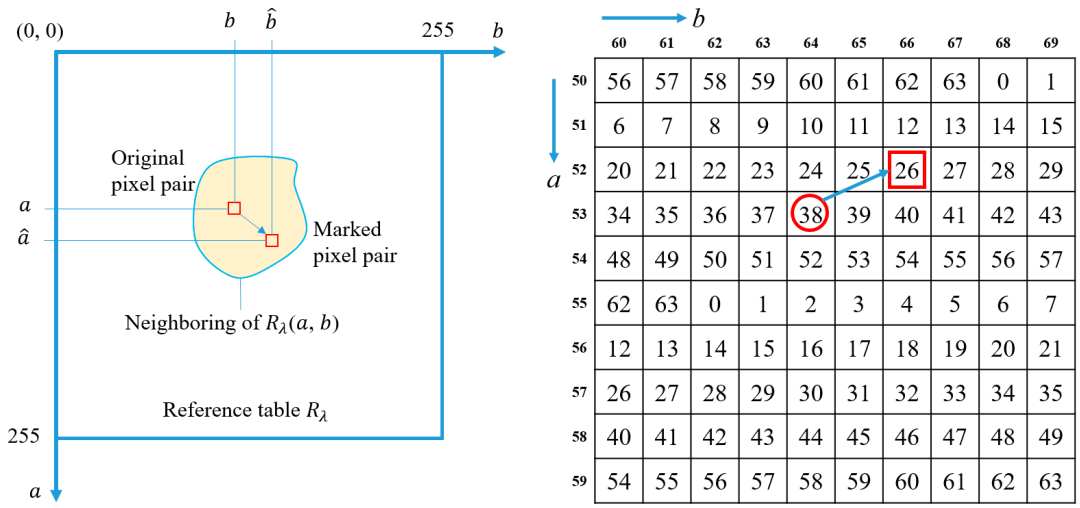
A simple example is given to introduce the procedures of AMBTC compression technique. Let I_i = [36, 34, 41, 42; 37, 35, 43, 46; 35, 39, 44, 41; 36, 41, 42, 48] be the original block of size 4×4. Calculate the mean value of I_i, and $m_i = 40$ is obtained. Pixels in I_i less than 40 are 36, 34, 37, 35, 35, 39 and 36, and the average of these pixels is $a_i = 36$. Similarly, $b_i = 43$ is calculated. To obtain B_i, if $I_{i,j} \geq 40$, then $B_{i,j} = 1$; otherwise, $B_{i,j} = 0$. Therefore, B_i = [0011; 0011; 0011; 0111]. Finally, the AMBTC compressed code C_i = (36, 43, [0011; 0011; 0011; 0111]) is obtained. To decompress C_i, bits 0 and 1 in B_i are decoded by quantized values $a_i = 36$ and $b_i = 43$, respectively, and D_i = [36, 36, 43, 43; 36, 36, 43, 43; 36, 36, 43, 43; 36, 43, 43, 43].

2.2. The Adaptive Pixel Pair Matching (APPM) Technique

In the proposed method, we use the APPM embedding technique to embed the authentication codes into quantized values of complex blocks. The limitation of APPM is that it can only embed at most 8 bits into a pair of quantized values, fortunately only 6 bits are required in our method. The principle of APPM [29] is to embed a digit of base λ into a pixel pair by referring to the reference table R_λ, where R_λ is a table of size 256×256 filled with elements of integers in the range $[0,\ \lambda - 1]$. Let $R_\lambda(a,\ b)$ be the element located in a-th row and b-th column of R_λ. $R_\lambda(a,\ b)$ can be calculated by the following equation:

$$R_\lambda(a,\ b) = (c_\lambda \times a + b) \bmod \lambda, \tag{1}$$

where c_λ is a constant, and $c_{64} = 14$ in this paper. To embed a digit d_λ into $(a,\ b)$, first locate a pixel pair $(\hat{a},\ \hat{b})$ in the vicinity of $R_\lambda(a,\ b)$ that satisfies $R_\lambda(\hat{a},\ \hat{b}) = d_\lambda$ and has the minimum distance to $(a,\ b)$. The located $(\hat{a},\ \hat{b})$ is then employed to replace $(a,\ b)$ and a marked pixel pair is obtained. When extracting the embedded digit d_λ, since R_λ and $(\hat{a},\ \hat{b})$ are known, d_λ can be calculated by $d_\lambda = R_\lambda(\hat{a},\ \hat{b})$. The schematic diagram of APPM is as shown in Figure 1a.



	60	61	62	63	64	65	66	67	68	69
50	56	57	58	59	60	61	62	63	0	1
51	6	7	8	9	10	11	12	13	14	15
52	20	21	22	23	24	25	26	27	28	29
53	34	35	36	37	38	39	40	41	42	43
54	48	49	50	51	52	53	54	55	56	57
55	62	63	0	1	2	3	4	5	6	7
56	12	13	14	15	16	17	18	19	20	21
57	26	27	28	29	30	31	32	33	34	35
58	40	41	42	43	44	45	46	47	48	49
59	54	55	56	57	58	59	60	61	62	63

Figure 1. The examples of APPM embedding method. (**a**) The schematic diagram of the APPM; (**b**) An example of R_{64}.

Following is a simple example to illustrate the APPM embedding procedures. Figure 1b shows a partial reference table R_{64}. Let $(a,\ b) = (53,\ 64)$ be the original pixel pair used to carry the digit $d_{64} = 26$ of base 64. Since $(52,\ 66)$ satisfies $R_{64}(52,\ 66) = 26$ and has the minimum distance to $(53,\ 64)$, the marked pixel pair $(\hat{a},\ \hat{b}) = (52,\ 66)$ is located. To extract the embedded digit, we only need to locate the coordinate $(52,\ 66)$ in R_{64}, and $d_{64} = R_{64}(52,\ 66) = 26$ can be obtained (see Figure 1b).

2.3. The Matrix Encoding

The Matrix encoding is used to embed the authentication codes into bitmaps of smooth blocks. The limitation of Matrix encoding is that only 3 bits can be embedded at a time, and our method requires 6 bits to be embedded, thus a block has to be embedded 2 times. Matrix encoding $(\beta,\ k)$ is an efficient embedding method [30] based on Hamming code, which is a linear error correction code proposed by Richard Wesley Hamming [31]. $(\beta,\ k)$ represents the k secret bits s embedded into the vector $V = \{V_i\}_{i=0}^{\beta-1}$ of length β. Matrix encoding $(\beta,\ k)$ is embedded based on a parity matrix H. To embed s into V, $p = ((H \times V^{\mathrm{T}}) \bmod 2) \bigoplus s^{\mathrm{T}}$ is calculated, where T, $\bigoplus$ and mod represent transpose, exclusive-or and modulo-2 operations, respectively. Note that p is column vector of length k. Let $(p)_{10}$ be the decimal value of p and initialize $V_{\mathrm{M}} = V$ as the marked vector of V. If $(p)_{10} = 0$, then no bit is required to be flipped in V_{M}. Otherwise, flip the $((p)_{10} - 1)$-th bit of V_{M}. The secret bits can be extracted directly by calculating $s = (H \times {V_{\mathrm{M}}}^{\mathrm{T}})^{\mathrm{T}} \bmod 2$.

Next, an example of $(\beta,\ k) = (7,\ 3)$ is taken to introduce the embedding and extraction procedures of matrix encoding. Following equation shows the H of (7, 3):

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}. \tag{2}$$

Let $V = [1, 1, 1, 1, 1, 0, 0]$ be the original vector used to embed the secret bits $s = [1, 0, 0]$. Calculate $p = ((H \times V^{\mathrm{T}}) \bmod 2) \bigoplus s^{\mathrm{T}}$ to get $p = [1,\ 0,\ 1]^{\mathrm{T}}$. Convert p to its decimal value, and $(p)_{10} = 5$. Then, flip the $5 - 1 = 4$-th bit of the initialized marked vector V_{M}, and $V_{\mathrm{M}} = [1, 1, 1, 1, 0, 0, 0]$. The embedded secret bits can be extracted by $s = (H \times {V_{\mathrm{M}}}^{\mathrm{T}})^{\mathrm{T}} \bmod 2 = [1,\ 0,\ 0]$.

