TSIA: A Novel Image Authentication Scheme for AMBTC-Based Compressed Images Using Turtle Shell Based Reference Matrix

Chih-Cheng Chen1,5, Member, IEEE, Chin-Chen Chang2,*, Fellow, IEEE, Chia-Chen Lin3,*, Member, IEEE, and Guo-Dong Su2,3

1School of Information Engineering, Jimei University, Xiamen, Fujian 361021, China
2Department of Information Engineering and Computer Science, Feng Chia University, Taichung 40724, Taiwan, R.O.C.
3Engineering Research Center for ICH Digitalization and Multi-Source Information Fusion, Fujian University
4Department of Computer Science and Information Management, Providence University, Taichung 43301, Taiwan
5Department of Information Engineering and Computer Science, National Chin-Yi University of Technology, Taichung 43301, Taiwan

Corresponding authors: Chin-Chen Chang and Chia-Chen Lin (e-mail: alan3c@gmail.com and mhlin3@pu.edu.tw)

This work was supported by the project of Ministry of Science and Technology of Taiwan (No. MOST 106-2221-E-035-013-MY3)

ABSTRACT This paper proposes a novel authentication scheme for absolute moment block truncation coding (AMBTC) of a compressed image using turtle shell based data hiding. For simplicity, we call it turtle shell based image authentication method (TSIA in short). For each block, a 3-bit authentication code (AC) is generated by combining the bitmap with a pseudo-random sequence and is concealed into the corresponding quantization levels with the use of a reference matrix. In order to solve the problem of having a high quantization level lower or equal to a low quantization level caused after the hiding operation, an iterative embedding mechanism is employed in the proposed TSIA scheme. Experimental results demonstrate that the proposed TSIA scheme outperforms previous works in the watermarked image quality with an increased PSNR of 0.16 dB and achieves high tampering detecting accuracy.

INDEX TERMS Image authentication, AMBTC, compressed code, turtle shell based data hiding

I. INTRODUCTION

As computer software and hardware continue to rapidly develop, people are increasingly using the Internet to communicate and exchange files. As a result, a tremendous number of multimedia files, such as images, videos or audios, either are quickly transmitted widely to almost anywhere or are increasingly stored in cloud servers. Among various kinds of multimedia files, images are a major media category. Since digital image processing techniques have made significant progress, the cost of processing a digital image is now much less than in the past. As a consequence, digital images can also easily carry extra information or can be manipulated by malicious users, which can be done so as to be almost imperceptible. Therefore, there is a need for image authentication to evaluate the integrity of received digital images. This is even more crucial today than in the past.

In general, image authentication schemes can be classified into two categories: cryptography-based authentication [1–4] and watermarking-based authentication [5–10, 16–23]. The most straightforward way to provide image authentication is by cryptographic methods, such as through the joint use of asymmetric-key encryption and a digital hash function. Traditional cryptographic methods can usually suffice to protect the security and integrity of a digital image with the help of signatures, which are stored in advance. However, a cryptography-based authentication method is poor in terms of tamper localization. Thus, watermarking-based authentication has been applied as a feasible solution to localize the tampered area of an image. Generally, in watermarking-based authentication, the process of generating the authentication code will be performed first, with the use of the content of an image or a random sequence. Subsequently, the authentication code, called a watermark, is embedded into the image with as little distortion as possible. In the authentication phase, it is possible to judge whether the image has been tampered or not by comparing the difference between the extracted authentication code from the watermarked image and the recalculated authentication code, which is done in the same manner as that in the generation phase. Furthermore, an advantage of the watermarking-based method is that it can localize the detailed positions of the tampered blocks.

The first fragile watermarking used to check the integrity of image was proposed in 1995 by Walton [5]. Later, other
Fragile watermarking schemes utilizing cryptographic technologies were proposed by Wong [6, 7]. For each block, the watermark is calculated by a hashing operation using the MSBs of pixel value, image size, and one key. After the generation of a watermark, it is embedded into the LSB of pixels in the image. Fridrich and Goljan [8] proposed a self-embedding fragile watermarking method to protect the content of an image. In their method, for each block, a specified number of the lowest frequency DCT coefficients are chosen as the self-recovery information, then these values are encoded into a fixed bit-string and embedded into the LSBs of the pixels in one block. The scheme of Fridrich and Goljan cannot survive image modification beyond the least two significant bits. In 2005, Lin et al. [9] proposed an effective fragile watermarking scheme. Their scheme divides an image into non-overlapping blocks sized $4 \times 4$ and a watermark is generated for each block, including six recovery bits and two authentication bits. Simple operations such as parity check and comparison between average intensities are then conducted. Their tamper detection is effective because it is based on a hierarchical structure, such that the accuracy of tamper localization can be ensured. However, it easily suffers from a tampering coincidence problem. To overcome this problem, Lee et al. [10] stored duplicate copies of the watermark for each block. These double watermarks are then embedded into different blocks, respectively. In this way, their scheme can resist the tampering coincidence problem. At the same time, such a mechanism provides a higher probability for recovery of the tampered blocks due to the duplicate copies of the watermark.

