A bit toggling approach for AMBTC tamper detection scheme with high image fidelity

Wien Hong¹, Dan Li¹, Der-Chyuan Lou²,³*, Xiaoyu Zhou¹, Chien-Hung Chang³,⁴

¹ School of Electrical and Computer Engineering, Nanfang College of Sun Yat-Sen University, Guangzhou, China, ² Department of Computer Science and Information Engineering, Chang Gung University, Taoyuan, Taiwan, ³ Stroke Center and Department of Neurology, Chang Gung Memorial Hospital, New Taipei, Taiwan, ⁴ College of Medicine, Chang Gung University, Taoyuan, Taiwan

* dclouprof@gmail.com

Abstract

The existing tamper detection schemes for absolute moment block truncation coding (AMBTC) compressed images are able to detect the tampering. However, the marked image qualities of these schemes can be enhanced, and their authentication methods may fail to detect some special tampering. We propose a secure AMBTC tamper detection scheme that preserves high image fidelity with excellent detectability. In the proposed approach, a bit in bitmaps of AMBTC codes is sequentially toggled to generate a set of authentication codes. The one that causes the least distortion is embedded into the quantization levels with the guidance of a key-generated reference table (RT). Without the correct key, the same reference table cannot be constructed. Therefore, the proposed method is able to detect various kinds of malicious tampering, including those special tampering techniques designed for RT-based authentication schemes. The proposed method not only offers better image quality, but also provides an excellent and satisfactory detectability as compared with previous works.

1. Introduction

With the rapid development of image acquisition devices, digital images are more popular than before, and their applications are ubiquitous. Images are the most seen digital media over the Internet; however, modern computing technologies make digital images to be easily modified maliciously. Therefore, authenticating a received image has become crucial for numerous applications. Fragile watermarking [1–2] is a commonly used technique to protect the integrity of digital images by embedding fragile watermarks (or authentication codes, ACs) into images. Since the embedded ACs can be easily destroyed when modifying the marked image, the presence of tampering is detected and tampered trios can be located.

Fragile watermarking techniques can be applied to images of spatial [3–4] or compressed domains [5–18]. Because most of the images are stored and transmitted in compressed formats, the investigation of compressed-domain authentication techniques has received extensive research attention. A number of compressed domain authentication techniques such as vector quantization [5–6], joint photographic experts group (JPEG) [7–8], and absolute
moment block truncation coding (AMBTC) [9–18] have been investigated. These techniques modify coefficients of compressed codes to embed authentication information, and the marked compressed codes can be decoded to generate an image that is similar to the original one. Since the AMBTC requires less computing cost but still offers acceptable compression ratio and image quality, it is applicable for applications demanding low-power consumption and limited transmission bandwidth such as devices for IoT. However, unprotected AMBTC codes are vulnerable to security threats such as malicious tampering. Therefore, protecting the transmitted AMBTC codes has become increasingly crucial for real applications. To protect the compressed codes from being tampered, several authentication techniques designed for AMBTC compressed images [9–18] have been investigated and gained wide attention due to the applicability of AMBTC.

The AMBTC compresses image blocks into trios. Each trio consists of two quantization levels and a bitmap. The quantization levels are used to record the intensities of image blocks while the bitmap is used to store the textures. The existing AMBTC authentication schemes embed ACs into either quantization levels or bitmaps. Hu et al. [11] propose a joint coding and authentication method for AMBTC codes. In their method, bitmaps are sub-divided, and ACs are embedded by matching the parity of sub-divided bitmaps. In Hu et al.’s another work [12], the length of ACs is designed to be adjustable for better embedding efficiency. Wu et al. [13] recognize that modifying the bitmap for AC embedment may cause larger distortions. Therefore, instead of bitmap, they embed ACs into quantization levels using a parity matching technique. Lin et al. [14] utilize the parity of bitmap to generate ACs and embed them into key-selected bits of quantization levels.

The early works [11–14] for authenticating AMBTC compressed codes mainly focus on the capability of detecting tampered trios. In addition to the detection performance, recent works [15–18] have paid more effort on improving the marked image quality by using reference table (RT) based embedding techniques [19–21] or adding the capability of recovery the tampered trios for some special applications [22]. Li et al. [15] present a RT-based AMBTC authentication scheme, in which two digits of base $\lambda$ are embedded into a compressed trio using a RT of size $\lambda \times \lambda$. Lin et al. [16] employ a hybrid scheme to embed ACs into smooth and complex blocks using different embedding strategies. Chen and Chang [17] notice that the bitmap in method [15] is unprotected, and present a new scheme such that the alteration of bitmap is detectable. Hong et al. [18] also propose a RT-based authentication scheme in which bitmaps are hashed to generate ACs. The ACs are then embedded into quantization levels using a RT-based technique [20].

While methods [15–18] remarkably protect the AMBTC compressed codes, some special designed tampering for the marked AMBTC trios may cause their authentication scheme failure. This is because the ACs generated from these methods are irrelevant to the codes they protect. As a result, the attackers may modify the marked trios in an intentional way to conceal the presence of tampering. Moreover, these methods all keep the bitmap intact when embedding the ACs. However, if quantization levels are employed to carry the ACs generated from the bitmap, toggling a bit in the bitmap might generate a different AC. The newly generated AC might lead to a smaller distortion than the one generated from the original bitmap. In light of this, this paper toggles bits in bitmaps to generate candidates of ACs. The best candidate that causes the least distortion is embedded into quantization levels using a key-protected RT-based embedding technique. Since the ACs are the hashed result of the toggled bitmap, any alteration of marked trios will extract incorrect ACs and thus the marked trios are fully protected.