3. The Proposed Method

In methods [25–27], authentication codes are generated independently of position information and MSBs of quantized values, which can result in some tampering that cannot be detected. Moreover, these embedding techniques are not designed based on block smoothness, leading to a relatively high image distortion. In this paper, the to-be-protected contents, such as quantized values, are used to generate authentication codes to enhance the security of the image. In addition, different embedding strategies based on block smoothness are used to improve image quality. The smoothness is determined by a predefined threshold T. Given an AMBTC compressed code $C_i = (a_i,\ b_i,\ B_i)$ of block I_i. If $|b_i - a_i| < T$, the block I_i is smooth; otherwise, it is complex. In our method, the authentication codes of smooth blocks are embedded in bitmaps using the matrix encoding, while complex blocks are embedded in quantized values using the APPM. The detailed embedding and authentication procedures are presented in the following subsections. Notice that the detection result is related to the length of authentication codes. In the proposed method, smooth and complex blocks are embedded with the same length of authentication codes, thus they have identical detection performance.

3.1. The Embedment Algorithm of Smooth Blocks

Let $C_i = (a_i,\ b_i,\ B_i)$ be the AMBTC compressed code of a smooth block I_i. Following gives the embedment algorithm of a smooth block I_i.

- Step 1: Divide B_i into $\{B_{i,j}\}_{j=0}^{1}$ and $\{B_{i,j}\}_{j=2}^{15}$, which are employed to generate and carry ac_i, respectively.
- Step 2: Use the bitmap $\{B_{i,j}\}_{j=0}^{1}$, low quantized value a_i, high quantized value b_i, and position information i to generate ac_i using the following equation:

$$ac_i = \text{hash}_6(\{B_{i,j}\}_{j=0}^{1},\ a_i, b_i, i), \tag{3}$$

  where $\text{hash}_6(x)$ is the function that hashes x using the MD5 [32] and reduces the hashed results to 6-bit ac_i using the xor operation.
- Step 3: The 6-bit ac_i is divided into 2 groups of 3 bits denoted as ac_i^0 and ac_i^1. The matrix encoding described in Section 2.3 is then employed to embed ac_i^0 and ac_i^1 into $\{B_{i,j}\}_{j=2}^{8}$ and $\{B_{i,j}\}_{j=9}^{15}$, and we obtain $\{\hat{B}_{i,j}\}_{j=2}^{8}$ and $\{\hat{B}_{i,j}\}_{j=9}^{15}$, respectively.
- Step 4: Concatenate $\{B_{i,j}\}_{j=0}^{1}$, $\{\hat{B}_{i,j}\}_{j=2}^{8}$, and $\{\hat{B}_{i,j}\}_{j=9}^{15}$, and we have the marked bitmap $\hat{B}_i$. Finally, the marked compressed code $\hat{C}_i = (\hat{a}_i,\ \hat{b}_i,\ \hat{B}_i)$ is outputted, where $(\hat{a}_i,\ \hat{b}_i) = (a_i,\ b_i)$.

A simple example is given to illustrate the procedure of generating an authentication code using the hash function $\text{hash}_6(x)$. Suppose the hashed result of x is a 32-bit string '00110110101101001100010000101100'. Then the first 16 bits are xor-ed with the last 16 bits to create a 16 bits '1111001010011000'. Repeat this xor-ed procedure one more time, and we obtain an 8 bits '01101010'. Since our method only requires 6 bits, the last 2 bits are discarded. Therefore, the authentication code '011010' is obtained.

3.2. The Embedment Algorithm of Complex Blocks

For a compressed code $C_i = (a_i,\ b_i,\ B_i)$, if $(b_i - a_i) \geq T$, the block I_i is a complex one and embed the authentication code ac_i into the quantized values $(a_i,\ b_i)$ using APPM with the following algorithm.

- Step 1: Use the following equation to construct the reference table $R_{64}^{\|_i}$:

$$R_{64}^{\|_i}(a_i,\ b_i) = ((c_{64} \times a_i + b_i) + \|_i) \bmod 64, \tag{4}$$

  where $\|_i$ is an random integer generated by a key $\mathcal{K}$, and $c_{64} = 14$.

- Step 2: Use the bitmap B_i and position information i to generate the 6-bit ac_i by

$$ac_i = \text{hash}_6(B_i, i). \tag{5}$$

- Step 3: Once ac_i is obtained, $(ac_i)_{10}$ of base 64 is embedded into the quantized values $(a_i,\ b_i)$ using APPM to obtain the marked quantized values $\hat{a}_i$ and $\hat{b}_i$, where $(ac_i)_{10}$ is the decimal value of ac_i.

In comparison to Equation (1), Equation (4) adds an additional integer $\|_i$ to generate the reference table. Actually, the image quality obtained by referring to $R_{64}^{\|_i}$ is equal to that of R_{64}. However, if ac_i is embedded based on R_{64}, it can be extracted publicly by Equation (1). Therefore, one can tamper with $(\hat{a}_i,\ \hat{b}_i)$ by finding an alternative pixel pair $(\dot{a}_i,\ \dot{b}_i)$ that satisfies $(c_{64} \times \dot{a}_i + \dot{b}_i) \bmod 64 = (c_{64} \times \hat{a}_i + \hat{b}_i) \bmod 64 = ac_i$ to escape detection. In contrast, ac_i in Equation (4) can be obtained only if $\|_i$ is known. Thus, the embedment using $R_{64}^{\|_i}$ is not only more secure than just using R_{64}, but also maintains the same image quality.

3.3. Embedding of Smoothness-Changed Blocks

Since the ac_i of complex block I_i is embedded into the quantized values $(a_i,\ b_i)$, the smoothness of I_i may change to the smooth one if $\left|\hat{b}_i - \hat{a}_i\right| < T$. In this case, it can be processed using the following procedures. Firstly, alter the value of a_i and b_i by one, and the altered results $a_i' = \{a_i - 1, a_i, a_i + 1\}$ and $b_i' = \{b_i - 1, b_i, b_i + 1\}$ are obtained. If $\left|b_i' - a_i'\right| < T$, use the embedding technique described in Section 3.1 to perform the embedment on $(a_i',\ b_i', B_i)$ and obtain $(a_i^*,\ b_i^*,\ B_i^*)$, where $(a_i^*,\ b_i^*) = (a_i',\ b_i')$. Otherwise, the embedment of $(a_i',\ b_i', B_i)$ is performed using the embedding technique of complex block described in Section 3.2 to obtain $(a_i^*,\ b_i^*,\ B_i^*)$, where $B_i^* = B_i$. From all possible codes of $(a_i^*,\ b_i^*,\ B_i^*)$, the code with the smallest embedding error is selected as the final output $(\hat{a}_i, \hat{b}_i, \hat{B}_i)$. Figure 2 shows the embedding framework of the proposed method.

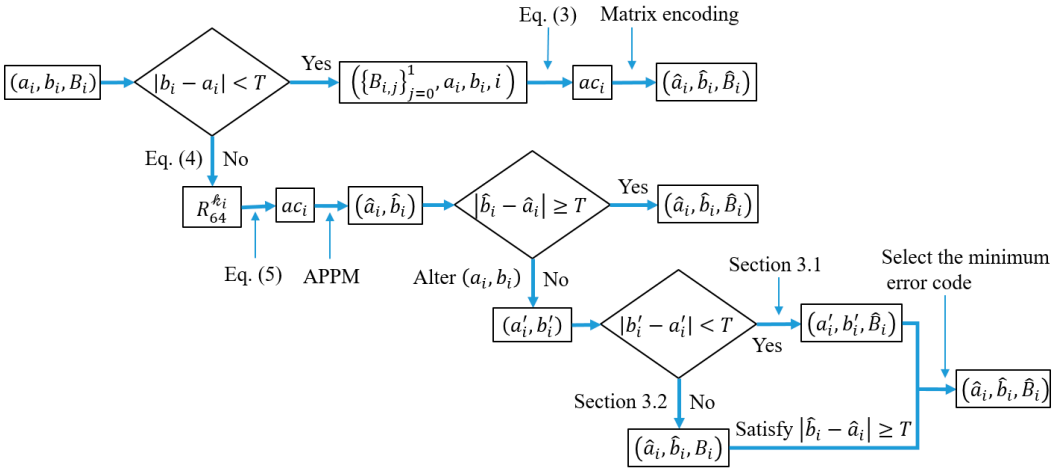


Figure 2. The embedding framework of the proposed method.