Image authentication can also be used in the compression domain. Digital images are generally stored in compressed formats because the raw image files require a considerable amount of storage space. To reduce the bandwidth requirement for transmitted digital images, researchers have developed diverse compression techniques, such as discrete cosine transform (DCT) [11], discrete wavelet transform (DWT) [12], vector quantization (VQ) [13], side match vector quantization (SMVQ) [14] and absolute moment block truncation coding (AMBTC) [15]. The first two methods are based on the frequency domain and provide great compression results, but require many complicated computations. In comparison, the last three methods compress images in the spatial domain using simple computations to obtain acceptable compression rates. Among these methods, AMBTC is the most efficient method for image compression with high efficiency and an acceptable compression ratio. In 2009, Jiang et al. [16] proposed an image authentication scheme in which a watermark is embedded into the host image according to the parity of the reconstruction levels of the BTC quantizers. In 2013, a joint image compressed and image authentication technique was proposed for compressed images of AMBTC [17]. In this scheme, a pseudo random generator is used to generate the authentication code that is embedded into the bitmaps of AMBTC-compressed image blocks. The embedded bitmaps and these quantization levels are further compressed to reduce the required storage space. To obtain higher visual quality, in [18], Hu et al. proposed a novel tamper detection scheme for the compressed images of BTC. In Hu et al.’s scheme, the authentication codes of the image blocks are generated by a predefined random seed and are then embedded into the difference values between each pair of the quantization levels to avoid a significant loss of image quality. The size of their authentication codes for each compressed image block can be determined by users. In 2014, Lin et al.[19] employed the parity of the bitmap, which is derived by AMBTC technology, to generate an authentication code. In their scheme, the authentication code is embedded into the quantization levels of each AMBTC-compressed image block. Comparing to Hu et al. [17], Lin et al.’s scheme offers better visual quality and good detection accuracy. Later in 2016, Li et al. [20] proposed a fragile watermark scheme using AMBTC technology with the help of a designed reference matrix. The watermark is generated using a random number generator and embedded into the quantization levels of compressed code for every image block. In the same year, Nguyen et al. [21] proposed a novel reversible image authentication scheme to protect the integrity of an image based on an adaptive prediction error expansion (PEE) technique. The watermark is inserted flexibly into different properties of an image block. Their scheme achieves good tamper detection and can restore the watermarked image to its original image if there is no damage to the image. In 2017, Lin et al. [22] proposed a hybrid image authentication method using AMBTC. They classified the image into two groups: smooth and complex. For each group, a different embedding method is adopted. The authentication code is embedded into the bitmap for smooth block. Meanwhile, the authentication code is embedded into the quantization levels for a complex block according to a reference matrix. In the detection phase, a hybrid strategy is used to ensure a higher detection rate. Their scheme also provides improved image quality. In 2018, an efficient image authentication method for AMBTC-compressed images using adaptive pixel pair matching was proposed by Hong et al. [23]. In their scheme, the image blocks are classified into edge and non-edge. Each block type utilizes an embedding strategy. For each block, the bitmap and location information are fed to a hashing function to generate the authentication code. Their scheme can adaptively justify the size of the authentication code for each block according to user requirements. The authentication codes including the bitmap information are embedded into the quantization levels using an adaptive reference table. Their scheme can detect tampering in the bitmap or quantization levels, which is quite a novel point of view and different from previous works. However, their embedding strategy may break the relation between high and low quantization values, such that attackers can transfer the watermarked AMBTC compression codes into a special domain and conduct the AMBTC encoding.
Moreover, the hidden authentication codes could be damaged and increase the probability of false detection.

In this paper, we propose a novel authentication scheme for AMBTC-compressed image using turtle shell based data hiding, called TSIA, to address various problems mentioned above and increase detection performance while maintaining good image quality. The proposed TSIA scheme aims to improve the quality of the embedded image while retaining high detection accuracy. The rest of this paper is organized as follows. We briefly review the basic concept of AMBTC and prior relevant AMBTC-based image authentication schemes in Section 2. Section 3 presents a detailed exposition of the proposed algorithm. In Section 4, we experimentally investigate the relationship between the size of the authentication codes and the quality of the embedded image. We also provide a performance comparison for our proposed TSIA and existing image authentication schemes. Finally, we present our brief conclusions in Section 5.

II. RELATED WORKS

This section briefly reviews the basic concepts of AMBTC and several prior relevant AMBTC-based image authentication schemes. Finally, we review turtle shell based data hiding which will be used as our hiding strategy.

A. AMBTC COMPRESSION TECHNIQUE

Block truncation coding was first proposed by Delp and Mitchell in 1979 for grayscale image compression. Absolute moment block truncation coding (AMBTC) [15] is a variant of BTC that can further simplify the computation while maintaining better image quality for the compressed image.

