Article

Pixel P Air-Wise Fragile Image Watermarking Based on HC-Based Absolute Moment Block Truncation Coding

Chia-Chen Lin ^{1,*} , Si-Liang He ² and Chin-Chen Chang ^{2,3}

¹ Department of Computer Science and Information Engineering, National of Chin-Yi University of Technology, Taichung 41170, Taiwan

² Department of Information Engineering and Computer Science, Feng Chia University, Taichung 40724, Taiwan; ruby3896976@gmail.com (S.-L.H.); ccc@o365.fcu.edu.tw or dio_he@wistron.com (C.-C.C.)

³ School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310018, China

* Correspondence: mhl3@pu.edu.tw or ally.cclin@ncut.edu.tw

Abstract: In this paper, we first designed Huffman code (HC)-based absolute moment block truncation coding (AMBTC). Then, we applied Huffman code (HC)-based absolute moment block truncation coding (AMBTC) to design a pixel pair-wise fragile image watermarking method. Pixel pair-wise tampering detection and content recovery mechanisms were collaboratively applied in the proposed scheme to enhance readability even when images have been tampered with. Representative features are derived from our proposed HC-based AMBTC compression codes of the original image, and then serve as authentication code and recovery information at the same time during tamper detection and recovery operations. Recovery information is embedded into two LSB of the original image with a turtle shell-based data hiding method and a pre-determined matrix. Therefore, each non-overlapping pixel-pair carries four bits of recovery information. When the recipient suspects that the received image may have been tampered with, the compressed image can be used to locate tampered pixels, and then the recovery information can be used to restore the tampered pixels.

Keywords: image authentication; Huffman coding; absolute moment block truncation coding; fragile watermarking; pixel pair-wise; tamper detection

Citation: Lin, C.-C.; He, S.-L.; Chang, C.-C. Pixel P Air-Wise Fragile Image Watermarking Based on HC-Based Absolute Moment Block Truncation Coding. *Electronics* **2021**, *10*, 690. <https://doi.org/10.3390/electronics10060690>

Academic Editor: Cheonshik Kim

Received: 1 January 2021

Accepted: 24 February 2021

Published: 15 March 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The advancement of image editing software, such as Fotor, Gimp, Painter, Photoshop, etc., allows users easily to edit digital images. It also means, unacceptable consequences of digital images, including falsification and disinformation, could be easily created by malicious attackers. Under such circumstances, the integrity and ownership of digital images are critical issues, and these criteria are essential for determining the usability of received images.

To solve such problems, fragile watermarking has been proposed [1–25]. Additionally, traditional cryptographic schemes have been applied [26–28]. Fragile watermarking is a method that exposes changes in the image content by identifying variations in the information of embedded watermark bits. Early approaches to fragile watermarking only focused on locating tampered areas in a watermarked image. Later, in order to retain the usability of damaged images, some schemes were designed with recovery features to produce fragile watermarking [4–7,9–12,15,16,20,25]. Recovery information stored in non-tampered image areas is used to repair the tampered areas as long as the tampered areas are localized. As these fragile watermarking methods usually embed the original image's own message into itself, these types of proposed schemes are also called self-embedded fragile watermarking schemes [11,12,14,16,20,25].

In most fragile watermarking schemes, the process of tamper detection and image restoration adopts a block-based mechanism [6–8,11,14,17,18,20,23,25,28] but some of them adopted a pixel-based concept to design their watermarking scheme [18]. In 2009,