Following is a simple example to illustrate the case where a complex block changes to a smooth one after embedment. Let $(a_i,\ b_i,\ B_i) = (67, 73, [0011; 0011; 0011; 0101])$ be the original AMBTC code with $n = 4$, and $T = 6$. Since $|b_i - a_i| = |73 - 67| = 6 \geq T$, the I_i is classified as a complex block. Then, use B_i and i to generate the authentication code ac_i using Equation (5), and employ APPM to embed ac_i into $(a_i,\ b_i)$ to obtain $(\hat{a}_i,\ \hat{b}_i)$. Suppose the marked quantized values $(\hat{a}_i,\ \hat{b}_i) = (68, 73)$. Since $\hat{b}_i - \hat{a}_i = 73 - 68 = 5 < T$, the smoothness of $(\hat{a}_i,\ \hat{b}_i)$ is different from that of $(a_i,\ b_i)$. Thus, the embedment of $(a_i,\ b_i,\ B_i)$ requires additional processing. Alter the values of $(a_i,\ b_i)$ by one unit, and the altered results $(a_i',\ b_i')$ are (66, 72), (66, 73), (66, 74), (67, 72), (67, 73), (67, 74), (68, 72), (68, 73) and

(68, 74). The differences of a'_i and b'_i less than $T = 6$ are (67, 72), (68, 72) and (68, 73), and the technique described in Section 3.1 is used to embed them. The differences of a'_i and b'_i larger than or equal to 6 are (66, 72), (66, 73), (66, 74), (67, 73), (67, 74) and (68, 74), and employ the technique described in Section 3.2 to embed them. Let (67, 72, [0011; 0$\underline{1}$11; 0011; 0$\underline{0}$01]), (68, 72, [0011; 001$\underline{0}$; 0011; 01$\underline{1}$1]), and (68, 73, [001$\underline{0}$; 0011; 0011; $\underline{1}$101]) are the embedded codes with differences of quantified values less than 6, where the underlined bits are the flipped bits. Suppose that (65, 73, B_i), (67, 76, B_i), (66, 73, B_i), (67, 75, B_i), (66, 74, B_i) and (68, 75, B_i) are the codes with differences of quantified values larger than or equal to 6. Decompress these codes and the original codes $(a_i,\ b_i,\ B_i)$ using AMBTC to obtain the decompress block D_i^* and D_i. Then, calculate the squared differences of D_i^* and D_i, which are 68, 64, 68, 32, 72, 8, 32, 16 and 40. Since the code (66, 73, B_i) has the least embedding distortion 8, it can be selected and outputted as the final code $(\hat{a}_i,\ \hat{b}_i, \hat{B}_i) = (66,\ 73,\ B_i)$.

3.4. The Embedding Procedures

In this section, the embedding procedures of the proposed method are described in the following algorithm. Let $C = \{a_i,\ b_i,\ B_i\}_{i=0}^{N-1}$ be the AMBTC codes used to embed the authentication codes. The embedding procedures are listed below.

- Input: AMBTC compressed codes $C = \{a_i,\ b_i,\ B_i\}_{i=0}^{N-1}$, key $\mathcal{K}$, and parameters n and T.
- Output: Marked AMBTC codes $\{\hat{a}_i,\ \hat{b}_i,\ \hat{B}_i\}_{i=0}^{N-1}$.
- Step 1: Scan each code $(a_i,\ b_i,\ B_i)$ in $\{a_i,\ b_i,\ B_i\}_{i=0}^{N-1}$ and calculate the difference of a_i and b_i.
- Step 2: If $|b_i - a_i| < T$, use Equation (3) to hash $\{B_{i,j}\}_{j=0}^{1}$, a_i, b_i and i to generate the 6-bit ac_i. Embed ac_i into $\{B_{i,j}\}_{j=2}^{15}$ using the matrix encoding described in Section 2.3 to obtain $\{\hat{B}_{i,j}\}_{j=2}^{15}$. Concatenate $\{B_{i,j}\}_{j=0}^{1}$ and $\{\hat{B}_{i,j}\}_{j=2}^{15}$, and the marked bitmap $\hat{B}_i$ is obtained. Then, we have the marked code $(\hat{a}_i,\ \hat{b}_i,\ \hat{B}_i)$, where $(\hat{a}_i,\ \hat{b}_i) = (a_i,\ b_i)$.
- Step 3: If $|b_i - a_i| \geq T$, use the key $\mathcal{K}$ to generate $\|_i$, and construct the reference table $R_{64}^{\|_i}$ using Equation (4). Then, use the APPM to embed ac_i into $(a_i,\ b_i)$, and we have $(\hat{a}_i,\ \hat{b}_i)$. If $\left|\hat{b}_i - \hat{a}_i\right| \geq T$, the marked code $(\hat{a}_i,\ \hat{b}_i,\ \hat{B}_i)$ is outputted and $\hat{B}_i = B_i$. Otherwise, $(a_i,\ b_i,\ B_i)$ is embedded using the technique described in Section 3.3 to obtain $(\hat{a}_i,\ \hat{b}_i,\ \hat{B}_i)$.
- Step 4: Repeat Steps 1–3 until all codes are embedded and output the marked codes $\{\hat{a}_i,\ \hat{b}_i,\ \hat{B}_i\}_{i=0}^{N-1}$, key $\mathcal{K}$, and parameters n and T.

3.5. The Authentication Procedures

To authenticate whether the codes $\{\widetilde{a}_i,\ \widetilde{b}_i,\ \widetilde{B}_i\}_{i=0}^{N-1}$ have been tampered with, we first regenerate the authentication code $\widetilde{ac}_i$ using either Equations (3) or (5) according to the difference of $\widetilde{a}_i$ and $\widetilde{b}_i$. Then, extract the code $\widetilde{eac}_i$ embedded in $(\widetilde{a}_i,\ \widetilde{b}_i,\ \widetilde{B}_i)$, and compare $\widetilde{eac}$ with $\widetilde{ac}_i$. If $\widetilde{eac} = \widetilde{ac}_i$, the code $(\widetilde{a}_i,\ \widetilde{b}_i,\ \widetilde{B}_i)$ is untampered with; otherwise, it is tampered with. The detailed authentication procedures are listed as follows.

- Input: To-be-authenticated codes $\{\widetilde{a}_i,\ \widetilde{b}_i,\ \widetilde{B}_i\}_{i=0}^{N-1}$, key $\mathcal{K}$, and parameters n and T.
- Output: The detection result.
- Step 1: Scan each code $(\widetilde{a}_i,\ \widetilde{b}_i,\ \widetilde{B}_i)$ in $\{\widetilde{a}_i,\ \widetilde{b}_i,\ \widetilde{B}_i\}_{i=0}^{N-1}$ and calculate the difference of $\widetilde{a}_i$ and $\widetilde{b}_i$.
- Step 2: If $\left|\widetilde{b}_i - \widetilde{a}_i\right| < T$, use Equation (3) to hash $\{\widetilde{B}_{i,j}\}_{j=0}^{1}$, $\widetilde{a}_i$, $\widetilde{b}_i$ and i to regenerate 6-bit $\widetilde{ac}_i$. The matrix encoding is employed to extract $\widetilde{eac}_i$ from $\{\widetilde{B}_{i,j}\}_{j=2}^{15}$.