Here, let us assume that the image is sized of M × N. Let each image be divided into a number of blocks sized n × n. Let \( p_{(1,1)}, p_{(1,2)}, \ldots, p_{(n,1)}, p_{(n,2)}, \ldots, p_{(n,n)} \) be values of the pixels for each block. Then, firstly, the mean value \( \overline{p} \) of this block is computed by:

\[
\overline{p} = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} p_{(i,j)}.
\]  

(1)

Later, in a given block, all pixels are classified into two sub-groups: \( SG_0 \) and \( SG_1 \), where \( SG_0 \) contains pixels whose values are less than the mean value \( \overline{p} \), and \( SG_1 \) contains pixels whose values are greater than or equal to the mean value \( \overline{p} \). The classification rule is described by:

\[
p_{(i,j)} = \begin{cases} 
SG_0 & \text{if } p_{(i,j)} < \overline{p} \\
SG_1 & \text{if } p_{(i,j)} \geq \overline{p}.
\end{cases}
\]  

(2)

After classification, we can calculate the two quantization values H and L by:

\[
H = \frac{1}{h_{num}} \sum_{i=1}^{n} \sum_{j=1}^{n} p_{(i,j)} & \text{if } p_{(i,j)} \in SG_1,
\]  

(3)

\[
L = \frac{1}{l_{num}} \sum_{i=1}^{n} \sum_{j=1}^{n} p_{(i,j)} & \text{if } p_{(i,j)} \in SG_0,
\]  

(4)

where \( h_{num} \) and \( l_{num} \) is the number of elements of sub-groups \( SG_0 \) and \( SG_1 \), respectively. The bitmap BM of the given block is derived as follows:

\[
BM = \begin{cases} 
0 & \text{if } p_{(i,j)} \in SG_0 \\
1 & \text{if } p_{(i,j)} \in SG_1.
\end{cases}
\]  

(5)

Finally, the block trio \((H, L, BM)\) is derived and it is the compression code for a given block. Generally, a block is sized as 4×4 pixels with either BTC or AMBTC. The decoding process of an AMBTC-based compressed image can be done as follows:

\[
p_{(i,j)} = \begin{cases} 
L & \text{if } BM_{(i,j)} = 0 \\
H & \text{if } BM_{(i,j)} = 1,
\end{cases}
\]  

for \( 1 \leq i \leq n \) and \( 1 \leq j \leq n \).

B. REVIEW OF HONG ET AL.’S METHOD

In 2018, Hong et al. [23] proposed an efficient authentication method for AMBTC-compressed images using adaptive pixel pair matching. In their method, the embedding capacity is up to 4 bits for each block while guaranteeing good image quality. First, it is necessary to construct reference table \( R_e \) by:

\[
R_e(x, y) = \text{mod}(c_x \times x + y, \delta),
\]  

(7)

where \( \delta \) is a base that means the length of the authentication code used, \((x, y)\) is the coordinate in the reference table and \( R_e(x, y) \) is its corresponding value. \( c_x \) is a constant with values of 1, 2, 3, 6, and 7 when \( \delta \) is 2, 4, 8, 16, and 32, respectively. Here, \( 0 \leq x, y \leq 255 \).

Later, the original image is divided into non-overlapping blocks sized 4×4. Then compression operation is performed using AMBTC technology to generate the compressed code \((H, L, BM)\). At the same time, the blocks are separated into edge blocks and non-edge blocks by:

\[
\text{block}_i \in \begin{cases} 
\text{edge}, & |H_i - L_i| > T, \\
\text{non-edge}, & |H_i - L_i| \leq T,
\end{cases}
\]  

(8)

where \( i \) represents the \( i \)-th block, and \( T \) is a threshold. In Hong et al.’s [23] scheme, the requirements for the length of the authentication code can be chosen adaptively and generated for different types of blocks. For the \( i \)-th block, the authentication code is generated by:

\[
AC_i^e = \text{Hash}(BM_i, \text{location information } i),
\]  

(9)

where \( AC_i^e \) means that the \( e \)-bit authentication code is used for the \( i \)-th block.

For an edge block, the \( e \)-bit authentication code is embedded with the help of reference table \( R_{e_i} \) by:

\[
\text{Minimize} : (H_i - H_i')^2 + (L_i - L_i')^2,
\]  

(10)

where \( R_{e_i}(H_i', L_i') = AC_i^e \) and \( |H_i' - L_i'| > T \).

For a non-edge block, the \( e \)-bit authentication code is embedded with the help of reference table \( R_{e_{\infty}} \) by:

\[
\text{Minimize} : (H_i - H_i')^2 + (L_i - L_i')^2,
\]  

(11)
where \( R_{0,0} (H', L') = AC_{j}^{\infty} \) and \( |H' - L'| \leq T \).

Hong et al.'s scheme classifies blocks into two types to protect the edges over non-edge because the edge blocks help the human eye to understand the image features.