Zhang et al. proposed a self-embedding fragile watermarking that can tolerate up to a 59% tampering rate [4]. Their embedded watermark is a reference derived from the compressed original content using discrete cosine transform (DCT). Experimental results show that the average of recovered image quality with Zhang et al.'s scheme is 27.5 dB. Later, Zhang et al. proposed another scheme based on a reference-sharing mechanism [7]. Their embedded watermark is a reference, but is derived from different regions of the original content and is then shared by these regions for later restoration once a tampered area is identified. Experimental results showed that Zhang et al.'s scheme is able to restore the five most significant bits (MSBs) accurately, as long as the tampering rate is below 24% of an image size. In 2014, to improve the visual quality of the watermarked image, Yang et al. applied a halftone mechanism to derive recovery bits from a resized original image, and then used a Hash function to produce the authentication bits [10]. Since only one LSB plane is used for embedding, thus, the average PSNRs of the watermarked image with Yang et al.'s scheme is 51.1 dB, which is higher than those offered by Zhang et al.'s scheme. In 2017, Qin et al. proposed their scheme based on an overlapping embedding strategy [19]; the reference bits are derived from the average of each overlapping block and then embedded into the adaptive LSB layer of the image according to block's complexity. Since the six MSB of the mean value is used as the reference bits, their restored image quality can be up to 41 dB. In 2018, in order to reduce the computational cost, a scheme based on tornado-like code is proposed by Bravo-Solorio et al. [16]. The five MSB of the original image are used to generate two corresponding reference bit sequences, and then allocated to the three LSB of image with authentication bits. After the image is tampered with, an iterative mechanism is performed to restore the tampered regions of reference bit sequences. In the following year, Huang et al. classified blocks of an image into region of interest (ROI) and region of background (ROB) by adopting a graph-based visual saliency (GBVS) model. They collected the ROI backup information losslessly and allocated hidden data in a water-filling order to remain the high image quality of ROI [20]. In 2020, Chang et al. also applied the compression code derived by weight-based AMBTC to design self-recovery-based fragile watermarking [28]. In their scheme, the image quality of the restored image is slightly lower than that offered by scheme [10]; this is because the size of bitmap derived by weight-based AMBTC is only half of the original bitmap's size. However, their scheme could offer more stable image quality of the restored image compared with that of scheme [10] with the advantages of their proposed weight-based AMBTC.

In this paper, we propose a self-embedding fragile watermarking algorithm based on pixel pairs to improve the accuracy of tamper detection and to restore the tampered areas of an image. Recovery information derived from 16×16 non-overlapping blocks of image is pre-generated by our proposed Huffman-based absolute moment block truncation coding (HC-AMBTC) mechanism, which will be introduced in Section 2 in detail. The generated recovery information is then embedded into two LSBs of the original image with a turtle shell-based data hiding method [29] and a pre-determined matrix for later tamper detection and content recovery. At the recipient side, once the tampering detection is performed, the tampered pixel-pairs of the watermarked image can be restored to an acceptable condition as long as the tampered area is less than half of the watermarked image.

The rest of this paper is organized as follows. Section 2 briefly introduce our proposed HC-based absolute moment block truncation coding. Section 3 explains the proposed pixel pair-wise self-embedding fragile watermarking scheme. Section 4 demonstrates the experimental results, and compares our proposed scheme to five existing representative schemes. Finally, brief conclusions are given in Section 5.

2. HC-Based Absolute Moment Block Truncation Coding

To generate fewer representative features to serve as the recovery information and authentication bits at the same time, in this paper we proposed a variant of AMBTC called HC-based Absolute moment block truncation coding (HC-based AMBTC), which is a lossy compression scheme. To meet our objective, Huffman coding was adopted to transform

the bitmap of each 16×8 sized block into 6 different patterns denoted by Huffman codes. HC-based AMBTC divides an image into $n \times n$ non-overlapping blocks, and then divides pixels into two groups based on the mean value of all of the pixels in a block. One group contains pixels which are greater than or equal to the mean value, and the other group contains pixels that are smaller than the mean value. Two reconstruction values are derived and then used to represent these two groups with a bitmap. Finally, the bitmap is further compressed according to the Huffman code table, so that a compression effect can be achieved while preserving acceptable visual quality in the reconstructed image.

The encoding procedure for HC-based AMBTC is demonstrated in the following paragraphs. First, a grayscale image is divided into non-overlapping blocks with a size of $n \times n$ pixels. Next, calculate the mean value for each block by Equation (1). Then, record the corresponding bitmap with single bit “1” or “0” according to rules listed in Equation (3). Let P_i be a pixel in a block, where $1 \leq i \leq N$ and $N = n \times n$.

$$\bar{X} = \frac{1}{N} \sum_{i=1}^N P_i \quad (1)$$

And the first order absolute central moment can be calculated as follows:

$$\mu = \frac{1}{N} \sum_{i=1}^N |P_i - \bar{X}| \quad (2)$$

The rules for bitmap construction of HC-based AMBTC are the same as the conventional AMBTC. If a pixel value is greater than or equal to the mean value $\bar{X}$, it is set to the value “1” in the bitmap; otherwise, it is set to “0”:

$$B_i = \begin{cases} 1, & \text{if } P_i \geq \bar{X} \\ 0, & \text{if } P_i < \bar{X} \end{cases} \quad (3)$$

here B_i ($i = 1, 2, \dots, N$) is the value of the bitmap.