- Step 3: If $\left| \widetilde{b}_i - \widetilde{a}_i \right| \geq T$, employ $\mathcal{K}$ to generate $\|_i$, and construct the $R_{64}^{\|_i}$ by Equation (4). Besides, $\widetilde{B}_i$ and i are employed to regenerate $\widetilde{ac}_i$ by Equation (5). Then, use APPM to extract $\widetilde{eac}_i$ embedded in $(\widetilde{a}_i, \widetilde{b}_i)$.
- Step 4: Compare $\widetilde{ac}_i$ and $\widetilde{eac}_i$ to judge whether the code $(\widetilde{a}_i, \widetilde{b}_i, \widetilde{B}_i)$ has been tampered with. If $\widetilde{ac}_i = \widetilde{eac}_i$, the code is untampered with. Otherwise, it is tampered with.
- Step 5: Repeat Steps 1–4 until all blocks have been detected, which refers to the coarse detection in our method.
- Step 6: The refined detection, described here, is used to improve detection accuracy. If the top and bottom, left and right, top left and bottom right, or top right and bottom left blocks of an untampered block have been determined as tampered with, the untampered block is redetermined to be a tampered one. Repeat this procedure until no other blocks are redetermined and we have finished the authentication procedures.

4. Experimental Results

To evaluate the effectiveness of our method, we perform several experiments on a set of grayscale images. Eight images of size 512 × 512, namely, Lena, Jet, Baboon, Tiffany, Sailboat, Splash, Peppers, and House are used as test images for the experiments, as shown in Figure 3. These test images can be obtained from the USC-SIPI image database [33]. Comparisons of image quality and detectability between prior works [25–27] and the proposed method are also shown in this section. In the experiments, the peak signal-to-noise ratio (PSNR) is employed to measure the marked image quality:

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

where MSE is the mean square error of the marked image and the original image. Equation (6) shows that the smaller the MSE is, the larger is the PSNR. The structural similarity (SSIM) metric [34] is also employed to measure the similarity between the marked and original images. The SSIM is calculated by:

$$\text{SSIM}(x, y) = \frac{(2\mu_x\, \mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{7}$$

where x and y represent the original and marked images. μ_x, μ_y and σ_x, σ_y are the mean value and standard deviation of x, y, respectively. σ_{xy} is the covariance of x and y, and C_1 and C_2 are constants. The value of SSIM is within the range [0, 1], and the larger the SSIM is, the higher the visual quality of the marked image is.

4.1. The Performance of the Proposed Method

In this paper, we use a threshold T to classify blocks into smooth and complex ones and embed them using different techniques. The PSNR and SSIM are related to the value of T. Table 1 shows the comparisons between PSNR and SSIM for the test images when the threshold T is set from 0 to 10. As shown in the table, the highest PSNR is achieved when $T = 6$ or $T = 7$. As T becomes smaller or larger, the PSNR decreases gradually. For example, the Jet image has a PSNR of 43.32 dB for $T = 6$, which is higher than when $T = 5$ and $T = 7$. Besides, the peak values of SSIM are achieved when $T = 6$. Thus $T = 6$ is adopted in the proposed method. This experiment was performed using the Python programming language, and the average time required to embed an image is less than one second when $T = 6$.

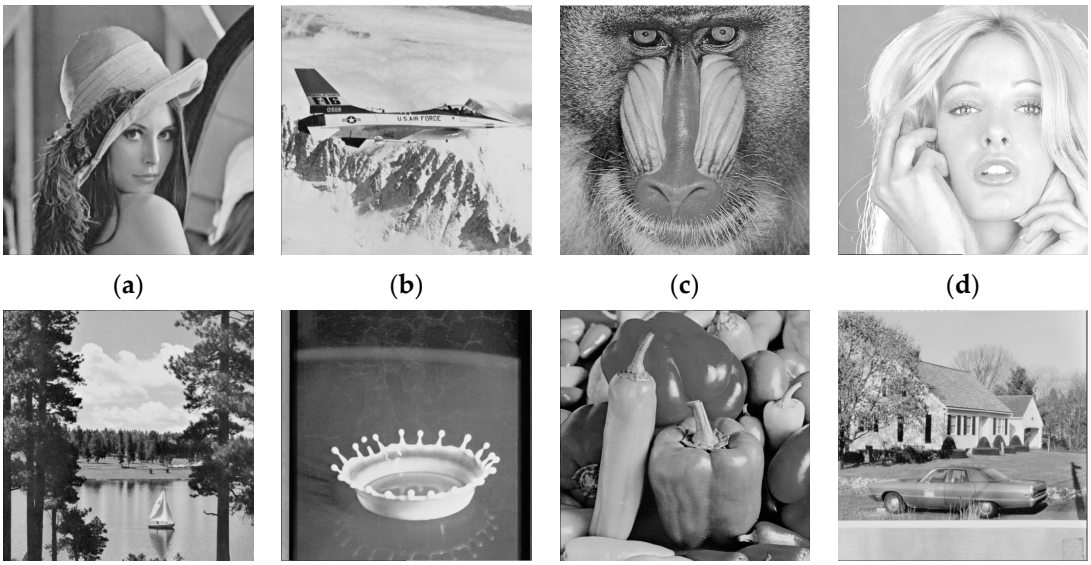


Figure 3. Eight test images. (**a**) Lena; (**b**) Jet; (**c**) Baboon; (**d**) Tiffany; (**e**) Sailboat; (**f**) Splash; (**g**) Peppers; (**h**) House.

Table 1. PSNR and SSIM comparisons with various T value.

T	Metric	Lena	Jet	Baboon	Tiffany	Sailboat	Splash	Peppers	House
$T = 0$	PSNR	41.02	41.02	41.00	41.01	41.05	41.02	41.02	41.06
	SSIM	0.9564	0.9505	0.9867	0.9612	0.9723	0.9501	0.9613	0.9654
$T = 1$	PSNR	41.08	41.14	41.01	41.09	41.07	41.11	41.06	41.12
	SSIM	0.9574	0.9523	0.9867	0.9622	0.9727	0.9514	0.9618	0.9664
$T = 2$	PSNR	41.31	41.80	41.01	41.45	41.21	41.63	41.18	41.84
	SSIM	0.9613	0.9638	0.9867	0.9666	0.9746	0.9580	0.9637	0.9788
$T = 3$	PSNR	41.62	42.39	41.02	41.80	41.34	42.04	41.34	42.02
	SSIM	0.9660	0.9727	0.9868	0.9709	0.9764	0.9626	0.9657	0.9810
$T = 4$	PSNR	42.03	42.94	41.03	42.32	41.49	42.61	41.57	42.22
	SSIM	0.9710	0.9784	0.9870	0.9756	0.9780	0.9679	0.9683	0.9828
$T = 5$	PSNR	42.31	43.19	41.04	42.64	41.59	43.06	41.79	42.39
	SSIM	0.9738	0.9805	0.9870	0.9780	0.9792	0.9714	0.9707	0.9844
$T = 6$	PSNR	42.48	43.32	41.05	42.78	41.64	43.31	41.93	42.48
	SSIM	0.9748	0.9808	0.9871	0.9786	0.9797	0.9727	0.9716	0.9849
$T = 7$	PSNR	42.49	43.31	41.04	42.77	41.63	43.33	41.94	42.47
	SSIM	0.9741	0.9806	0.9870	0.9783	0.9793	0.9722	0.9711	0.9847
$T = 8$	PSNR	42.39	43.26	40.99	42.70	41.53	43.24	41.83	42.43
	SSIM	0.97301	0.9798	0.9865	0.9776	0.9783	0.9711	0.9696	0.9843
$T = 9$	PSNR	42.24	43.17	40.90	42.55	41.37	43.09	41.65	42.35
	SSIM	0.9719	0.9793	0.9855	0.9766	0.9765	0.9698	0.9675	0.9837
$T = 10$	PSNR	42.06	43.02	40.78	42.33	41.13	42.93	41.41	42.22
	SSIM	0.9705	0.9786	0.9843	0.9753	0.9747	0.9688	0.9654	0.9830

Figure 4 shows the plot of threshold T versus PSNR for various test images. When $T = 0$, the PSNRs of the eight test images are around 41 dB. Among these images at $T = 6$, Jet has the highest PSNR, while Baboon achieves the lowest PSNR, which is about 2 dB lower than that of Jet. This is because image quality is dependent on the ratio of smooth blocks for a given threshold, which is demonstrated in the following experiments.