C. TURTLE SHELL BASED DATA HIDING

Before a data embedding procedure is executed, a reference matrix \( M \) of size 256 \( \times \) 256 needs to be generated [30]. Reference matrix \( M \) is composed of several adjacent turtle shells. Each turtle shell is a hexagon that contains eight different elements, including six edge digits and two back digits, ranging from 0 to 7. The construction rules of the reference matrix \( M \) must be followed: in the same row, the value of each element goes up with a magnitude of “1” and is conducted with a modulo 8 operation; and in the same column, the value of each element goes up with a magnitude of “2” or “3” in turn. The final value of each element in matrix \( M \) should be result which is computed by modulo 8.

This reference matrix \( M \) is utilized in data embedding and data extraction, since the construction rules need to be shared in these two procedures to create a consistent matrix. We can define an identical matrix by setting a value with coordinate (0, 0) equal to 0, as shown in Fig. 1. Each pixel pair in the cover image corresponds to the row value \( P_i \) and column value \( P_{i+1} \) in matrix \( M \). The location where the pixel \((P_i, P_{i+1})\) is located is denoted as \( M(P_i, P_{i+1}) \).

![FIGURE 1. Example of the partial reference matrix M](Image)

Once reference matrix \( M \) is generated, proceed to data hiding. Two examples are given to show the process of turtle shell data hiding. First, assume that the to-be-carried secret data is 0, and the current pixel pair \((P_i, P_{i+1})\) of the cover image is (6, 3). The value of \( M(6, 3) \) is 5 in reference matrix \( M \). Moreover, \( M(6, 3) \) is the back digit and its value is not equal to 0. Hence, we should look for a pixel pair whose value is equal to 0 in the current turtle shell. Finally, a pixel pair (6, 4) is used to carry secret data 0 since \( M(6, 4) = 0 \). Again, let us assume the secret data is also 0 and the current pixel pair \((P_i, P_{i+1})\) is (3,7). The value of \( M(3, 7) \) is different from 0 and it localizes at an edge of the turtle shell. Hence, a pixel pair whose value is equal to 0 will be searched for within these three turtle shells related to \( M(3,7) \). As a result, a pixel pair (4, 8) is used to carry secret data 0.

III. PROPOSED SCHEME

This section introduces the proposed TSIA scheme based on turtle shell-based data hiding. The proposed TSIA scheme consists of three procedures: an image authentication code hiding procedure, an image authentication code extraction procedure and tamper detection procedure. In our TSIA scheme, reference matrix \( M \) serves as our reference matrix. Therefore, the reference matrix construction procedure is skipped here.

A. AUTHENTICATION CODE HIDING PROCEDURE

Before conducting the image authentication code hiding procedure, it is necessary to generate the authentication code that will be used to verify the tampered area. To generate the authentication code, we first use cover image \( I \) to derive its AMBTC compression codes, in which each block carries 4 \( \times \) 4 pixels. For each block, a pseudo-random number generator (PRNG) is employed by a predefined seed to generate a 112-bit random stream. Later, 128-bit \( \text{BM}' \) is generated by combining the 112-bit random stream with a 16-bit BM. Finally, MD5 is applied to generate the ac-bit authentication code, where \( ac \) can be 3 bits or 4 bits. Once the image authentication code is derived, the image authentication code hiding procedure begins. Given a cover image \( I \) with the size of \( M \times N \), cover image \( I \) conducts the AMBTC encoding procedure and generates a block trio \((H, L, \text{BM})\) for each block. Then, according to reference matrix \( M \), a authentication code \( ac \) can be embedded into quantization levels \((H, L)\) of block trio \((H, L, \text{BM})\) in each block as shown in Fig. 2.

The algorithm for the authentication code hiding procedure is described as follows.

Input: Original image \( I \).

Output: New set of compression codes \((H, L, BM')\).

Step1: Divide the original cover image \( I \) into several non-overlapping blocks and perform AMBTC compression encoding operation as mentioned in Subsection 2.1, to generate a block trio\((H, L, \text{BM})\) for each block.

Step2: Employ a pseudo-random number generator (PRNG) with seed key1 to generate a 112-bit random stream. Later, contact the 112-bit random stream with a 16-bit BM to generate a 128-bit BM', for each block.

Step3: Use MD5 to work on a 128-bit BM' to generate a 3-bit authentication code \( ac \) for the current processing block.

Step4: Construct reference matrix \( M \).
FIGURE 2. Flowchart of data embedding procedure

Step5: Embed authentication code $ac$ into a pixel pair $(H, L)$ and then obtain the new pair $(\tilde{H}, \tilde{L})$.

Step6: Check whether the $\tilde{H} > \tilde{L}$. If it is satisfied, output the new block trio $(\tilde{H}, \tilde{L}, BM)$ and go to Step9; otherwise, go to Step 7.

Step7: Change the current bitmap $BM$ by modifying one bit “1” to “0” or one bit “0” to “1” and maintain the caused distortion to be the least.

Step8: Generate a new 3-bit authentication code for the next block and go to Step 5.

Step9: If all blocks have been processed, collect all new block trios as a new set of compression codes $(\tilde{H}, \tilde{L}, BM)$. Otherwise, select block trio $(H, L, BM)$ for the next block and go to Step 2.

