International Journal of Computer Science and Information Technology
ISSN: 3005-9682 (Print), ISSN: 3005-7140 (Online) | Volume 4, Number 3, Year 2024
DOI: https://doi.org/10.62051/ijcsit.v4n3.52
Journal homepage: https://wepub.org/index.php/IJCSIT/index

Reversible Image Authentication Method Based on Pre-ordered IPVO

Chang Yang, Zhengwei Zhang *, Hao Yue

College of Computer and Software Engineering, Huaiyin institute of technology University, Huai an, China

*Corresponding Author: Zhengwei Zhang

ABSTRACT

To enhance the tampering accuracy of current reversible image authentication methods, we propose a novel algorithm based on pre-sorting-IPVO. The process begins by dividing the image into 2×2 blocks organized in a black and white checkerboard pattern and further categorizing them into smooth and textured blocks based on image characteristics to generate an authentication watermark. In smooth blocks, the pre-sorting-IPVO method is applied to pre-sort pixel values, prioritizing larger pixel values. Authentication watermarks are embedded in the textured blocks using the difference expansion method. Experimental results demonstrate that this algorithm effectively leverages pixel correlation to improve tampering accuracy, maintaining high detection accuracy across various conditions. It is capable of authenticating images with both smooth and complex textures.

KEYWORDS

Image authentication; Tamper detection; IPVO; Difference expansion; Reversible image watermarking

1. INTRODUCTION

The widespread transmission of digital information has introduced significant security risks, intensifying the urgent need for robust information security measures. As image processing software becomes more accessible, tampering with images and other multimedia files has become increasingly easy. Ensuring the originality and integrity of these vast multimedia files during storage and transmission is a critical challenge that needs immediate attention. Reversible watermarking [1, 2] has emerged as a vital solution for tampering detection, particularly in specialized sectors such as the military, remote sensing, and medical industries. Utilizing reversible watermarking technology to authenticate and protect these files is of paramount importance.

Reversible authentication (RA) [3, 4] is a technique that combines reversible information hiding and fragile watermarking to achieve sensitive image authentication. Lo et al. [5] pioneered a reversible authentication method featuring tamper localization capabilities. In their approach, an image is divided into several sub-blocks, with the central pixel of each block serving as a reference value. The differences between this reference value and the other pixels in the block are used to construct a prediction error histogram. This histogram is then shifted to embed an N-bit authentication code into the N sub-blocks. Following Lo's foundational work, a variety of new methods have been developed, including those based on Pixel Value Ordering (PVO) [6] and its enhanced version, Improved Pixel Value Ordering (IPVO) [7].

Recently, Li et al. [6] introduced a reversible watermarking scheme utilizing PVO, which excels in prediction accuracy due to its consideration of the correlation between pixel values and their positions. In this method, the carrier image is divided into non-overlapping blocks of equal size. Authentication information is embedded into the maximum or minimum pixels of each block by adjusting their prediction differences. This ensures that the PVO within each block remains unchanged, maintaining authentication reversibility. Building on this concept, several advancements have been made, such as the IPVO method [7-10], which incorporates spatial information—specifically, the relative positioning of pixels—to create a new prediction error histogram. This method uses smoother blocks for reversible embedding.

Yin et al. [11] further refined this approach by initially scanning the image using a Hilbert curve, dividing the resulting pixels into N sub-blocks, and then applying the IPVO method to embed an N-bit authentication code. However, a practical challenge arises when some sub-blocks exhibit zero embedding capacity, making it difficult to authenticate these non-embeddable blocks. To address this issue, Hong et al. [12] enhanced the method by integrating the Least Significant Bit (LSB) substitution technique with the IPVO method, thereby achieving authentication for these challenging sub-blocks. Yao et al. [12] advanced this work by proposing an adaptive block-based reversible authentication method, which further divides images into smaller sub-blocks to enhance authentication accuracy.