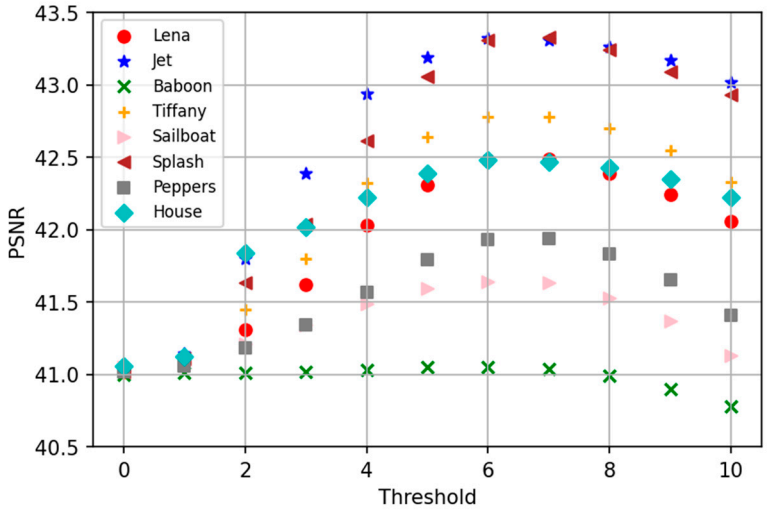


Figure 4. The plot of threshold T versus PSNR for various test images.

Table 2 shows the ratio of smooth blocks and the PSNR of images when $T = 6$. We observe that under the same threshold, the larger the ratio of smooth blocks, the higher the PSNR, implying that the proposed method is more effective. The reason is that the differences of quantized values of smooth blocks are close to each other, thus the error caused by flipping bits of bitmaps is also small. For example, Lena, Jet, Tiffany, Splash and House contain more than 40% smooth blocks, and the PSNR of these images can reach more than 42 dB. However, the PSNR of Baboon is the lowest (41.04 dB), which is due to the fact that it has only 6.99% smooth blocks. PSNR is a metric to measure an embedding method, and higher PSNR means that the embedding method is more effective. In these 8 test images, the PSNRs of all images are higher than 41 dB, implying the effectiveness of our method.

Table 2. The relation between the ratio of smooth blocks and the PSNR of images.

Image	Number of Smooth Blocks	Ratio of Smooth Blocks	PSNR
Lena	8695	53.07%	42.49
Jet	9758	59.56%	43.31
Baboon	1146	6.99%	41.04
Tiffany	9518	58.09%	42.78
Sailboat	3638	22.20%	41.64
Splash	11,820	72.14%	43.33
Peppers	5524	33.71%	41.91
House	6616	40.38%	42.47

The following experiment demonstrates the detectability of the proposed method. The marked codes of Lena are tampered such that the tampered decompressed image shows a daisy on Lena's hat, as shown in Figure 5a. Figure 5b shows the tampered region indicated by black blocks, while Figure 5c,d present the coarse and refined detection results of Figure 5a. An image contains a total of $(512 \times 512)/(4 \times 4) = 16384$ blocks, and this

experiment tampers with 2176 blocks with a tampering rate of 13.28%. Figure 5c shows that a few scattered blocks are not detected in the coarse detection. Nevertheless, those undetected blocks can be detected in the refined detection, as shown in Figure 5d.

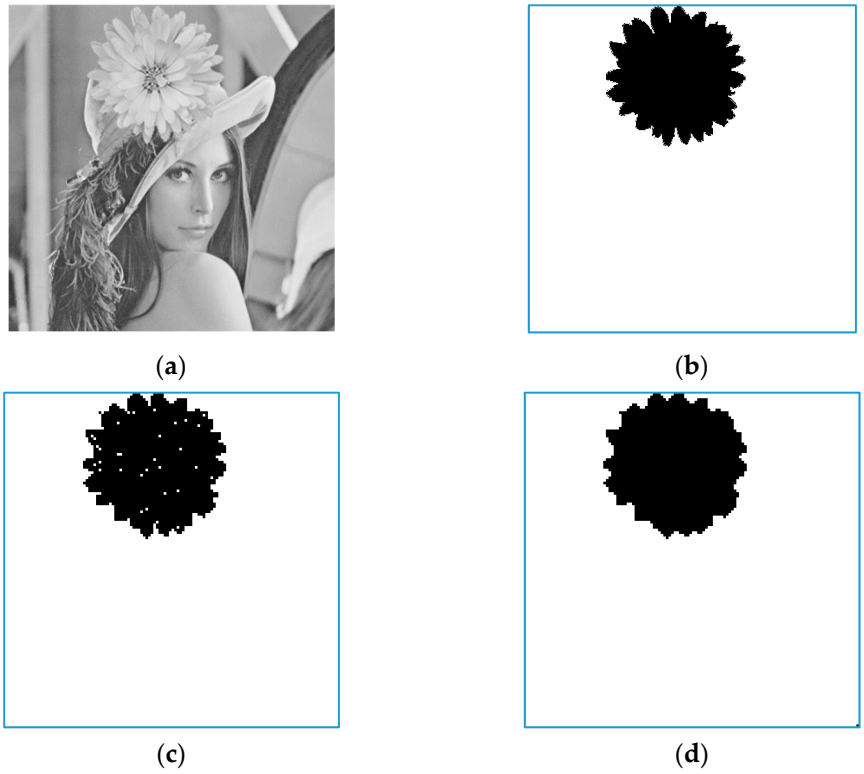


Figure 5. The tampered and detection results. (**a**) The tampered image; (**b**) The tampered region; (**c**) The coarse detection result; (**d**) The refined detection result.

4.2. PSNR Comparisons with Prior Works

In this section, we compare the PSNR between [25–27] and the proposed method, as shown in Table 3. In this experiment, 6-bit authentication code is embedded in all methods, and $T = 6$ is used in the proposed method. As shown in the table, the PSNR of the proposed method is the highest compared to [25–27]. The averaged PSNR of our method is 42.36 dB, which is $42.36 - 39.71 = 2.65$, $42.36 - 40.97 = 1.39$ and $42.36 - 40.35 = 2.01$ dB higher than [25–27], respectively. Moreover, if an image contains a larger ratio of smooth blocks, the improvement of PSNR obtained by our method is higher. For example, compared to [27], the improvement of Baboon is $41.04 - 40.69 = 0.35$ dB, while Splash is $43.33 - 40.13 = 3.20$ dB, reflecting the effectiveness of the proposed method.

Table 3. PSNR comparisons with [25–27] (in dB).

Method	Lena	Jet	Baboon	Tiffany	Sailboat	Splash	Peppers	House	Average
[25]	39.71	39.69	39.73	39.76	39.67	39.67	39.70	39.70	39.71
[26]	40.98	41.08	40.96	40.89	40.96	40.98	40.99	40.93	40.97
[27]	40.32	40.07	40.69	40.22	40.55	40.13	40.49	40.25	40.35
Proposed	42.49	43.31	41.04	42.78	41.64	43.33	41.94	42.47	42.36

An additional 200 images of sized 512×512 obtained from BOWS-2 image database [35] are used to explore the applicability of the compared methods influenced by image smoothness. To better evaluate the performance, we sort 200 images according to the number of smooth blocks of images in the ascending order, as shown in Figure 6. The figure shows

that when the number of smooth blocks is close to 0, the PSNR improvement of our method is not significant compared to [25–27]. However, as the number of smooth blocks increases, the improvement becomes more obvious, as shown in Figure 6. From the above analysis, the proposed method is more suitable for smooth images compared to complex ones.