After this process, the compressed code $(\tilde{H}, \tilde{L}, \tilde{BM})$ of a cover image is generated. Then the owner can decide to directly transmit the watermarked compression codes or transform the watermarked image into stego-image $I'$ and then transmit to the receivers. Note that Hong et al.'s scheme achieves an efficient authentication method for AMBTC-compressed code using adaptive pixel pair matching, but their method could be compromised if the watermarked AMBTC-compressed code is simply decoded with the traditional AMBTC first and then is encoded again with the traditional ABMTC. This is because Hong et al.'s method could not always keep $H$ larger than $L$ after the authentication code hiding procedure. Therefore, we propose a novel authentication code scheme to solve this problem.

B. AUTHENTICATION CODE EXTRACTION PROCEDURE

When stego-image $I'$ is transmitted to the receiver, we can create the same reference matrix $M$ with the processing steps mentioned in Subsection2.3. The stego-image is divided into several non-overlapping blocks, as was done in the authentication code embedding procedure. For each block, we can encode a compressed code denoted as $(H, L, BM)$ for convenience. Each pixel pair $(H, L)$, is mapped to reference matrix $M$. Then the authentication code extraction procedure is given as follows:

Input: Stego-image $I'$.

Output: Extracted authentication code $AC_{extract}$.

Step1: Divide stego-image $I'$ into several non-overlapping blocks, and compress each block into AMBTC code $(H, L, BM)$ as mentioned in Subsection 2.1.

Step2: Generate reference matrix $M$, as it was done in the authentication code embedding procedure.

Step3: Extract the value based on reference matrix $M$ with the coordinate $(H, L)$. The extracted value is the authentication code for one block.

Step4: Perform Step 3 until all blocks are completed.

Step5: Collect all extracted authentication codes to be $AC_{extract}$.

C. TAMPER DETECTION PROCEDURE

Details for the tamper detection procedure algorithm is described as follows.

Input: Stego-image $I'$, random seed $key_1$.

Output: Tampered map $TM$.

Step1: Divide the stego-image $I'$ into several non-overlapping blocks, and compress each block into AMBTC code $(H, L, BM)$ as mentioned in Subsection 2.2.

Step2: Employ PRNG with random seed $key_1$ to generate a 112-bit random stream again. Later, combine the 112-bit random stream with a 16-bit $BM$ to generate a 128-bit code, for one block. Use MD5 to perform a hash function operation on the 128-bit code and then generate a 3-bit authentication code for the current block and denote this as $AC_{recalu}$.

Step3: Extract the authentication code as mentioned in Subsection 3.2 and denote this as $AC_{extract}$. 

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/
Step 4: Perform the first round of tamper detection by comparing $AC_{\text{extract}}$ and $AC_{\text{recalcu}}$. If they are different, the corresponding block is marked as a tampered block in $TM$. If they are equal, the corresponding block is marked as a non-tampered block.

Step 5: If all blocks are accomplished, the first round for the tampered map is completed. Otherwise, find the next block, compress it into the AMBTC code $(H, L, BM)$, and then go to Step 3.

Step 6: Perform the second round of tamper detection. Conduct neighborhood elimination for all blocks. Perform neighborhood elimination from left to right and from top to bottom. Blocks marked as tampered are further marked as non-tampered if there are four or less tampered blocks in its $3 \times 3$ block-neighborhood. Blocks marked as non-tampered are further marked as tampered if there are six or more tampered blocks in its $3 \times 3$ block-neighborhood. After this process, the second round of the tampered map is achieved.

### IV. EXPERIMENTAL RESULTS

This section presents the experimental results to prove the superiority of our proposed TSIA scheme. The proposed TSIA scheme was tested with the USC-SIPI image database [29], which included the standard grayscale images of “Lena”, “Baboon”, “Tiffany”, “Boat”, “Jet”, “House”, “Splash”, and “Peppers” shown in Fig. 3. Our experiments were implemented by Matlab R2017b, running on a personal computer with a Windows 10 operating system. The CPU was an Intel Xeon E3-1225 v5, 3.3GHz, and had 8GB of memory. The size of each test image was $512 \times 512$ pixels.

![Images](a) Lena (b) Baboon (c) Tiffany (d) Boat

![Images](e) Jet (f) House (g) Splash (h) Peppers

**FIGURE 3.** Eight $512 \times 512$ greyscale images

In the first experiment, peak signal-to-noise ratio (PSNR) was used to evaluate the image quality of the watermarked AMBTC-compressed images (dB) as:

$$\text{PSNR} = 10 \log_{10} \left( \frac{255^2 \times R \times C}{\sum_{i=1}^{R} \sum_{j=1}^{C} (I_{ij} - I'_{ij})^2} \right), \quad (12)$$

where $I_{ij}$ and $I'_{ij}$ refer to the pixels located at the $i$-th row and the $j$-th column of cover image $I$ and stego-image $I'$, respectively. Additionally, parameters $R$ and $C$ represent the height and width of an image.