The uniqueness of the proposed method is to present a method that perturbs bits of AMBTC’s bitmap to produce a set of selectable ACs. With the aids of the RT-based embedding
technique, the AC with the smallest embedding distortion is securely embedded into the quantization levels. To the best of our knowledge, the proposed bitmap toggling technique is the first work to exploit the bitmap perturbation in combination with the RT-based embedding techniques in the discipline of AMBTC authentication. The experimental results demonstrate that the proposed method not only successfully detects any kinds of tampering, but also gives the best marked image quality when compared with prior works.

The rest of this paper is organized as follows. Section 2 briefly describes related works. Section 3 presents the proposed method, while Section 4 gives the experimental results. Concluding remarks are given in the last section.

2. Related works

In this section, the AMBTC compression technique will be briefly introduced. A RT-based method, which will be used as the embedding technique in the proposed method, is presented in this section. We also briefly introduce Chen and Chang’s work [17] in the last sub-section.

2.1 The absolute moment block truncation coding

The AMBTC [23], a variant of block truncation coding [24], is an efficient lossy compression technique proposed by Lema and Mitchell in 1984. To compress the original image \(I\), the AMBTC partitions \(O\) into blocks \(\{O_i\}_{i=1}^N\) of size \(n \times n\), where \(N\) is the total number of blocks. Each block \(O_i\) is represented by a lower quantization level \(a_i\), a higher quantization level \(b_i\), and a bitmap \(B_i\). Let \(B_{ij}\) and \(O_{ij}\) be the \(j\)-th bit of \(B_i\) and \(j\)-th pixel of \(O_i\), respectively. To obtain \(\{B_{ij}\}_{j=1}^N\), the mean \(m_i\) of block \(O_i\) is calculated. If \(O_{ij} < m_i\), \(B_{ij} = 0\). Otherwise, \(B_{ij} = 1\). The two quantization levels \(a_i\) and \(b_i\) are calculated by rounding the mean value of \(O_{ij}\) with \(B_{ij} = 0\) and \(B_{ij} = 1\), respectively. Each block is compressed in the same manner, and the compressed trios \(\{a_i, b_i, B_i\}_{i=1}^N\) are then obtained. To decompress the \(i\)-th trio \((a_i, b_i, B_i)\), a block \(\{C_{ij}\}_{j=1}^N\) for storing decompressed results is initialized, where \(C_{ij}\) is the \(j\)-th pixel of block \(C_i\). Then, pixels in \(\{C_{ij}\}_{j=1}^N\) are sequentially visited. If \(B_{ij} = 0\), \(C_{ij} = a_i\) is set. Otherwise, \(C_{ij} = b_i\). Using the same procedure to decompress all trios, the decompressed AMBTC image \(\{C_i\}_{i=1}^N\) is then obtained.

Here is an example to show the AMBTC compression method. Suppose \(O_i = [81, 67, 96, 96; 70, 19, 96, 54; 22, 35, 25, 82; 92, 59, 97, 33]\) is the original block to be compressed, where semicolons represent a row separator. The mean \(m_i\) of \(O_i\) is 64. Therefore, we have \(B_i = [1111; 1010; 0001; 1010]\), \(a_i = 35\) and \(b_i = 86\). Once the receiver has the compressed trio \((a_i, b_i, B_i) = [35, 86, 1111; 1010; 0001; 1010]\), the decompressed block \(C_i = [86, 86, 86, 86; 86, 35, 86, 35; 35, 35, 86, 35; 86, 35, 86, 35]\) is obtained.

2.2 The reference table based embedding technique

The RT-based embedding technique employs a pixel pair as an embedding unit, and embeds a secret digit \(d_i\) of base \(\lambda\) by referencing a reference table \(RT_{\lambda}\). \(RT_{\lambda}\) is a matrix of size 256×256 filled with digits ranging from 0 to \(\lambda - 1\), and the element located in the \(a\)-th row and \(b\)-th column is denoted by \(RT_{\lambda}(a, b)\). To embed \(d_i\in[0, \lambda - 1]\) into a pixel pair \((a, b)\), the neighboring elements of \(RT_{\lambda}(a, b)\) are searched. The coordinate \((a', b')\) satisfying \(RT_{\lambda}(a', b') = d_i\) and having the nearest distance to \((a, b)\) is selected as the marked pixel pair. The embedded digit \(d_i\) in coordinate \((a', b')\) can be easily extracted by locating the element in \((a', b')\) of \(RT_{\lambda}\), i.e., \(d_i = RT_{\lambda}(a', b')\).

Fig 1 gives the illustration of RT-based embedding technique.

The design of reference table greatly affects the embedding performance. Several RT-based embedding methods [19–21] have been proposed during the past decade. Among these
Fig 1. The illustration of RT-based embedding technique.

https://doi.org/10.1371/journal.pone.0230997.g001

Fig 2. An example of the APPM method.

https://doi.org/10.1371/journal.pone.0230997.g002
methods, the adaptive pixel pair matching (APPM) [20] achieves the lowest embedding distortion. The RT used in APPM is generated by using the following function

\[ RT_{\lambda}(a, b) = (c_{\lambda} \times a + b) \mod \lambda, \]  

where \( c_{\lambda} \) is a \( \lambda \)-dependent coefficient, \((a, b) \in \text{Integers}, 0 \leq a \leq 255 \) and \( 0 \leq b \leq 255 \). The often-used coefficients are \( c_4 = 2, c_8 = 3, c_{16} = 6, c_{32} = 7, c_{64} = 14 \) and \( c_{256} = 60 \). The full list of \( c_{\lambda} \) can be seen in [20].