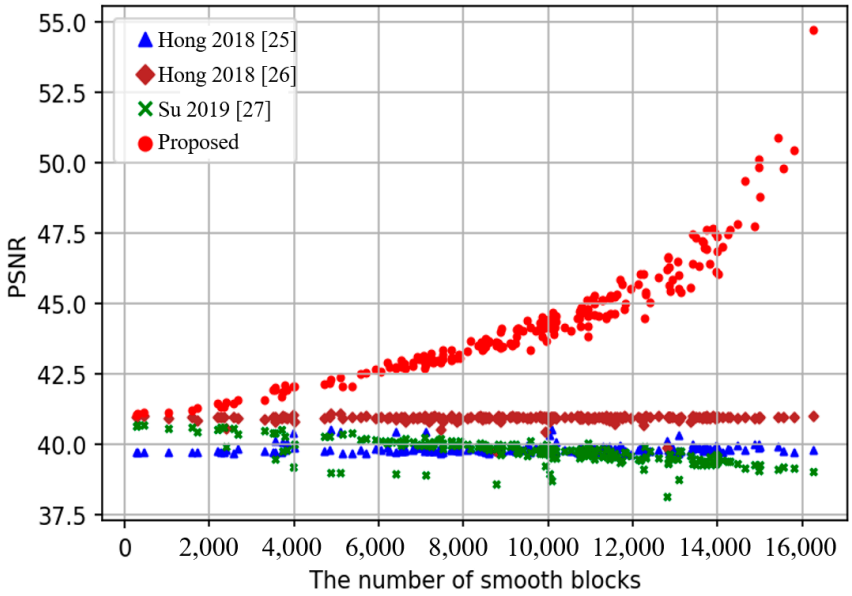


Figure 6. Comparisons between PSNR and the number of smooth blocks for 200 images.

4.3. Detectability Comparisons with Prior Works

This section shows the detectability comparisons of [25–27] and the proposed method. In this experiment, the marked compressed codes of Sailboat are tampered with by adding a bird. Figure 7a,b show the tampered image and the region of tamper, while Figure 7c–f present the detection results of [25–27] and the proposed method. The results show all methods can achieve a satisfactory detection result.

Different metrics are used to measure the detection results of Figure 7, as shown in Table 4. The number of tampered pixels (NTB) in Figure 7a is 19,712, and the tampering rate is $19712/(512 \times 512) = 7.52\%$, where. True positive (TP) represents the number of tampered pixels detected as tampered ones, while false negative (FN) is the number of tampered pixels incorrectly detected as untampered ones. The refined detection rate is calculated by TP/(NTB). In all methods, the length of authentication codes embedded in each block is 6. Therefore, the collision probability is $1/2^6 = 1.56\%$, i.e., the coarse detection rate is as high as $100\% - 1.56\% = 98.44\%$. The table shows that the refined detection rates of [25–27] and the proposed method are up to 99%, which are better than the coarse detection rates and meet the theoretical values.

Table 4. Detectability comparisons using different metrics.

Method	[25]	[26]	[27]	Proposed
Number of tampered blocks (NTB)	19,712			
Tampering rate	7.52%			
True positive (TP)	19,680	19,696	19,696	19,712
False negative (FN)	32	16	16	0
The refined detection rate	99.83%	99.91%	99.91%	100%

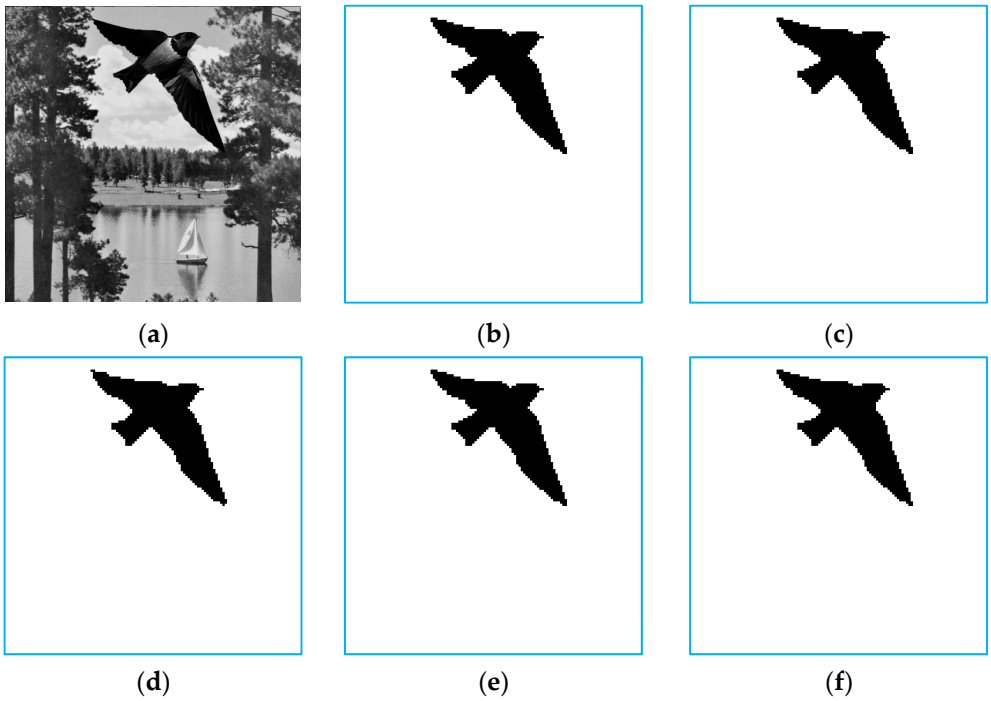
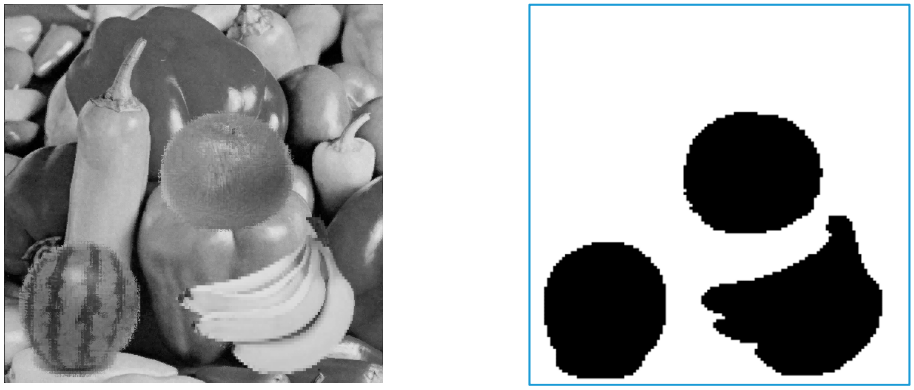


Figure 7. The tampered and detection results of [25–27] and the proposed method. (**a**) The tampered image; (**b**) The tampered region; (**c**) Detection result of [25]; (**d**) Detection result of [26]; (**e**) Detection result of [27]; (**f**) Detection result of the proposed method.