Once two groups are identified based on the generated bitmap, the representative quantizers for each group, denoted as “H” and “L”, can be derived by Equations (4) and (5), respectively, which are the same as those in the conventional block truncation coding [30,31].

$$H = \bar{X} + \frac{N\mu}{2k} \quad (4)$$

$$L = \bar{X} - \frac{N\mu}{2(N-k)} \quad (5)$$

where k is the number of pixels that are greater than or equal to mean value $\bar{X}$.

To reconstruct the image, elements of a bitmap assigned with “1” are replaced with the value “H” and elements assigned “0” are replaced with the value “L” according to Equation (6).

$$P_i = \begin{cases} H, & \text{if } B_i = 1 \\ L, & \text{if } B_i = 0 \end{cases} \quad (6)$$

The bitmap is divided into several non-overlapping sub-blocks with a size of 2×2 . Then, according to Figure 1, each sub-block can match a pattern, where a black grid represents “0” in the bitmap, and a white grid represents “1” in the bitmap. Each pattern reserves an edge direction of the original image, including the vertical, horizontal, slash, and full directions. In more detail, pattern—1 represents vertical, pattern—2 represents horizontal, pattern—3 represents left slash, pattern—4 represents right slash, and pattern—5 and pattern—6 represent full H and full L, respectively. After classifying all the sub-blocks, the bitmap of the sub-blocks can be replaced by the Huffman code. Figure 2 depicts each pattern’s Huffman code, and the Huffman coding length of each pattern is different, depending on the frequency of each pattern. An example is given below:

Pattern

Figure 1. Table of 2×2 block corresponding patterns.

Pattern	code
1	001
2	0001
3	00000
4	00001
5	1
6	01

Figure 2. Huffman code of six patterns.

Example 1. Assume that there is an 8×8 – sized image block with reconstruction values of 135 and 142, respectively, and Figure 3a shows the corresponding bitmap. Each bitmap of the sub-block can be replaced by a pattern number according to Figure 2, as shown in Figure 3b. Finally, the compression code of the block is $(10000111\ 10001110\ 0111000001110001000110011010111)_2$; the first 16 bits are binary reconstruction values, and the remaining 31 bits are bitmaps in the Huffman coded form.

Figure 3. A bitmap and its corresponding patterns. (a) Original 8×8 bitmap, (b) Corresponding patterns from (a).

3. Proposed Pixel Pair-Wise Fragile Watermarking Scheme

In order to improve the image quality of the watermarked image, we applied HC-based AMBTC to design a new self-embedding fragile watermarking scheme. With the assistance of our proposed HC-based AMBTC, the representative features of the image will remain, with fewer data. The compression code derived from our proposed HC-based AMBTC will serve as the authentication code and recovery information at the same time, and these compression codes will be called recovery information (*RI*) in the following paragraphs. Considering distortion may be caused during the watermark embedding, we designed a predetermined reference matrix and adopted a turtle-shell data hiding concept to embed *RI* into the original image. Two neighboring pixels are used to carry 4-bits of *RI* during watermark embedding, and pixel-based tamper detection/recovery can be conducted; therefore, our proposed watermarking scheme is also called pixel pair-wise fragile watermarking scheme. Our proposed scheme has three stages: (1) watermark embedding, (2) tamper detection, and (3) recovery.

In the first stage, set two LSB of original image I_o to zero to produce processed image I_p , then recovery information *RI* is generated and embedded into image I_p sized with $h \times w$ pixels. Later, in the tamper detection stage the hidden *RI* are then extracted and compared with new *RI* generated from tampered image I_t to determine whether or not pixels have been tampered with. Finally, untampered recovery information *RI* is then used to reconstruct a tampered area at the recovery stage. Details about these three stages are given in Sections 3.1 and 3.2, respectively.

3.1. Watermark Embedding

Before embedding the recovery information into the original gray-scale image I_o sized with $h \times w$ pixels, I_o is divided into several non-overlapping blocks. Each block is sized as 16×16 pixels and is denoted as B_{ROIi} , where $i = (1, 2, \dots, \frac{h}{16} \times \frac{w}{16})$. A flowchart of watermark embedding is depicted in Figure 4. Details for generation of the recovery information and the watermarking embedding procedures are given in the following subsections.

Figure 4. Flowchart of watermark embedding procedure.