An example is given to show the APPM method as follows. Suppose the secret digit \( d_{16} = 15 \) of base 16 is to be embedded into the pixel pair \((a, b) = (45,85)\). Fig 2 shows a portion of \( RT_{16} \), which can be generated from Eq (1). Since the element located in \((43, 84)\) is 15, and the distance between \((43, 84)\) and \((45, 85)\) is the shortest, we obtain the marked pixel pair \((a_0, b_0) = (43, 84)\). Given the reference table \( RT_{16} \) and marked pixel pair \((43, 84)\), the embedded digit \( RT_{16}(43, 84) = 15 \) can be extracted.

2.3 Chen and Chang’s scheme
In 2018, Chen and Chang [17] analyzed the weakness of Li et al.’s method [15] and proposed a secure authentication scheme for AMBTC compressed images. Li et al.’s scheme generates authentication digits using a random number generator, and embeds the generated digits into the quantization levels of the AMBTC codes by referencing a pre-designed reference matrix. Li et al.’s method offers a high visual quality of marked AMBTC image and is capable of detecting the alteration of quantization levels. However, the protection of bitmap is ignored in their method. In light of this, Chen and Chang proposes an improved version of Li et al.’s method by removing the aforementioned security concerns.

In Chen and Chang’s scheme [17], instead of using the random number generator, the authentication code \( ac_i \) is generated by calculating \( ac_i = (rv \oplus B_i) \mod 2^\omega \), where \( rv \) is a random number, \( \oplus \) denotes the exclusive-OR operator, and \( \omega \) is the number of bits to be embedded into the \( i \)-th pair of the quantization level. Since the bitmap is used to generate the ACs, the alteration of bitmap would extract incorrect ACs. Therefore, the presence of tampering will be detected. The embedment and authentication procedures are similar to those of Li et al.’s method. See [17] for more details.

3. The proposed method
The existing RT-based AMBTC authentication schemes reduce image distortions to a large extent. However, the embedded ACs can be easily extracted from the marked trios once the used RT is known. As a result, one can tamper marked trios with some easy modifications to evade the detection by these methods. For example, [18] uses the APPM to embed ACs. Suppose \((a', b')\) is a pair of marked quantization levels with the authentication code \( d_i \) embedded. The embedded authentication code can be simply extracted by calculating \( d_i = (c_{\lambda} \times a' + b') \mod \lambda \) (see Eq (1)). However, \((k\lambda + (c_{\lambda} \times a' + b')) \mod \lambda \) also gives the same \( d_i \) for any integer \( k \). As a result, one can tamper the marked image without being detected by [18] if adding \( a_i' \) and \( b_i' \) by \( k\lambda \). The same problem also exists in methods [16] and [17]. Besides, the existing methods do not fully exploit the characteristic of bitmaps. In fact, since the ACs are generated from bitmaps, a slight modification of bitmaps could possibly introduces a smaller distortion after embedding the newly generated ACs. However, none of existing RT-based authentication schemes take the advantage of this property to reduce the distortion of marked images.

In this paper, we hash bitmaps and other information to generate ACs. A key-generated RT is used to further secure the authentication method. Since the same RT can only be reconstructed using the correct key, the tampering using aforementioned techniques can be easily...
detected. Moreover, a bit in the bitmap is toggled sequentially to generate a set of toggled bit-
maps. The one that causes the least distortion after embedment will be selected as the marked
bitmap. Therefore, the distortion is guaranteed to be smaller than or equal to the one without
using this approach.

3.1 Generation of key-selected reference table
To prevent the leakage of embedded authentication codes, as has been discussed at the begin-
ing of this section, we use a key $\kappa$ to generate key-specified reference tables $\{RT_i^{(\psi)}\}_{i=1}^N$. The element in the $a_i$-th row and $b_i$-th column of $RT_i^{(\psi)}$ is generate by

$$RT_i^{(\psi)}(a_i, b_i) = (RT_{\lambda}(a_i, b_i) + \sigma_i) \mod \lambda,$$

(2)

where $RT_{\lambda}(a_i, b_i) = (c_i \times a_i + b_i) \mod \lambda$, and $\sigma_i$ is a random integer generated by the key $\kappa$. Eq (2) successfully shifts the RT to the left circularly by $\sigma_i \mod \lambda$ units. The reference table $RT_i^{(\psi)}$ will be used in the embedment of the $i$-th trio. The embedded ACs can only be successfully extracted with the correct key.