This paper introduces a reversible image authentication method based on pre-sorting-IPVO. This approach improves upon the existing IPVO algorithm by fully leveraging pixel correlations to enhance tampering accuracy.

2. RELATED WORK

The reversible image authentication algorithm explored in this paper primarily leverages techniques such as Improved Pixel Value Ordering (IPVO) and difference expansion to efficiently detect image tampering.

2.1. IPVO Algorithm

The section headings are in boldface capital and lowercase letters. Second level headings are typed as part of the succeeding paragraph (like the subsection heading of this paragraph).

The Improved Pixel Value Ordering (IPVO) algorithm enhances the original Pixel Value Ordering (PVO) method by utilizing the spatial relationships between predicted pixels to embed data within prediction errors of 0 and 1, thereby effectively improving detection accuracy. The implementation of the IPVO-based method is as follows:

S1: Initially, the image is divided into non-overlapping 2×2 blocks, with each block containing n pixels. The pixels within each block are then sorted in ascending order to obtain the desired arrangement.to get $\left\{p_{(1)}, p_{(2)}, \cdots, p_{\sigma(n)}\right\}$, This unique one-to-one mapping occurs when the condition $\sigma:\ \{1,2,\cdots\} \to \{1,2,\cdots\}$ holds true, If $p_{\sigma(i)} = p_{\sigma(j)}$, and i＜j, then $p_{\sigma(1)} \le p_{\sigma(2)} \le \cdots \le p_{\sigma(n)}$. The second largest pixel is used as the predicted value $p_{\sigma(n)}$ of the largest pixel p, and the second smallest pixel $p_{\sigma(2)}$ is used as the predicted value $p_{\sigma(1)}$ of the smallest pixel p. The two prediction errors, dmax and dmin, can be derived from equations (1) and (2) as follows:

$$d_{\max} = p_u - p_v \tag{1}$$

$$d_{\min} = p_s - p_r \tag{2}$$

Here, (v) and (t) represent the maximum indicator values of the predicted pixel and the actual pixel, respectively, while (u) and (s) denote the minimum indicator values of the predicted pixel and the actual pixel.

S2: A watermark b∈{0, 1} is embedded by modifying the maximum pixel $p_{\sigma(n)}$, as illustrated in Equation (3):

$$\hat{P}_{\sigma(n)} = \begin{cases} P_{\sigma(n)} + b & \text{if } d_{max} \in \{1,0\} \\ P_{\sigma(n)} + 1 & \text{if } d_{max} > 1 \text{ or } d_{max} < 0 \end{cases} \tag{3}$$

Using the minimum pixel $p_{\sigma(1)}$ and maximum pixel $p_{\sigma(n)}$, Equation (4) describes the process of embedding a watermark in (p).

$$\hat{P}_{\sigma(1)} = \begin{cases} P_{\sigma(1)} - b & \text{if } d_{max} \in \{1,0\} \\ P_{\sigma(1)} - 1 & \text{if } d_{min} > 1 \text{ or } d_{min} < 0 \end{cases} \tag{4}$$

S3: The smallest pixel $p_{\sigma(1)}$ follows the largest pixel $p_{\sigma(n)}$. After embedding the secret data, the order of pixel values within the block remains unchanged. The secret data can be extracted from the largest pixel $p_{\sigma(n)}$, as shown in Equation (5).

$$b = \begin{cases} 0 & \text{if } \hat{d}_{max} \in \{1,0\} \\ 1 & \text{if } \hat{d}_{max} \in \{2, -1\} \end{cases} \tag{5}$$

Where dˆmax is the corrected forecast error and dmax

S4: For the smallest pixels $p_{\sigma(1)}$, data extraction and image recovery are conducted as per Equations (6) and (7).

$$b = \begin{cases} 0 & \text{if } \hat{d}_{min} \in \{1,0\} \\ 1 & \text{if } \hat{d}_{min} \in \{2, -1\} \end{cases} \tag{6}$$

$$P_{\sigma(1)} = \begin{cases} \hat{P}_{\sigma(1)} & \text{if } \hat{d}_{min} \in \{0, 1\} \\ \hat{P}_{\sigma(1)} - 1 & \text{if } \hat{d}_{min} < 0 \text{ or } \hat{d}_{min} > 1 \end{cases} \tag{7}$$

The $\hat{d}_{min}$ corrected forecast error is dmin.