3.1.1. Recovery Information Generation

In order to generate recovery information (*RI*), two region types must be defined: a region of interest (*ROI*) and region of non-interest (*RONI*). *ROI* covers the crucial content of image I_o ; while *RONI* covers non-crucial image content. In our scheme, the size of the *ROI* region is 480×480 , and only the *ROI* region can be recovered with our proposed scheme. In other words, the *RI* will only be generated from the *ROI*. Examples of *ROI* and

RONI are demonstrated in Figure 5. Then, the HC-based AMBTC described in Section 2 is used to compute each 16×16 non-overlapping block B_{ROIi} , B_{ROIi} , where $i = (1, 2, \dots, \frac{h_{ROI}}{16} \times \frac{w_{ROI}}{16})$, from the preprocessed image I_p . Finally, each B_{ROIi} will generate recovery information with unfixed bits length RI_i , where $i = (1, 2, \dots, \frac{h_{ROI}}{16} \times \frac{w_{ROI}}{16})$, which contains 8-bits of high average value H , 8-bits of low average value L and an unfixed bits length pattern code.

Figure 5. Example of ROI and RONI.

In a 16×8 sized block, recovery information RI_i , where $i = (1, 2, \dots, \frac{h_{ROI}}{16} \times \frac{w_{ROI}}{16})$, is embedded into pixel pairs and is used later to detect a tampered area and to reconstruct the corresponding damaged block. Since each RI_i contains unfixed bits, the embedding strategy used in the proposed scheme can hide 4 bits in each pixel pair. In other words, each 16×8 sized block provides 256 $(= (16 \times 8/2) \times 4)$ bits as hidden space, and each RI will first repeat its own information until the length is 256 bits, then embed into I_p . The benefit of doing this is that RI can then be used as authentication.

3.1.2. Embedding Strategy

Once the RI for each 16×16 non-overlapping block is generated, a reference matrix M as shown in Figure 6 is constructed according to Equation (9). This matrix acts as a reference table to embed recovery information RI into processed image I_p . Reference matrix M is sized as 256×256 pixels, with a range of value $V_{(P_i, P_{i+1})}$, which is between 0 and 15. In other words, each value V in reference matrix M can be expressed in 4-bit binary and value $V_{(P_i, P_{i+1})}$ is defined as Equation (7). An example of the partial content of reference matrix M is shown in Figure 6. In reference matrix M , P_i and P_{i+1} represent a non-overlapped pixel pair in I_p . The embedding strategy is to find value V , which is close to the coordinate (P_i, P_{i+1}) and is the same as the decimal value of (4-bits RI) from reference matrix M .

$$V_{P_i} = P_i, \quad V_{P_{i+1}} = \begin{cases} 0, & \text{if } P_{i+1} = 0 \\ V_{P_{i+1}-1} + 3, & \text{if } P_{i+1} \bmod 3 = 1, \\ V_{P_{i+1}-1} + 4, & \text{else} \\ = (V_{P_i} + V_{P_{i+1}}) \bmod 16. \end{cases} \quad V_{(P_i, P_{i+1})} \quad (7)$$

original pixel pairs, these three pairs are located in reference matrix M as $V(4, 4) = 2$, $V(5, 6) = 11$ and $V(6, 2) = 13$, respectively.

- i. To embed $(0111)_2$ into pixel pair (4, 4): First, convert $(0111)_2$ into a decimal value as 7, search for a coordinate that is closest to (4, 4) and with a V value that is equal to 7. Finally, coordinate (5, 5) is found since it is the closest digit to “7” from (4, 4).
- ii. To embed $(1011)_2$ into pixel pair (5, 6): First, convert $(1011)_2$ into a decimal value as 11, search for a coordinate that is closest to (5, 6) and has a V value that is equal to 11. Since the original coordinate (5, 6) maps to $V(5, 6) = 11$, the coordinate remains unchanged.
- iii. To embed $(0011)_2$ into pixel pair (6, 2): First, convert $(0011)_2$ into a decimal value as 3, search for a coordinate that is closest to (6, 2) and with a V value that is equal to 3. In this case, there are two candidates mapped to 3, which are (5, 4) and (8, 3). Finally, the coordinate (8, 3) is selected based on the search path demonstrated in Figure 8.

Based on the above explanation, each 16×16 block of ROI produces a 256-bit RI . To begin the watermarking embedding process, image I_p is divided into several non-overlapping blocks with a size of 16×8 denoted as B_{pi} , where $i = (1, 2, \dots, \frac{h}{16} \times \frac{w}{8})$; each block B_{pi} contains 64-pixel pairs and each pixel pair carry 4-bits of RI . In order to increase the probability of image recovery, recovery information RI of each block B_{ROIi} located in ROI is embedded into B_{pi} with a random seed, and vice versa. Finishing the above processes produces the watermarked image I_w .