3.2 Generation and embedment of ACs
Given the $i$-th AMBTC compressed trio $(a_i, b_i, B_i)$, we firstly generate a reference table $RT_i^{(\psi)}$, as described in Section 3.1. Since $RT_i^{(\psi)}$ is filled with digits ranging from 0 to $\lambda - 1$, $w = \lfloor \log_2 \lambda \rfloor$-bit information can be embedded into a trio. To generate a $w$-bit authentication code $ac_i^w$, we hash the bitmap $B_i$, position information $i$, and the image identification $I_d$ using the MD5 [25] hash function, and fold the hashed result into $w$ bits using the exclusive-or operation. The authentication code $ac_i^w$ is then embedded into $(a_i, b_i)$ to obtain the marked quantization level pair $(a_i', b_i')$ using the APPM method under the guidance of $RT_i^{(\psi)}$. The embedment of $ac_i^w$ can be formulated as

Minimize : $(a_i - a_i')^2 + (b_i - b_i')^2$  \hspace{1cm} (3)

Subject to : $ac_i^w = \text{hash}_w(B_i, i, I_d)$, \hspace{1cm} (4)

$$RT_i^{(\psi)}(a_i', b_i') = ac_i^w,$$  \hspace{1cm} (5)

where $\text{hash}_w(x)$ returns the decimal value of $w$-bit hashed result of $x$. Once Eqs (3)–(5) have been solved, the marked AMBTC trio $(a_i', b_i', B_i)$ can then be obtained. We continue the example given in Section 2.2 to illustrate the embedment of $ac_i^w$. Suppose the random generated integer is $\sigma_i = 17763$ and a portion of $RT_i^{(\psi)}$ is given in Fig 3. Let $a_i = 43$, $b_i = 87$ and $ac_i^w = 12$. Because $RT_i^{(\psi)}(44,88) = 12$ and $(44,88)$ is the closest to $(43,87)$, we have $(a_i', b_i') = (44, 88)$.

3.3 Bit toggling technique
With the marked AMBTC trio $(a_i', b_i', B_i)$, the corresponding decompressed image block $\psi(a_i', b_i', B_i)$ can be obtained. When compared with the original AMBTC block $\psi(a_i, b_i, B_i)$, the distortion occurs if $a_i \neq a_i'$ or $b_i \neq b_i'$. However, if the $k$-th bit of $B_i$ is toggled, the AC generated from the toggled bitmap $B_i'$ is likely to be very different from the one obtained from the un-
toggled counterpart. After embedment, the decompressed block $\psi(a_i', b_i', B_i')$ is possible to have a larger or smaller distortion than that of $\psi(a_i, b_i', B_i)$ because the $k$-th bit of $B_i$ is toggled.
Therefore, bits in $B_i$ can be toggled sequentially to find a toggled bitmap that causes the least distortion.

The bit toggling technique (BTT) is described as follows. Let $B_i^k$ be the bitmap $B_i$ with $k$-th bit toggled and $B'_0 = B_i$, $0 \leq k \leq n \times n$. A total of $n \times n + 1$ authentication codes $ac_i^{w,k}$ = hash$_w(B_i^k, i, I_d)$ can be generated. When embedding $\{ac_i^{w,k}\}_{k=0}^{n \times n}$ into $(a_i, b_i)$ using APPM, a set of embedded regions $\{a_i^{r,k}, b_i^{r,k}, B'_i\}_{k=0}^{n \times n}$ is obtained, and the mean square error between $\psi(a_i, b_i, B_i)$ and $\psi(a_i^{r,k}, b_i^{r,k}, B'_i)$ can be evaluated. The embedded trio that makes the decoded block having the least distortion is selected as the marked trio $(a_i^{r*}, b_i^{r*}, B'_i)$. The solution to the
In this section, we conduct several experiments to demonstrate the effectiveness and applicability of the proposed method. We use eight grayscale images of size 512×512 shown in Fig 6 as the test images, and compress them to obtain the AMBTC trios. The AC embedding is then applied to obtain marked trios. The eight test images can be found in the USC-SIPI image database [26]. We also compare the performance of the proposed method with methods [15–18]. In all experiments, a block size of 4×4 is set.

We use the peak signal-to-noise ratio (PSNR) to measure the marked image quality. A higher PSNR indicates that the quality of marked image is more close to its unmarked version.
In this paper, the PSNR measurement is defined by

\[
\text{PSNR} = 10 \log_{10} \left( \frac{255^2}{\sum_{i=1}^{N} (x_i - \hat{x}_i)^2} \right),
\]

(10)

Fig 5. The schematic diagram of authentication procedures.

https://doi.org/10.1371/journal.pone.0230997.g005

Fig 6. Eight test images of size 512×512. (a) Tiffany. (b) Splash. (c) Jet. (d) Airplane. (e) Peppers. (f) Boat. (g) Baboon. (h) House.

https://doi.org/10.1371/journal.pone.0230997.g006
where $x_i$ and $x'_i$ represent the pixel values of images decompressed from the original and marked AMBTC trios, respectively.

### 4.1 Performance of the proposed method

It has been pointed out in Section 3 that for each trio, the distortion caused by using the BTT is always equal to or smaller than that of without using this technique. Fig 7(A)–7(D) demonstrate the effect of using BTT for $\lambda = 4, 16, 64$ and 256, respectively. In this figure, the black dots represent that the distortions of corresponding image blocks are reduced when the BTT is applied.

As seen from Fig 7, the number of black dots increases as $\lambda$ increases. It means that the effect of using BTT becomes more apparent when more bits of ACs are embedded. The reason is that embedding a digit of larger base may cause more distortion, and toggling a bit in the bit-map has a greater chance to generate an AC that causes a less distortion. It is interesting to note that black dots distribute densely in smooth regions (e.g., Tiffany’s face and backgrounds) but distribute sparsely in complex regions (e.g., Tiffany’s hair). This is because toggling a bit in complex regions leads to more distortion than that of smooth ones. Therefore, the BTT is more effective when it is applied in smooth regions.