To evaluate how the base of embedded authentication codes affect the image quality of the watermarked AMBTC-compressed image, we set the authentication code to be 3 bits for each block in the test image. Table 1 shows that our proposed image authentication scheme maintains a slightly higher image quality than that of the Hong et al.’s scheme [23]. For example, the average PSNR of Hong et al.’s scheme using $\alpha = 8$ and our proposed scheme was 32.44 dB and 32.60 dB, respectively.

**Table 1. Comparison of AMBTC, Hong et al.’s scheme and the proposed scheme for image quality (unit: dB, block size: 4x4)**

<table>
<thead>
<tr>
<th>Images</th>
<th>AMBTC</th>
<th>Hong et al.’s scheme [23] ($\alpha = 8$)</th>
<th>Our proposed TSIA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lena</td>
<td>33.21</td>
<td>33.13</td>
<td>33.13</td>
</tr>
<tr>
<td>Baboon</td>
<td>26.97</td>
<td>26.95</td>
<td>26.96</td>
</tr>
<tr>
<td>Tiffany</td>
<td>35.76</td>
<td>35.61</td>
<td>36.86</td>
</tr>
<tr>
<td>Boat</td>
<td>31.16</td>
<td>31.11</td>
<td>31.11</td>
</tr>
<tr>
<td>Jet</td>
<td>31.99</td>
<td>31.92</td>
<td>31.89</td>
</tr>
<tr>
<td>House</td>
<td>30.90</td>
<td>30.85</td>
<td>30.93</td>
</tr>
<tr>
<td>Splash</td>
<td>36.77</td>
<td>36.58</td>
<td>36.59</td>
</tr>
<tr>
<td>Peppers</td>
<td>33.42</td>
<td>33.33</td>
<td>33.31</td>
</tr>
<tr>
<td>Average</td>
<td>32.53</td>
<td>32.44</td>
<td>32.60</td>
</tr>
</tbody>
</table>

Furthermore, we also compared the HVS-PSNRs [27] (human visual system PSNR) provided by the proposed scheme and Guo and Liu’s scheme [27] under various number of embedded authentication codes. HVS-PSNR is considered as another image quality evaluation approach, since it is involved in the former measurement to provide more objective quality assessment. Fig. 4 shows the average HVS-PSNRs of the proposed scheme is around 45.28 dB, which is better than that of Guo and Liu’s scheme (40.10 dB). This means that our approach has a satisfactory visual effect.

**FIGURE 4.** HVS-PSNR comparison between proposed scheme and Guo and Liu’s scheme
A. TAMPER DETECTION PERFORMANCE

In order to evaluate the effectiveness of the proposed image authentication scheme, five different criteria were applied and are defined as follows:

1. False positive rate (FPR) [29]: error in classifying non-tampered pixels as tampered ones. Mathematically defined as:

\[ FPR = \frac{\text{False classified pixels}}{\text{Total tampered pixels}} \times 100\% \]  

(13)

2. False negative rate (FNR) [29]: error in classifying tampered pixels as non-tampered ones. Mathematically defined as:

\[ FNR = \frac{\text{False classified pixels}}{\text{Total non-tampered pixels}} \times 100\% \]  

(14)

3. Tamper detection rate (TDR) [29]: the detection rate of tampered pixels in the overall tampered area. Mathematically defined as:

\[ TDR = \frac{\text{Detected tampered pixels}}{\text{Total no. of tampered pixels}} \times 100\% \]  

(15)

The following paragraphs present six cases to test and demonstrate the effectiveness of our proposed image authentication scheme.

Case 1: Lena is attacked by a cropping operation.

Figs. 5 and 6 show first-stage, second-stage detection, error detection, and Err2 and Err3 results of the proposed scheme when ac is set to 3 bits. Fig. 5(d) presents the region tampered which is conducted by a cropping attack that was successfully identified with the second-stage detection. Fig. 6 shows that TDR, FPR and FNR were 88.0333%, 0%, and 0.7401%, respectively, when only the first-stage detection is employed. However, when the second-stage detection is applied, TDR, FPR and FNR improved to 100%, 0%, and 0%, respectively.

Case 2: Lena is tampered with a flower that is placed on Lena’s shoulder. After embedding the authentication code into the Lena image, the obtained AMBTC-compressed images quality was 33.1325 dB, as shown in Fig. 7(a). Tamper detection results for each stage with various embedded foundations for Case 2 are shown in Fig. 8.

Case 3: The quantization level of the AMBTC-compressed code of Lena is tampered. Here, the quantization levels of Lena’s shoulder are modified and the PSNR of the modified AMBTC-compressed images is 33.0675 dB as shown in Fig.
9(b). Tamper detection results for each stage for Case 3 are shown in Fig. 10.

![Tamper detection results for each stage for Case 3](image)

**FIGURE 9.** Attack the H or L: (a) Lena, and (b) tampered Lena with modified quantization levels

Case 4: The bitmap of the AMBTC-compressed code of the Lena image is tampered as shown in Fig. 11(b). Tamper detection results for each stage for Case 4 are shown in Fig. 12.