3.2. Detection and Recovery of Tampered Area

Assume that the watermarked image I_w has been tampered during transmission and the tampered image is denoted as I_t . First, the hidden 4 bits of each pixel pair of I_t can be extracted with the assistance of shared reference matrix M . Next, use the HC-based AMBTC for I_t to generate a new RI , then compare the new RI' with the extracted RI . If the extracted 2-bit RI_i from a single pixel is the same as the new one, this means the current pixel has not been tampered. Otherwise, it means that the current pixel pair has been attacked. Moreover, according to the recovery information extracted from undamaged pixel pairs, a pixel-based recovery mechanism is exploited to recover damaged pixels. A flowchart of tamper detection and recovery for the proposed scheme is given in Figure 9. Additionally, details regarding tampered pixel detection and recovery are presented in Sections 3.2.1 and 3.2.2, respectively.

Figure 9. Flowchart of tamper detection and recovery.

3.2.1. Tampered Pixel Detection

As mentioned in Section 3.2, RI' is generated from I_t with a pre-shared seed. At the tampering detection stage, the generated RI' is compared with RI , which is extracted from

tampered image I_t . If both RI s are the same, then the corresponding pixel has not been tampered. Otherwise, the corresponding pixel pair has been attacked.

Since each pixel carries a 2-bit RI , the possibility of misjudgment is equal to 2^{-2} . Thus, there is a 25% chance that the tampered pixel will be verified and misjudged as “untampered.” As such, post-processing is done here to reduce such misjudgments. Based on a majority voting policy, post-processing is done after tampered pixel detection. Our rule is quite straightforward—if a pixel marked as “untampered” has more than four “tampered” neighboring pixels surrounding it, this pixel should be re-judged as “tampered”.

3.2.2. Tampered Pixel Recovery

After tamper pixel detection is completed, all recovery information can be extracted from pixel pairs that have been marked as “untampered”. Extract RI from all “untampered” pixels. The 2-bit RI extracted from each pixel pair is recombined into 8 bits of a high average value, 8 bits of a low average value, and an unfixed bits length pattern code.

4. Experiment Results

To test the performance of the proposed scheme on the image quality of a water-marked image, test its hiding capacity, and to observe the results under various attacks, we implemented our proposed scheme by using Python 3.7. Experiments were conducted on a computer comprised of an Intel i7 – 4790 (3.60 GHz) CPU, with 8 GB of memory and the Windows 10 Home basic 64-bit operating system. Four standard grey-scale images were used for the test images: Lena, Elaine, Baboon, and Airplane, as shown in Figure 10. The size of each grey-scale image is 512×512 pixels.

Figure 10. Four original grey-scale images: (a) Lena, (b) Elaine, (c) Baboon, and (d) Airplane.

Two general criterions were used to evaluate the performance of the proposed scheme: image quality and hiding capacity. The peak signal-to-noise ratio (PSNR) is defined as follows to evaluate the quality of the watermarked images and recovered images.

$$\text{PSNR} = 10 \log_{10} \left(\frac{255^2}{\text{MSE}} \right), \quad (8)$$

where mean square error (MSE) is as follows:

$$\text{MSE} = \frac{1}{H \times W} \sum_{i=1}^H \sum_{i=1}^W (I_i - I'_i)^2, \quad (9)$$

where W stands for width and H stands for height of the images; I_i and I'_i are the pixel values of the cover images and the watermarked images, respectively. As shown in Equation (9), the smaller the MSE, the larger the PSNR. Generally speaking, once a PSNR value is larger than 30 dB, it means a human being would find it difficult to distinguish the difference between the cover image and watermarked image. Note that a good watermarking scheme should provide a higher image quality for a watermarked image while maintaining a larger hiding capacity. Unfortunately, many existing schemes have pointed out there is a trade-off between image quality and hiding capacity.

The four watermarked images generated from Figure 10 with the proposed scheme are demonstrated in Figure 11 with PSNR values of 46.8 dB, 46.8 dB, 46.8 dB, and 46.8 dB, respectively. It is proven that our strategies work successfully: using HC-based AMBTC to derive the features of images as RI and adopting the turtle-shell data hiding method to embed RI into the original image. This is because our proposed HC-based AMBTC mentioned in Section 2 successfully reduces the size of compression code comparing with the conventional ABMTC. Once the compression code of our HC-based AMBTC is used to serve as RI , the amount of RI can be remained the least. In other words, by combining our HC-based AMBTC and turtle-shell data hiding, more representative features of the original image, which, using the least amount, are embedded into two neighboring pixels in the original image at the cost of causing the least distortion. Therefore, the visual difference between the original image and watermarked image is quite small, which means it is difficult for the human eye to notice a difference.