Fig 8 shows the visual quality comparisons of the enlarged marked Tiffany image when using and without using the BTT. In this experiment, we set $\lambda = 256$, which is equivalent to embedding 8-bit ACs into each trio. As seen from Fig 8(A) where the BTT is not applied, apparent noises distribute in smooth parts of the image such as Tiffany’s face and backgrounds. However, with the application of BTT, no apparent noises are found, demonstrating the effectiveness in enhancing the image quality.

The PSNR comparisons of using and without using the BTT with different embedding bases are list in Table 1. The results show that the gain of PSNR is more significant for a larger $\lambda$, conforming to previous discussions. It is worth to note that the gain of PSNR for $\lambda = 64$ and $\lambda = 256$ of the Airplane image is the highest because smooth regions of this image are more than others. On the other hand, the Baboon image has the smallest PSNR improvement because it possesses the richest textures.

Fig 9 shows the performance of coarse and refined detections of the proposed method when setting $\lambda = 4, 16$ and 64. We tamper the Tiffany image by splicing a rose on Tiffany’s hat (Fig 9(A)), and the tampered region is shown in Fig 9(B). Fig 9(C)–9(H) give the detection results for different $\lambda$, where the black dots represent that the corresponding blocks are reported as tampered ones.

The experiments show that for all coarse detections, some white spots appear in the tampered regions due to hash collisions. In fact, the hash collision rate of the coarse detection is $1/2^4$. Therefore, the number of white spots decreases as $\lambda$ increases. Nevertheless, with the application of refined detection, white spots are successfully eliminated. As seen in Fig 9(F) and 9(H), the refined detections perform equally well when setting $\lambda = 16$ and $\lambda = 64$ because no white spots exist for both settings. However, setting $\lambda = 64$ causes larger distortions than those of $\lambda = 16$. Therefore, we recommend setting $\lambda = 16$ to achieve a balance between image quality and detectability.

### 4.2 Image quality comparisons with other works

In this section, we compare the proposed method with other related works, including Hong et al.’s [18], Chen et al.’s [17], Lin et al.’s [16], and Li et al.’s [15] methods. To make a fair comparisons, the Hong et al.’s method [18] is implemented such that smooth and complex blocks carry an equal number of authentication bits. In [17] and [15], a reference table of size $4 \times 4$ is
used so that each trio is capable of carrying ACs of 2 or 4 bits. In [16], we set the threshold equal to 7 to achieve the best image quality, as suggested in the original paper. Since the trios in methods [15–17] cannot carry ACs larger than 4 bits, we only compare the marked image quality when 2-bit and 4-bit ACs are embedded. The results are shown in Table 2.
As shown in the table, the image qualities of all test images of [16] are the lowest when 2-bit ACs are embedded. The reason is that the RT they used is a Sudoku table, which introduces more distortions than the one constructed using Eq (1). Methods [15] and [17] offer almost identical image qualities because both methods adopt the same embedding technique. However, Chen and Chang’s method [17] is more secure than that of Li et al.’s method [15] because the bitmaps in [17] are fully protected. The Hong et al.’s method [18] provides better PSNR than that of [15–17]; however, their method cannot detect some types of special tampering, as has been described in Section 3. Obviously, the proposed method offers the best image quality for all test images due to the proposed BTT. Moreover, the length of AC of the proposed method is adjustable and can be extent to embed 3- or 4-bit ACs per trio, while methods [15–

---

**Table 1. PSNR comparisons of using and without using the BTT (in dB).**

| Base | Method | Tiffany | Splash | Jet | Airplane | Peppers | Boat | Baboon | House |
|------|--------|---------|--------|-----|----------|---------|------|--------|-------|
| λ = 4 | BTT    | 52.45   | 52.59  | 52.84| 52.41     | 52.28   | 52.33| 52.36  | 52.72 |
|      | w/o BTT| 52.23   | 52.24  | 52.17| 52.12     | 52.25   | 52.29| 52.36  | 52.14 |
|      | Gain   | 0.22    | 0.35   | 0.67 | 0.29      | 0.03    | 0.04 | 0.00   | 0.58  |
| λ = 16 | BTT    | 47.92   | 48.19  | 48.35| 49.07     | 47.32   | 47.17| 46.88  | 47.69 |
|      | w/o BTT| 46.86   | 46.87  | 46.87| 46.87     | 46.86   | 46.89| 46.86  | 46.79 |
|      | Gain   | 1.06    | 1.32   | 1.48 | 2.20      | 0.46    | 0.28 | 0.02   | 0.90  |
| λ = 64 | BTT    | 43.48   | 44.43  | 43.71| 45.75     | 42.97   | 42.15| 41.23  | 42.45 |
|      | w/o BTT| 40.91   | 40.97  | 40.94| 40.95     | 40.96   | 40.94| 40.93  | 40.84 |
|      | Gain   | 2.57    | 3.46   | 2.77 | 4.80      | 2.01    | 1.21 | 0.30   | 1.61  |
| λ = 256 | BTT    | 39.42   | 40.65  | 39.07| 41.64     | 39.11   | 37.90| 36.04  | 37.47 |
|      | w/o BTT| 34.89   | 34.97  | 35.03| 35.02     | 35.04   | 35.04| 35.01  | 34.94 |
|      | Gain   | 4.53    | 5.68   | 4.04 | 6.62      | 4.07    | 2.86 | 1.03   | 2.53  |

---

https://doi.org/10.1371/journal.pone.0230997.t001
Notice that the proposed method provides significantly better image quality than that of [18], especially for larger length of AC. For example, when the lengths of AC are 6 ($\lambda = 64$) and 8 ($\lambda = 256$), the improvements in PSNR of the Tiffany image are $43.48 - 40.92 = 2.56$ dB and $39.42 - 34.93 = 4.49$ dB, respectively. Besides, the proposed method is more secure in protecting the AMBTC trios than that of [18], as will be presented in the next subsection.