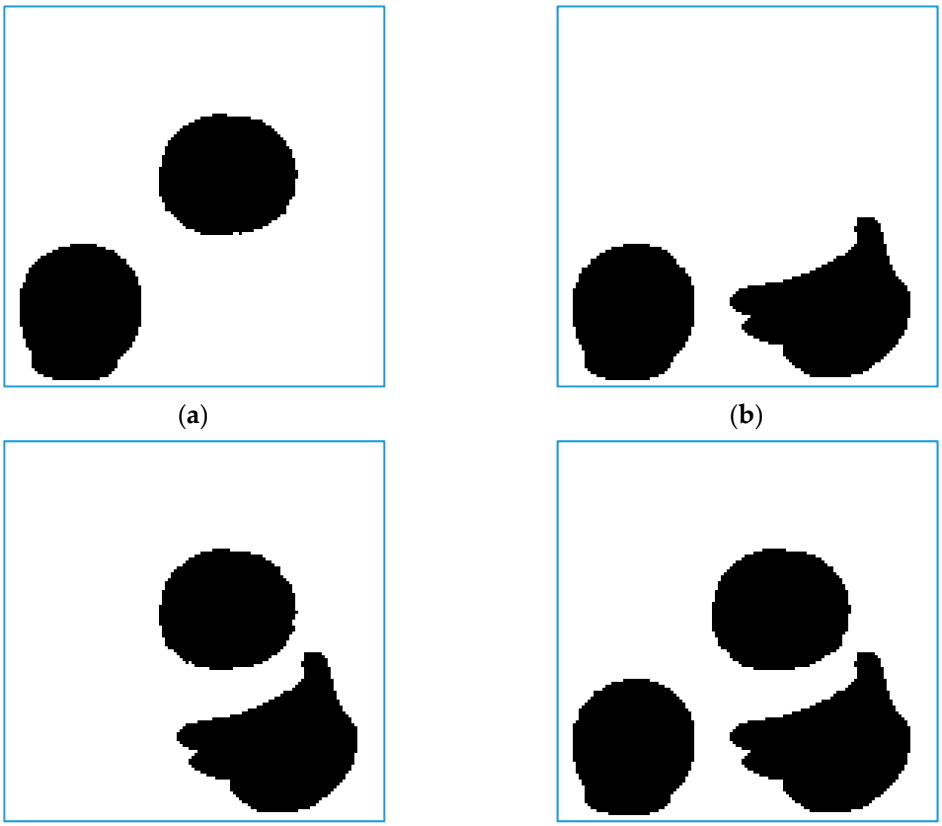
A good detection method should be able to detect any kind of tampering. The following experiments are performed with some special tampering to further compare the detectability of [25–27] and the proposed method. The special tampering consisted of the marked codes of Peppers by adding bananas, orange and watermelon, and Figure 8a,b present the tampered image and regions. Let $\{\hat{a}_i, \hat{b}_i, \hat{B}_i\}_{i=0}^{N-1}$ be the marked codes of Peppers. Suppose $\{a_i^{\mathrm{B}}, b_i^{\mathrm{B}}, B_i^{\mathrm{B}}\}_{i=0}^{N^{\mathrm{B}}-1}$, $\{a_i^{\mathrm{O}}, b_i^{\mathrm{O}}, B_i^{\mathrm{O}}\}_{i=0}^{N^{\mathrm{O}}-1}$ and $\{a_i^{\mathrm{W}}, b_i^{\mathrm{W}}, B_i^{\mathrm{W}}\}_{i=0}^{N^{\mathrm{W}}-1}$ are the codes of the bananas, orange and watermelon, and N^{B}, N^{O} and N^{W} represent the number of blocks of these images, respectively. For tampering with bananas, codes $\{a_i^{\mathrm{B}}, b_i^{\mathrm{B}}, B_i^{\mathrm{B}}\}_{i=0}^{N^{\mathrm{B}}-1}$ are used to replace $\{\hat{a}_i, \hat{b}_i, \hat{B}_i\}_{i=0}^{N^{\mathrm{B}}-1}$. Then, from $\{\hat{a}_i, \hat{b}_i, \hat{B}_i\}_{i=0}^{N-1}$, find the codes $\{\hat{a}_i^{\mathrm{B}}, \hat{b}_i^{\mathrm{B}}, \hat{B}_i^{\mathrm{B}}\}_{i=0}^{N^{\mathrm{B}}-1}$ which are the closest to $\{a_i^{\mathrm{B}}, b_i^{\mathrm{B}}, B_i^{\mathrm{B}}\}_{i=0}^{N^{\mathrm{B}}-1}$, and the found codes are used to replace $\{a_i^{\mathrm{B}}, b_i^{\mathrm{B}}, B_i^{\mathrm{B}}\}_{i=0}^{N^{\mathrm{B}}-1}$. For orange's tampering, the codes $\{a_i^{\mathrm{O}}, b_i^{\mathrm{O}}, B_i^{\mathrm{O}}\}_{i=0}^{N^{\mathrm{O}}-1}$ are used to replace $\{\hat{a}_i, \hat{b}_i, \hat{B}_i\}_{i=0}^{N^{\mathrm{O}}-1}$. Then, the pixel pair $(\dot{a}_i^{\mathrm{O}}, \dot{b}_i^{\mathrm{O}})$ that satisfy $(14 \times \dot{a}_i^{\mathrm{O}} + \dot{b}_i^{\mathrm{O}}) \bmod 64 = (14 \times \hat{a}_i + \hat{b}_i) \bmod 64$ and has the minimum distance to $(a_i^{\mathrm{O}}, b_i^{\mathrm{O}})$ is found. Finally, we use $(\dot{a}_i^{\mathrm{O}}, \dot{b}_i^{\mathrm{O}})$ to replace $(a_i^{\mathrm{O}}, b_i^{\mathrm{O}})$ for $0 \le i \le N^{\mathrm{O}} - 1$. To achieve the tampering of watermelon, the 5 MSBs of $\{a_i^{\mathrm{W}}, b_i^{\mathrm{W}}\}_{i=0}^{N^{\mathrm{W}}-1}$ are use to replace that of $\{\hat{a}_i, \hat{b}_i\}_{i=0}^{N^{\mathrm{W}}-1}$.

Figure 9 shows the refined detection results of [25–27] and the proposed method. Figure 9a–c show that [25–27] do not detect the tampering of bananas, orange and watermelon, respectively. This is because the generation of authentication codes of [25–27] is independent of the protected contents, e.g., position information and MSBs of quantized values. However, the proposed method can detect these tampering, as shown in Figure 9d, demonstrating the superiority of our method against other works.



(**a**) (**b**)

Figure 8. The tampering of Peppers image. (**a**) The tampered image; (**b**) The tampered regions.



(**a**) (**b**)

(**c**) (**d**)

Figure 9. Detectability comparisons between [25–27] and the proposed method. (**a**) Detection result of [25]; (**b**) Detection result of [26]; (**c**) Detection result of [27]; (**d**) Detection result of the proposed method.

The components for generating authentication codes, embedding techniques and detectability of methods [25–27] and the proposed method are summarized in Table 5. Furthermore, the performance in recovery domain for methods [20,21] are also compared. Comparisons are made between the various detection methods for "Detection of bananas", "Detection of orange" and "Detection of watermelon" with tampering collages of banana, orange and watermelon, respectively. The table shows that methods [20,25–27] cannot detect all these types of tampering. The reason is that the components used for generating authentication codes are not included in all to-be-protected contents. In contrast,

method [20] and the proposed method use contents that need to be protected to generate authentication codes, and thus they are able to detect these tampering.

Table 5. Comparisons between prior works and the proposed method.

Method	Components for Generating Authentication Codes	Embedding Techniques	Detection of Bananas	Detection of Orange	Detection of Watermelon
[20]	Bitmaps and position	APPM	Yes	No	Yes
[21]	Recovery codes and position	Turtle Shell	Yes	Yes	Yes
[25]	MSBs of quantized values and bitmaps	LSB	No	Yes	Yes
[26]	Bitmaps and position	APPM	Yes	No	Yes
[27]	Bitmaps and position	Matrix encoding	Yes	Yes	No
Proposed	quantized values, bitmaps and position	APPM and matrix encoding	Yes	Yes	Yes

5. Conclusions

In this paper, we proposed an efficient authentication method with a high image quality for AMBTC images. Based on the smoothness of blocks, blocks are classified into smooth and complex ones. To enhance the security, to-be-protected contents including position information, bitmaps and quantized values are employed to generate the authentication codes. Moreover, a key is added to construct the reference table used in APPM to further protect the authentication codes. According to the smoothness of blocks, the authentication codes are embedded into the bitmap using matrix encoding for smooth block and in the quantized values using APPM for complex blocks. Experimental results show that in comparisons to prior works, the proposed method achieves a better detection results and higher image quality.