Four kinds of attacks are presented in Figure 12. The tampering rates α for these four tampered images, which are represented as the ratios between the amount of tampered pixels and the amount of all pixels in an image, were 1.93%, 0.6%, 3.44%, and 1.41%, respectively. The results of the tampered pixel detection are shown in Figure 13, where a black area denotes “tampered” and white areas denote “untampered”. Table 1 presents the tampering detection results of the proposed scheme, using the following abbreviations: true positive (TP), true negative (TN), false positive (FP), false negative (FN), true positive rate (TPR), and false positive rate (FPR). Based on the detection tampering rules mentioned in Section 3.2.1, Table 1 indicates an improvement in the performance of TPR and FPR. The average TPR is up to 0.8637, and the average of FPR is 0.0009.

Figure 11. Four watermarked images: (a) PSNR = 46.8 dB, (b) PSNR = 46.8 dB, (c) PSNR = 46.8 dB, and (d) PSNR = 46.8 dB.

Figure 12. Four tampered images: (a) $\alpha = 1.93\%$, (b) $\alpha = 0.6\%$, (c) $\alpha = 3.44\%$, (d) $\alpha = 1.41\%$.

Figure 13. Results of tampered region detection for Figure 12a,d: (a) Lena, (b) Elaine, (c) Baboon, and (d) Airplane.

Table 1. TP, TN, FP, FN, TPR and FPR of the proposed scheme.

Images	TP	TN	FP	FN	TPR	FPR
Lena	4950	256882	110	202	0.9608	0.0004
Elaine	1447	260183	121	393	0.7864	0.0004
Baboon	8740	251707	287	1410	0.8611	0.0011
Airplane	3229	257870	459	586	0.8464	0.0018

Table 2 shows comparisons of theoretical values for the proposed scheme and five state-of-the-art schemes [4,7,10,18,23,28], including the PSNRs of the watermarked images, PSNRs of the recovered images, and the maximum tolerable tampering rate. Due to the lower distortion caused by the watermark embedding in our proposed scheme, the visual quality of the watermarked image for the proposed scheme is better than the other schemes, except for scheme [10]. The column “PSNR of recovered image” presents the average image quality of recovered results from the tampered images. “Condition of recovery” shows the conditions for successful recovery in all the schemes, i.e., maximum tolerable tampering rates. The advantage of scheme [18] is that it can completely restore the watermarked image quality with their proposed Iterative restoration algorithm; however, it can only endure a 26% tampering rate. Scheme [7] is capable of recovering the original image at 5 MSB if the tampering rate is lower than 24%. The maximum tolerable tampering rates of [10,23,28] and proposed scheme are about the same, and the image quality of watermarked image offered

by schemes [10,28] is higher than that of ours, but the visual quality of the recovered image provided our proposed scheme is better and more reliable. As for the scheme from [23], the data related to image quality of watermarked image and restored images are mapped to condition of recovery (α) which ranges from 5% to 50%. The average image quality of watermarked image and restored image under $\alpha = 50\%$ are around 45 dB and 29.98 dB, respectively. Therefore, we can conclude our proposed scheme still outperforms schemes of [10,23,28] on image quality.

Table 2. Performance comparisons of the proposed scheme versus five schemes [4,7,10,18,23,28].

Schemes	PSNR of Watermarked Image	PSNR of Recovered Image	Condition of Recovery
Scheme in [4]	37.9 dB	[26, 29] dB	$\alpha < 59\%$
Scheme in [7]	37.9 dB	40.7 dB	$\alpha < 24\%$
Scheme in [10]	51.3 dB	[24, 36] dB	$\alpha < 50\%$
Scheme in [18]	37.9 dB	$+\infty$	$\alpha < 26\%$
Scheme in [23]	[37.92, 54.13] dB	[28.63, 46.98] dB	$\alpha < 50\%$
Scheme in [28]	49.76 dB	34.65 dB	$\alpha < 50\%$
Proposed Scheme	46.8 dB	[32, 42] dB	$\alpha < 50\%$

To prove the recovery performance of our proposed scheme, Figure 14 demonstrates the recovery results for four attacked images, with the PSNRs of their recovered images being: 37.9 dB, 41.7 dB, 31.8 dB, and 35.5 dB, respectively. The average PSNR performance is 36.7 dB. Moreover, from a visual point of view, the human visual system cannot distinguish the difference between these four reconstructed images and the original images shown in Figure 11.

Figure 14. Four recovered images: (a) PSNR = 37.9 dB, (b) PSNR = 41.7 dB, (c) PSNR = 31.8 dB, and (d) PSNR = 35.5 dB.