Table 2. PSNR comparisons for various lengths of AC.

| AC  | Method | Tiffany | Splash | Jet | Airplane | Peppers | Boat | Baboon | House |
|-----|--------|---------|--------|-----|----------|---------|------|--------|-------|
|     | Proposed | 52.45   | 52.59  | 52.84| 52.41     | 52.28   | 52.33| 52.36 | 52.72 |
|     | [18]     | 52.18   | 52.26  | 52.12| 52.09     | 52.25   | 52.28| 52.39 | 52.18 |
|     | [17]     | 51.09   | 51.02  | 51.14| 51.17     | 51.24   | 51.10| 51.07 | 51.14 |
|     | [16]     | 48.33   | 47.74  | 49.29| 48.35     | 47.41   | 48.30| 50.17 | 49.84 |
|     | [15]     | 48.12   | 51.07  | 51.16| 51.10     | 51.13   | 51.13| 51.14 | 51.17 |
|     | Proposed | 46.92   | 46.86  | 46.79| 49.07     | 46.79   | 46.88| 46.87 | 46.87 |
|     | [18]     | 46.78   | 46.89  | 46.90| 46.89     | 46.89   | 46.88| 46.84 | 46.74 |
|     | [17]     | 44.35   | 44.36  | 44.28| 44.24     | 44.39   | 44.36| 44.38 | 44.33 |
|     | [16]     | 44.58   | 44.29  | 45.41| 44.63     | 43.71   | 44.22| 45.28 | 45.62 |
|     | [15]     | 44.43   | 44.48  | 44.38| 44.59     | 44.36   | 44.23| 44.28 | 44.30 |
|     | Proposed | 43.48   | 44.43  | 43.71| 42.97     | 42.97   | 42.15| 41.23 | 42.45 |
|     | [18]     | 40.92   | 40.93  | 40.95| 40.95     | 40.93   | 40.96| 40.96 | 40.93 |
|     | Proposed | 39.42   | 40.65  | 39.07| 41.64     | 39.11   | 37.90| 36.04 | 37.47 |
|     | [18]     | 34.93   | 35.04  | 34.92| 35.05     | 35.00   | 35.02| 35.00 | 34.97 |

https://doi.org/10.1371/journal.pone.0230997.t002
4.3 Detectability comparisons with other works

This section demonstrates the detectability comparisons of Hong et al.’s [18], Chen et al.’s [17], Lin et al.’s [16], and the proposed methods. In this experiment, the marked trios of Peppers image are tampered so that the tampered decompressed image exhibits an apple and bananas on the Peppers image, as seen in Fig 10(A). Fig 10(B) shows the tampered regions, where tampered blocks are represented by black color. To make a fair comparison, all methods are implemented to embed 4-bit AC into each trio (λ = 16). When implementing the Lin et al.’s method [16], the threshold used to distinguish smooth and complex blocks is set to 7, as suggested in their original paper.

To splice the Apple image onto the marked Peppers image, the original Apple image $I^A$ is partitioned into blocks $\{I^A_i\}_{i=1}^N$, and temporarily placed onto the Peppers image, where $r$ is the total number of blocks of $I^A$. Each block $I^A_i$ is then compared with all the blocks of the marked Peppers image $\{I^P_i\}_{i=1}^N$. The block $I^P_i$ in $\{I^P_i\}_{i=1}^N$ that is the most similar to $I^A_i$ is then used to replace $I^A_i$. We denote this type of tampering as Type 1 tampering. To tamper the marked Peppers image by the Apple image $I^A$, the blocks $\{I^P_i\}_{i=1}^N$ of $I^A$ are temporarily replaced blocks $\{I^P_i\}_{i=1}^N$ of the Peppers image, where $\varphi$ is the total number of blocks of $I^A$. Let $(a^A_i, b^A_i, b^B_i)_{i=1}^\varphi$ and $(a^B_i, b^B_i, b^B_i)_{i=1}^\varphi$ be the trios of $(I^P_i)_{i=1}^\varphi$ and $(I^P_i)_{i=1}^\varphi$, respectively. The quantization levels $(a^A_i, b^A_i)$ are then adjusted to find integers $k_{1,i}$ and $k_{2,i}$ such that $(16k_{1,i} + a^A_i - a^B_i)^2 + (16k_{2,i} + b^A_i - b^B_i)^2$ is the smallest. In this way, the final spliced Bananas image is very similar to its original image but identical ACs can be extracted from $(a^B_i, b^B_i)$ and $(16k_{1,i} + a^B_i, 16k_{2,i} + b^B_i)$ using the RT-based embedding method. We denote this type of tampering as Type 2 tampering.

The detection results are shown in Fig 11(A)–11(D), where the red circles represent that the corresponding blocks are reported as tampered ones. As shown in Fig 11(A), both the spliced Apple and Bananas images cannot be detected using the Hong et al.’s method [18]. The reason is that their method does not utilize the location information of trios to generate ACs. Therefore, the Apple image composed by the most similar blocks of marked Peppers image cannot be detected.