![Tamper detection results for each stage for Case 4](image)

**FIGURE 10.** Tamper detection results for each stage for Case 3

**FIGURE 11.** Attack the bitmap: (a) Lena, and (b) Lena with tampered bitmap

Case 5: The quantization levels of the AMBTC-compressed code of Lena are tampered by a “copy and paste a flower attack” on Lena’s shoulder as shown in Fig. 13. Tamper detection results for each stage for Case 5 are shown in Fig. 14.

![Tamper detection results for each stage for Case 5](image)

**FIGURE 12.** Tamper detection results for each stage for Case 4

**FIGURE 13.** Attack the H, L of the AMBTC-compressed code and copy and paste a flower: (a) Lena, and (b) tampered Lena with modified H, L by copying and pasting a flower

Case 6: The bitmap of the AMBTC-compressed code is tampered by a “copy and paste a flower attack” on Lena’s shoulder as shown in Fig. 15. Tamper detection results for each stage for Case 6 are shown in Fig. 16.

![Tamper detection results for each stage for Case 6](image)

**FIGURE 14.** Tamper detection results for each stage for Case 5

**FIGURE 15.** Attack the bitmap of the AMBTC-compressed code and copy and paste a flower: (a) Lena and (b) tampered Lena with a modified bitmap from a flower

**FIGURE 16.** Tamper detection results for each stage with various embedded foundations for Case 6
From Fig. 17, it is clear that the detection results of the second-stage are more specific than those of the first-stage no matter which attacks were involved.

Table 2 demonstrates the PSNRs of original Lena and FPR, FNR, and TDR results under six attacks with our proposed TSIA scheme. For Lena image, the average detection rate of tampered pixels in the overall tampered area are around 99.8175%. We also conducted experiments for the smooth Splash image and the complex Baboon image, resulting in the average TDR of 98.7272% and 99.2284%, respectively. This is because our proposed TSIA scheme is not only based on a turtle shell but also correctly maintain the relationship between H and L after data embedding operation.

### B. COMPARISONS WITH OTHER EXISTING METHODS

This section compares the authentication performance of our proposed TSIA scheme with other recently published works, including Lin et al.'s [24], Wu et al.'s [25], Zhong et al.'s [26], Hong et al.'s [23], Lin et al.'s [22], Guo and Liu's [27], and Chen and Chang's [28] schemes. The block size used in all of the methods is set to 4x4. To make a fair comparison, we embedded 3-bit authentication information into each block for all of the compared and proposed methods.

As mentioned earlier, Hong et al.'s embedding strategy may break the relation between high and low quantization values, making the hidden authentication codes be damaged and increasing the probability of false detection. To prove this point, we show the authentication results for the watermarked AMBTC compressed image provided by Hong et al.'s scheme [23] and the proposed scheme, as shown in Fig. 18. Herein, Figs. 18(a) and (d) are the watermarked AMBTC compressed image decoded from watermarked AMBTC compression codes in Hong et al.'s scheme and our approach, respectively. Then, we separately perform the authentication process to those two watermarked AMBTC compressed images by using Hong et al.'s strategy and our approach. It is worth note that those two watermarked AMBTC compressed images have not encountered any attacks in this experiment. Figs. 18(b) and (c) demonstrate the detection results for first-stage and second-stage provided by Hong et al.'s scheme, where some black points were distributed over the TM, indicating that their scheme indeed exists the misjudgements even if attacks have not occurred. The failure was such because the relation between high and low quantization values of some AMBTC compression codes compressed from Fig. 18(a) are different from that of original authenticated versions. On the other hand, the proposed scheme always maintains the relation between high and low quantization values before and after authentication code embedding. That means that our approach solves the problems when \( H \) is less than or equal to \( L \) after authentication code embedding so that our proposed scheme can effectively authenticate the integrity for watermarked AMBTC compression codes, as shown in Figs. 18(e) and (f).
Table 3. Comparisons of FPR, FNR, and TDR for six attacks provided by Hong et al.’s scheme and proposed scheme

<table>
<thead>
<tr>
<th>Type of Attack</th>
<th>Hong et al.’s scheme</th>
<th>Proposed scheme</th>
<th>Hong et al.’s scheme</th>
<th>Proposed scheme</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Splash</td>
<td>Lena</td>
<td>Splash</td>
<td>Lena</td>
</tr>
<tr>
<td></td>
<td>FPR(%)</td>
<td>FNR(%)</td>
<td>TDR(%)</td>
<td>FPR(%)</td>
</tr>
<tr>
<td>Tamper with cropping</td>
<td>33.861</td>
<td>0</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>Tamper with a flower</td>
<td>61.2954</td>
<td>0.0317</td>
<td>98</td>
<td>0.4149</td>
</tr>
<tr>
<td>Tamper with H&amp;L</td>
<td>33.7698</td>
<td>0</td>
<td>100</td>
<td>0</td>
</tr>
<tr>
<td>Tamper with bitmap</td>
<td>33.7474</td>
<td>0.0067</td>
<td>99.8959</td>
<td>0</td>
</tr>
<tr>
<td>Tamper H&amp;L with a flower</td>
<td>24.0076</td>
<td>0.0338</td>
<td>99.5871</td>
<td>0.0834</td>
</tr>
<tr>
<td>Tamper bitmap with a flower</td>
<td>24.2366</td>
<td>0.1350</td>
<td>98.3485</td>
<td>0.0843</td>
</tr>
<tr>
<td>Average</td>
<td>35.1530</td>
<td>0.0345</td>
<td>99.3053</td>
<td>0.0971</td>
</tr>
</tbody>
</table>