5. Conclusions

This paper proposed a new self-embedding fragile watermarking scheme that uses a reference matrix as an embedding method to provide improved embedding capacity and lower distortion than previous schemes. In order to maintain high quality watermark images, each non-overlapping pixel pair is used as a coordinate in a reference matrix to embed recovery information, and each provides a 4-bit capacity as hidden space. The embedded recovery information is compared with the recovery information generated from suspicious images to detect tampered regions, and to restore a watermarked image. The experimental results show that the visual quality of the watermark image of the scheme is 46.8 dB on average. Moreover, the proposed scheme also achieves tamper recovery with a higher tampering rate and improved performance compared to previous schemes.

Consider that artificial intelligence has been adopted in various areas [32], including image-based applications such as face recognition [33], and has been widely used for access control, in the future, we will consider the possibility of adopting artificial intelligence or deep learning techniques in designing watermarking schemes and plan to extend the applications of watermarking from the conventional images to specific images, such as faces or figures.

Author Contributions: Conceptualization, S.-L.H. and C.-C.L.; methodology, C.-C.L.; software, S.-L.H.; validation, C.-C.L.; writing—original draft preparation, S.-L.H.; writing—review and editing, C.-C.L.; project administration, C.-C.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Ministry of Science and Technology, Taiwan, grant number 109-2410-H-167-014.

Data Availability Statement: Data available in a publicly accessible repository.

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

References

1. Fridrich, J.; Goljan, M. Images with self-correcting capabilities. In Proceedings of the 1999 International Conference on Image Processing (Cat. 99CH36348), Kobe, Japan, 24–28 October 1999; pp. 792–796.
2. Zhu, X.Z.; Ho, T.S.; Marziliano, P. A New Semi-fragile Image Watermarking with Robust Tampering Restoration Using Ir-regular Sampling. *Signal Process. Image Commun.* **2007**, *22*, 515–528. [\[CrossRef\]](#)
3. Zhang, X.; Wang, S. Fragile Watermarking with Error-Free Restoration Capability. *IEEE Trans. Multimed.* **2008**, *10*, 1490–1499. [\[CrossRef\]](#)
4. Zhang, X.; Wang, S.; Feng, G. Fragile watermarking scheme with extensive content restoration capability. In *International Workshop on Digital Watermarking*; Springer: Berlin/Heidelberg, Germany, 2009; pp. 268–278.
5. Lee, T.-Y.; Lin, S.D. Dual watermark for image tamper detection and recovery. *Pattern Recognit.* **2008**, *41*, 3497–3506. [\[CrossRef\]](#)
6. Yang, C.-W.; Shen, J.-J. Recover the tampered image based on VQ indexing. *Signal Process.* **2010**, *90*, 331–343. [\[CrossRef\]](#)
7. Zhang, X.; Wang, S.; Qian, Z.; Feng, G. Reference Sharing Mechanism for Watermark Self-Embedding. *IEEE Trans. Image Process.* **2011**, *20*, 485–495. [\[CrossRef\]](#)
8. Raj, I.K. Image Data Hiding in Images Based on Interpolative Absolute Moment Block Truncation Coding. *Math. Model. Sci. Comput.* **2012**, *283*, 456–463.
9. Singh, D.; Shivani, S.; Agarwal, S. Self-embedding pixel wise fragile watermarking scheme for image authentication. In *International Conference on Intelligent Interactive Technologies and Multimedia*; Springer: Berlin/Heidelberg, Germany, 2013; Volume 10, pp. 111–122. [\[CrossRef\]](#)
10. Yang, S.; Qin, C.; Qian, Z.; Xu, B. Tampering detection and content recovery for digital images using halftone mechanism. In Proceedings of the 2014 Tenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Kitakyushu, Japan, 27–29 August 2014; pp. 130–133.
11. Chang, C.C.; Liu, Y.; Nguyen, T.S. A novel turtle shell based scheme for data hiding. In Proceedings of the 2014 Tenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Kitakyushu, Japan, 27–29 August 2014; pp. 89–93.
12. Dhole, V.S.; Patil, N.N. Self embedding fragile watermarking for image tampering detection and image recovery using self recovery blocks. In Proceedings of the 2015 International Conference on Computing Communication Control and Automation, Pune, India, 26–27 February 2015; pp. 752–757.
13. Qin, C.; Wang, H.; Zhang, X.; Sun, X. Self-embedding fragile watermarking based on reference-data interleaving and adaptive selection of embedding mode. *Inf. Sci.* **2016**, *373*, 233–250. [\[CrossRef\]](#)