The spliced Bananas image also cannot be detected by the Hong et al.’s method [18] because the trios $\{16k_{1,i} + a^B_i, 16k_{2,i} + b^B_i, b^B_i\}_{i=1}^\varphi$ of the spliced Bananas image always satisfy the equality $(c_{16} \times a^B_i + b^B_i) \mod 16 = (c_{16} \times (16k_{1,i} + a^B_i) + 16k_{2,i} + b^B_i) \mod 16$. Therefore, the same ACs can be extracted from the marked and tampered images, leading to a failure in detecting the manipulated image. Similar problems also happen in [17], where the spliced Bananas image cannot be detected. Notice that the spliced Bananas image can be partially detected by the Lin et al.’s method [16]. In fact, the ACs of [16] of the detected tampered blocks are embedded in bitmaps because these blocks are classified as complex ones. For those smooth blocks where the ACs are embedded in quantization levels, the presence of tampering cannot be detected because adding or subtracting the marked quantization levels by multiples of 16 will extract the same authentication codes. Nevertheless, the proposed method successfully detects both spliced Apple and Bananas images (Fig 11(D)), indicating that the proposed method is capable of detecting both types of malicious tampering.

To further validate the detection capability of the proposed method, we apply Type 1 and Type 2 tampering approaches to tamper the eight test images and compare the results with those of [18], [16] and [17]. The tampered images are shown in Fig 12. In these experiments, we splice a daisy, a faucet, an upper jet, an upper airplane, an apple, a seashell, a glasses, and a tree onto Fig 12(A)–12(H), respectively using the Type 1 tampering approach. Moreover, we
also splice a rose, a cup, a lower jet, a lower airplane, bananas, a snail, a mask, and a fence onto Fig 12(A)–12(H), respectively using Type 2 tampering approach.

We apply the proposed method and other related works [16–18] to eight test images to show the influence of image smoothness on the detectability. In these test images, Tiffany, Splash, Jet and Airplane are smoother than Peppers, Boat, Baboon and House. Fig 13 shows the detection results of smooth images whereas Fig 14 gives the results of complex ones. The red dots represent that the corresponding blocks are reported as tampered.

When authenticating the smooth images, the Hong et al.’s method [18] neither detects Type 1 nor Type 2 tampering. Lin et al.’s method [16] and Chen et al.’s method [17], are able to detect Type 1 tampering; however, they fail or partially fail to detect the tampering of Type 2. In contrast, both types can be detected by the proposed method (See Fig 13(A)–13(P)).

The detection results of complex images show the similar trends, as can be seen in Fig 14(A)–14(P). In fact, because all the compared methods embed equally amount of data into both smooth and complex blocks, the tamper detection rate is only relevant to the number of embedded bits, regardless of the image complexity. Therefore, the experiments on the smooth and complex images reveal the similar results. Notice that Lin et al.’s method [16] still cannot fully detect the Type 2 tampering. Since their method uses a threshold to control where should the authentication code be embedded, the setting of the threshold may affect the detection result. In this experiment, the threshold is also set to 7. If the threshold is set to 255, the tampered regions should be totally undetectable. However, this setting will significantly degrade the marked image quality. Nevertheless, the proposed method effectively detects both tampering types of the complex images.

In light of the experimental results shown above, the proposed method offers a better detectability and marked image quality, meaning that proposed method is more suitable for authenticating the AMBTC compressed images.
5. Conclusions

In this paper, a new AMBTC authentication method with high image quality and efficient detectability is proposed. To improve the marked image quality, a bit in the bitmap that causes
the least distortion after embedment is selected and toggled to generate the ACs. The introduced embedding distortion of the bit toggling technique is guaranteed to be smaller than or equal to that of Hong et al.’s method. To enhance the security, a key-specified reference table is used for guiding the embedment of ACs. Prior to the extraction of ACs, the same reference table that used in the embedding phase has to be constructed. Therefore, without the correct key, the embedded ACs cannot be extracted and attackers would be impossible to design undetectable tampered trios. The experiments show that the marked image quality of the proposed method is the highest when compared with prior related works. Moreover, the proposed method is able to detect all kinds of tampering while other methods can only detect some of them.

Fig 12. Eight tampered test images of size $512 \times 512$. (a) Tiffany. (b) Splash. (c) Jet. (d) Airplane. (e) Peppers. (f) Boat. (g) Baboon. (h) House.

https://doi.org/10.1371/journal.pone.0230997.g012

Fig 13. Detection results of [18], [16], [17] and the proposed method for smooth images. (a)-(d) Tiffany image. (e)-(h) Splash image. (i)-(l) Jet image. (m)-(p) Airplane image.

https://doi.org/10.1371/journal.pone.0230997.g013
Author Contributions

Conceptualization: Wien Hong.

Data curation: Wien Hong.

Formal analysis: Wien Hong, Dan Li, Xiaoyu Zhou.

Funding acquisition: Der-Chyuan Lou.

Investigation: Dan Li.

Methodology: Wien Hong.

Resources: Der-Chyuan Lou.

Software: Der-Chyuan Lou, Chien-Hung Chang.

Validation: Dan Li.

Visualization: Xiaoyu Zhou, Chien-Hung Chang.

Writing – original draft: Wien Hong.