Author Contributions: X.Z., J.C. and W.H. contributed to the conceptualization, methodology, and writing of this paper. Z.-F.L. and G.Y. conceived the simulation setup, formal analysis and conducted the investigation. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Jasra, B.; Moon, A.H. Color image encryption and authentication using dynamic DNA encoding and hyper chaotic system. *Expert Syst. Appl.* **2022**, *206*, 117861. [CrossRef]
2. Hossain, M.S.; Islam, M.T.; Akhtar, Z. Incorporating deep learning into capacitive images for smartphone user authentication. *J. Inf. Secur. Appl.* **2022**, *69*, 103290. [CrossRef]
3. Qin, C.; Ji, P.; Zhang, X.; Dong, J.; Wang, J. Fragile image watermarking with pixel-wise recovery based on overlapping embedding strategy. *Signal Process.* **2017**, *138*, 280–293. [CrossRef]
4. You, C.; Zheng, H.; Guo, Z.; Wang, T.; Wu, X. Tampering detection and localization base on sample guidance and individual camera device convolutional neural network features. *Expert Syst.* **2022**, *40*, e13102. [CrossRef]
5. Hussan, M.; Parah, S.A.; Jan, A.; Qureshi, G.J. Hash-based image watermarking technique for tamper detection and localization. *Health Technol.* **2022**, *12*, 385–400. [CrossRef]
6. Zhao, D.; Tian, X. A Multiscale Fusion Lightweight Image-Splicing Tamper-Detection Model. *Electronics* **2022**, *11*, 2621. [CrossRef]
7. Hussan, M.; Parah, S.A.; Jan, A.; Qureshi, G.J. Self-embedding framework for tamper detection and restoration of color images. *Multimed. Tools Appl.* **2022**, *81*, 18563–18594. [CrossRef]

8. Zhou, X.; Hong, W.; Weng, S.; Chen, T.S.; Chen, J. Reversible and recoverable authentication method for demosaiced images using adaptive coding technique. *J. Inf. Secur. Appl.* **2020**, *55*, 102629. [CrossRef]
9. Wang, Q.; Xiong, D.; Alfalou, A.; Brosseau, C. Optical image authentication scheme using dual polarization decoding configuration. *Opt. Lasers Eng.* **2019**, *112*, 151–161. [CrossRef]
10. Molina, J.; Ponomaryov, V.; Reyes, R.; Sadovnychiy, S.; Cruz, C. Watermarking framework for authentication and self-recovery of tampered colour images. *IEEE Lat. Am. Trans.* **2020**, *18*, 631–638. [CrossRef]
11. Wu, X.; Yang, C. Invertible secret image sharing with steganography and authentication for AMBTC compressed images. *Signal Process. Image* **2019**, *78*, 437–447. [CrossRef]
12. Zhang, X.; Wang, S.; Qian, Z.; Feng, G. Reversible fragile watermarking for locating tampered blocks in JPEG images. *Signal Process.* **2010**, *90*, 3026–3036. [CrossRef]
13. Hong, W.; Wu, J.; Lou, D.C.; Zhou, X.; Chen, J. An AMBTC authentication scheme with recoverability using matrix encoding and side match. *IEEE Access* **2021**, *9*, 133746–133761. [CrossRef]
14. Zhang, T.; Weng, S.; Wu, Z.; Lin, J.; Hong, W. Adaptive encoding based lossless data hiding method for VQ compressed images using tabu search. *Inform. Sci.* **2022**, *602*, 128–142. [CrossRef]
15. Pan, Z.; Wang, L. Novel reversible data hiding scheme for two-stage VQ compressed images based on search-order coding. *J. Vis. Commun. Image R.* **2018**, *50*, 186–198. [CrossRef]
16. Li, Y.; Chang, C.C.; Mingxing, H. High capacity reversible data hiding for VQ-compressed images based on difference transformation and mapping technique. *IEEE Access* **2020**, *8*, 32226–32245. [CrossRef]
17. Battiato, S.; Giudice, O.; Guarnera, F.; Puglisi, G. CNN-based first quantization estimation of double compressed JPEG images. *J. Vis. Commun. Image R.* **2022**, *89*, 103635. [CrossRef]
18. Yao, H.; Mao, F.; Qin, C.; Tang, Z. Dual-JPEG-image reversible data hiding. *Inform. Sci.* **2021**, *563*, 130–149. [CrossRef]
19. Cogranne, R.; Giboulot, Q.; Bas, P. Efficient steganography in JPEG images by minimizing performance of optimal detector. *IEEE Trans. Inf. Foren. Sec.* **2022**, *17*, 1328–1343. [CrossRef]
20. Chen, T.S.; Zhou, X.; Chen, R.; Hong, W.; Chen, K. A high fidelity authentication scheme for AMBTC compressed image using reference table encoding. *Mathematics* **2021**, *9*, 2610. [CrossRef]
21. Lin, C.C.; Liu, X.; Zhou, J.; Tang, C.Y. An image authentication and recovery scheme based on turtle Shell algorithm and AMBTC-compression. *Multimed. Tools Appl.* **2022**, *81*, 39431–39452. [CrossRef]
22. Hu, Y.C.; Lo, C.C.; Chen, W.L.; Wen, C.H. Joint image coding and image authentication based on absolute moment block truncation coding. *J. Electron. Imaging* **2013**, *22*, 013012. [CrossRef]
23. Li, W.; Lin, C.C.; Pan, J.S. Novel image authentication scheme with fine image quality for BTC-based compressed images. *Multimed. Tools Appl.* **2016**, *75*, 4771–4793. [CrossRef]
24. Chen, T.H.; Chang, T.C. On the security of a BTC-based-compression image authentication scheme. *Multimed. Tools Appl.* **2018**, *77*, 12979–12989. [CrossRef]
25. Hong, W.; Zhou, X.Y.; Lou, D.C.; Huang, X.Q.; Peng, C. Detectability improved tamper detection scheme for absolute moment block truncation coding compressed images. *Symmetry* **2018**, *10*, 318. [CrossRef]
26. Hong, W.; Chen, M.J.; Chen, T.S.; Huang, C.C. An efficient authentication method for AMBTC compressed images using adaptive pixel pair matching. *Multimed. Tools Appl.* **2018**, *77*, 4677–4695. [CrossRef]
27. Su, G.D.; Chang, C.C.; Lin, C.C. High-precision authentication scheme based on matrix encoding for AMBTC-compressed images. *Symmetry* **2019**, *11*, 996. [CrossRef]
28. Lema, M.; Mitchell, O. Absolute moment block truncation coding and its application to color image. *IEEE Trans. Commun.* **1984**, *32*, 1148–1157. [CrossRef]
29. Hong, W.; Chen, T.S. A novel data embedding method using adaptive pixel pair matching. *IEEE Trans. Inf. Foren. Sec.* **2012**, *7*, 176–184. [CrossRef]
30. Liu, S.; Fu, Z.; Yu, B. Rich QR codes with three-layer information using Hamming code. *IEEE Access* **2019**, *7*, 78640–78651. [CrossRef]
31. Hamming, R.W. Error detecting and error correcting codes. *Bell Labs Tech. J.* **1950**, *29*, 147–160. [CrossRef]
32. Menezes, A.J.; Van Oorschot, P.C.; Vanstone, S.A. *Handbook of Applied Cryptography*; CRC Press: Boca Raton, FL, USA, 1996.
33. The USC-SIPI Image Database. Available online: http://sipi.usc.edu/database/ (accessed on 10 January 2023).
34. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [CrossRef] [PubMed]
35. BOWS-2 Image Database. Available online: http://bows2.ec-lille.fr/ (accessed on 10 January 2023).