14. Manikandan, V.M.; Masilamani, V. A context dependent fragile watermarking scheme for tamper detection from de-mosaicked color images. In Proceedings of the ICVGIP '16: Tenth Indian Conference on Computer Vision, Graphics and Image Processing, Madurai, India, 18–22 December 2016; pp. 1–8.
15. Qin, C.; Ji, P.; Wang, J.; Chang, C.-C. Fragile image watermarking scheme based on VQ index sharing and self-embedding. *Multimed. Tools Appl.* **2016**, *76*, 2267–2287. [[CrossRef](#)]
16. 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](#)]
17. Bravo-Solorio, S.; Calderon, F.; Li, C.-T.; Nandi, A.K. Fast fragile watermark embedding and iterative mechanism with high self-restoration performance. *Digit. Signal Process.* **2018**, *73*, 83–92. [[CrossRef](#)]
18. Lin, C.C.; Huang, Y.H.; Tai, W.L. A Novel Hybrid Image Authentication Scheme Based on Absolute Moment Block Truncation Coding. *Multimed. Tools Appl.* **2017**, *76*, 463–488. [[CrossRef](#)]
19. Li, W.; Lin, C.-C.; Pan, J.-S. Novel image authentication scheme with fine image quality for BTC-based compressed images. *Multimed. Tools Appl.* **2015**, *75*, 4771–4793. [[CrossRef](#)]
20. Liu, X.-L.; Lin, C.-C.; Yuan, S.-M. Blind Dual Watermarking for Color Images' Authentication and Copyright Protection. *IEEE Trans. Circuits Syst. Video Technol.* **2018**, *28*, 1047–1055. [[CrossRef](#)]
21. Huang, R.; Liu, H.; Liao, X.; Sun, S. A divide-and-conquer fragile self-embedding watermarking with adaptive payload. *Multimed. Tools Appl.* **2019**, *78*, 26701–26727. [[CrossRef](#)]
22. Wang, X.; Li, X.; Pei, Q. Independent Embedding Domain Based Two-stage Robust Reversible Watermarking. *IEEE Trans. Circuits Syst. Video Technol.* **2019**, *30*, 2406–2417. [[CrossRef](#)]
23. Roy, S.S.; Basu, A.; Chattopadhyay, A. On the Implementation of A Copyright Protection Scheme Using Digital Image Watermarking. *Multimed. Tools Appl.* **2020**, *79*, 13125–13138.
24. Su, G.D.; Chang, C.C.; Chen, C.C. A hybrid-Sudoku based fragile watermarking scheme for image tampering detection. *Multimed. Tools Appl.* **2021**, 1–23. [[CrossRef](#)]
25. Huy, P.Q.; Anh, D.N. Saliency guided image watermarking for anti-forgery. In *Soft Computing for Biomedical Applications and Related Topic*; Springer: Cham, Switzerland, 2021; pp. 183–195.
26. Chang, C.-C.; Lin, C.-C.; Su, G.-D. An effective image self-recovery based fragile watermarking using self-adaptive weight-based compressed AMBTC. *Multimed. Tools Appl.* **2020**, *79*, 24795–24824. [[CrossRef](#)]
27. Gola, K.K.; Gupta, B.; Iqbal, Z. Modified RSA Digital Signature Scheme for Data Confidentiality. *Int. J. Comput. Appl.* **2014**, *106*, 13–16.
28. National Institute of Standards and Technology. Secure Hash Standard (SHS). In *Federal Information Processing Standards*; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2015.
29. Stallings, W. *Cryptography and Network Security Principles and Practices*, 7th ed.; Pearson Education: London, UK, 2016.
30. Delp, E.; Mitchell, O. Image Compression Using Block Truncation Coding. *IEEE Trans. Commun.* **1979**, *27*, 1335–1342. [[CrossRef](#)]
31. Lema, M.; Mitchell, O. Absolute Moment Block Truncation Coding and Its Application to Color Images. *IEEE Trans. Commun.* **1984**, *32*, 1148–1157. [[CrossRef](#)]
32. Huh, J.-H.; Seo, Y.-S. Understanding Edge Computing: Engineering Evolution with Artificial Intelligence. *IEEE Access* **2019**, *7*, 164229–164245. [[CrossRef](#)]
33. Lee, H.; Park, S.-H.; Yoo, J.-H.; Jung, S.-H.; Huh, J.-H. Face Recognition at a Distance for a Stand-Alone Access Control System. *Sensors* **2020**, *20*, 785. [[CrossRef](#)] [[PubMed](#)]