References

1. Bravo-Solório S., Calderon F., Li C. T., and Nandi A. K., “Fast fragile watermark embedding and iterative mechanism with high self-restoration performance,” Digital signal processing, vol. 73, pp. 83–92, 2018.

2. Qin C., Ji P., Zhang X., Dong J., and Wang J., “Fragile image watermarking with pixel-wise recovery based on overlapping embedding strategy,” Signal processing, vol. 138, pp. 280–293, 2017.

3. Qin C., Wang H., Zhang X., and Sun X., “Self-embedding fragile watermarking based on reference-data interleaving and adaptive selection of embedding mode,” Information sciences, vol. 370, pp. 233–250, 2016.

4. Lin C. H., Chen T. H., and Chiu C. W., “Color image authentication with tamper detection and remedy based on BCH and Bayer pattern,” displays, vol. 34, no. 1, pp. 59–68, 2013.

5. Tiwari A. and Sharma M., “Novel watermarking scheme for image authentication using vector quantization approach,” Radioelectronics and communications systems, vol. 60, no. 4, pp. 161–172, 2017.

6. Qin C., Ji P., Wang J., and Chang C. C., “Fragile image watermarking scheme based on VQ index sharing and self-embedding,” Multimedia tools and applications, vol. 76, no. 2, pp. 2267–2287, 2017.

7. Chen J., Chen T. S., and Cheng C. Y., “New image tampering detection and recovery system of JPEG2000 region of interest area,” The imaging science journal, vol. 53, no. 1, pp. 12–19, 2005.

8. Wattanachote K., Shih T. K., Chang W., and Chang H., “Tamper detection of JPEG image due to seam modifications,” IEEE transactions on information forensics and security, vol. 10, no. 12, pp. 2477–2491, 2015.
9. Kim C., Shin D., and Yang C. N., “Self-embedding fragile watermarking scheme to restoration of a tampered image using AMBTC,” Personal and ubiquitous computing, vol. 22, no. 1, pp. 11–22, 2018.
10. Nguyen T. S., Chang C. C., and Chung T. F., “A tamper-detection scheme for BTC-compressed images with high-quality images,” KSII transactions on internet and information systems, vol. 8, no. 6, pp. 2005–2021, 2014.
11. Hu Y. C., Lo C. C., L Chen W., and Wen C. H., “Joint image coding and image authentication based on absolute moment block truncation coding,” Journal of electronic imaging, vol. 22, no. 1, pp. 013012, 2013.
12. Hu Y. C., Lo C. C., Wu C. M., Chen W. L., and Wen C. H., “Probability-based tamper detection scheme for BTC-compressed images based on quantization levels modification,” International journal of information & computer security, vol. 7, no. 3, pp. 11–32, 2013.
13. Wu C. M., Hu Y. C., Liu K. Y., and Chuang J. C., “A novel active image authentication scheme for block truncation coding,” International journal of signal processing, image processing & pattern recognition, vol. 7, no. 5, pp. 13–26, 2014.
14. Lin C. C., Huang Y., and L Tai W., “A high-quality image authentication scheme for AMBTC-compressed images,” KSII transactions on internet & information systems, vol. 8, no. 12, pp. 4588–4603, 2014.
15. Li W., Lin C. C., and Pan J. S., “Novel image authentication scheme with fine image quality for BTC-based compressed images,” Multimedia tools and applications, vol. 75, no. 8, pp. 4771–4793, 2016.
16. Lin C. C., Huang Y., and Tai W. L., “A novel hybrid image authentication scheme based on absolute moment block truncation coding,” Multimedia tools and applications, vol. 76, pp. 463–488, 2017.
17. Chen T. H. and Chang T. C., “On the security of a BTC-based-compression image authentication scheme,” Multimedia tools and applications, vol. 77, no. 10, pp. 12979–12989, 2018.
18. Hong W., Chen M. J., Chen T. S., and Huang C. C., “An efficient authentication method for AMBTC compressed images using adaptive pixel pair matching,” Multimedia tools and applications, vol. 77, no. 4, pp. 4677–4695, 2018.
19. Chen J., Shiu C. W., and Wu M. C., “An improvement of diamond encoding using characteristic value positioning and modulus function,” Journal of systems and software, vol. 86, no. 5, pp. 1377–1383, 2013.
20. Hong W. and Chen T. S., “A novel data embedding method using adaptive pixel pair matching,” IEEE transactions on information forensics and security, vol. 7, no. 1, pp. 176–184, 2012.
21. Hong W., “Adaptive image data hiding in edges using patched reference table and pair-wise embedding technique,” Information sciences, 201, pp. 473–489, 2013.
22. Hong W., Zhou X., and C Lou D., “A recoverable AMBTC authentication scheme using similarity embedding strategy,” PLOS ONE, vol. 14, no. 2, e0212802, 2019. https://doi.org/10.1371/journal.pone.0212802 PMID: 30811472
23. Lema M. and Mitchell O., “Absolute moment block truncation coding and its application to color image,” IEEE transactions on information forensics and security, vol. 10, no. 3, pp. 507–518, 2015.
24. Delp E. and Mitchell O., “Image compression using block truncation coding,” IEEE transactions on communications, vol. 27, no. 9, pp. 1336–1342, 1979.
25. Menezes A. J., Van Oorschot P. C., and Vanstone S. A., “Handbook of applied cryptography,” CRC press, 1996.
26. The USC-SIPI image database. Available online: http://sipi.usc.edu/database/