FIGURE 18. Comparisons of the authentication results for watermarked AMBTC compressed image without other attacks. Hong et al.’s scheme: (a) Watermarked Lena image, (b) First-stage, (c) Second-stage. Proposed scheme: (d) Watermarked Lena image, (e) First-stage, (f) Second-stage.

To further verify the detectability of watermarked AMBTC compressed image, the comparisons of FPR, FNR, and TDR provided by Hong et al.’s scheme and proposed scheme under six attacks. The experiments were performed on smooth type images Splash and Lena, and the corresponding results are listed in Table 3. In Hong et al.’s scheme, the relation between quantization levels would be increasingly broken for smooth type images because there are more numbers of smooth blocks. As expected, we can observe from Table 3 that, for image Splash and Lena, the average FPRs under six attacks obtained from Hong et al.’s scheme are 35.1530% and 11.7331%, respectively, which is far higher than that of the proposed TSIA scheme. It means that the performance in tampering detection of the proposed TSIA scheme is more stable.

Table 4 provides a summary and comparisons of the proposed method and other recently published methods. Since Wu et al.’s method modifies a bitmap to embed authentication data, their method achieves lower image quality than the other methods. The compared methods [22, 24-26, 27] embed block-independent ACs generated by PRNG. Only ACs of Hong et al.’s [23] and our schemes are generated by MD5. Most of them failed to detect some intentional modifications of the bitmap or quantization levels. Herein, Guo and Liu’s scheme [27] embedded AC into bitmap by using the majority-parity-guided error diffusion algorithm and their scheme cannot resist this type of modifications, i.e., flipping any two bits of bitmap. Chen and Chang’s scheme [28] combined the bitmap into the operation of generating the AC, so their scheme can against the special attack on the bitmap. We also can observe that, in some cases, schemes’ [23-26, 28] embedding strategies broke the relation between quantization levels, resulting in these schemes failing to effectively work on the detectability of watermarked AMBTC compressed image. Moreover, our proposed TSIA scheme does not require the original image as a reference to embed AC, which facilitates the use of the proposed TSIA scheme with a...
wider range of applications. Based on the experimental results, the proposed TSIA scheme exceeds previous methods by providing an average PSNR of 32.60dB. Although the average PSNR of our proposed TSIA scheme is less than that of Lin et al.’s scheme [24]. However, Lin et al.’s scheme fails to detect the attacks which either only occurred on the bitmap or on the two quantization levels. Therefore, in general, it can be concluded that our proposed TSIA scheme successfully conquers existing problems occurred in the previous six schemes while maintaining acceptable image quality of the watermarked AMBTC-compressed images.

Table 4. Comparison with seven existing schemes

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Component to embed AC</td>
<td>Quantization levels</td>
<td>Bitmap</td>
<td>Quantization levels</td>
<td>Quantization levels</td>
<td>Quantization levels</td>
<td>Bitmap and quantization levels</td>
<td>Quantization levels</td>
<td>Bitmap</td>
</tr>
<tr>
<td>Requirement of the original image to embed AC</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Generation of AC</td>
<td>MD5</td>
<td>PRNG</td>
<td>PRNG</td>
<td>PRNG</td>
<td>MD5</td>
<td>PRNG</td>
<td>Bitmap and PRNG</td>
<td>PRNG</td>
</tr>
<tr>
<td>Detection of the special modification of bitmap</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Solving the problem of H=L and H=L</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Average PSNR</td>
<td>32.60dB (HVSPSNR: 45.28 dB)</td>
<td>34.11dB</td>
<td>29.66 dB</td>
<td>34.26 dB</td>
<td>32.44 dB</td>
<td>33.07 dB</td>
<td>32.99 dB (HVSPSNR: 40.10 dB)</td>
<td></td>
</tr>
</tbody>
</table>

V. CONCLUSIONS

This paper proposed a novel authentication code scheme called TSIA based on turtle shell to solve the problem where a modified H cannot be larger than the modified L after embedding the authentication code. While previous work by Hong et al. achieved an efficient authentication method for AMBTC-compressed images using adaptive pixel pair matching, their method does not solve the above-mentioned problem.

The results for the proposed scheme obtained almost the same embedding capacity as that of the Hong et al. method, but our authentication code scheme for the embedded image is twice as efficient as that of the Hong et al.’s method. In the future, we will try to search for another way, such as using the bitmap of smooth block adequately, to provide strong tampering detection performance while enhancing image quality, and we will pay more attention to innovate the AMBTC technique with the aim of improving image quality while maintaining a lower computation required.

REFERENCES