This paper introduces a pre-sorting scheme within the smoothing block embedding process. It arranges the pixels in ascending order, replacing larger position values with the original smaller ones. By prioritizing the collection of larger pixel values, this approach enhances the embedding capacity and improves tampering detection accuracy.

2.2. Calculation of Forecast Error and Pixel Classification

First, the original image is divided into 2 × 2 sub-blocks, which are further categorized into two sets following a black-and-white checkerboard pattern: one set of black pixels and one set of white pixels. White pixels are predicted and processed based on their adjacent black pixels, while black pixels are

predicted and processed using their adjacent white pixels. Since the prediction value calculation method is identical for both black and white pixels, the following explanation will focus on black pixels to illustrate how the algorithm calculates the prediction error.

Taking the black pixel block as an example, the prediction error is determined using the pixels from the four nearest adjacent blocks as reference points. Within each 2 × 2 sub-block $\{x_1, x_2, x_3, x_4\}$, the surrounding eight neighboring pixel values are employed for two predictions. This approach aims to fully leverage the correlation between pixels, thereby increasing the available positions for embedding the authentication watermark and enhancing tamper detection accuracy, as illustrated in Figure 1.

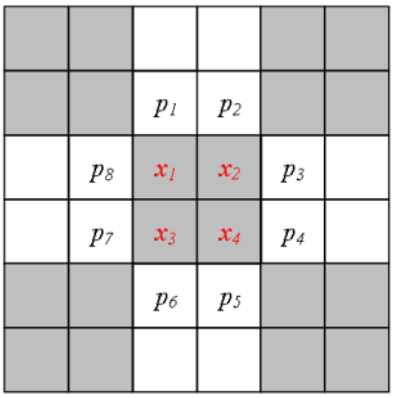


Figure 1. Schematic Diagram for Calculating Prediction Error

(1) Compute $\{x_1, x_2, x_3, x_4\}$, ph1 and ph2 by taking the average of the upper and lower pixels among the surrounding eight neighboring pixels p1-p8. Similarly, calculate pv1 and pv2 by averaging the left and right pixels, as shown in Equations (8)–(11).

$$p_{h1} = round\left(\frac{p_3 + p_8}{2}\right) \tag{8}$$

$$p_{h2} = round\left(\frac{p_4 + p_7}{2}\right) \tag{9}$$

$$p_{v1} = round\left(\frac{p_1 + p_6}{2}\right) \tag{10}$$

$$p_{v2} = round\left(\frac{p_2 + p_5}{2}\right) \tag{11}$$

(2) Determine the predicted value of $\{x_1, x_2, x_3, x_4\}$ using $\{\hat{x}_1, \hat{x}_2, \hat{x}_3, \hat{x}_4\}$ Equations (12)–(15). For instance, when predicting the value of x_1, consider the mean value derived from the pixels located above and to the left of x_1. Additionally, incorporate the average values of the upper and lower pixels, as well as the left and right pixels within the black block where x_1 is situated. This method aims to concentrate the prediction errors of the pixels within the block, thereby increasing the embedding capacity and enhancing tamper detection accuracy when embedding watermarks.

$$\hat{x}_1 = round\left(\frac{p_1 + p_8 + p_{v1} + p_{h1}}{4}\right) \tag{12}$$

$$\hat{x}_2 = round\left(\frac{p_2 + p_3 + p_{v2} + p_{h1}}{4}\right) \tag{13}$$

$$\hat{x}_3 = round\left(\frac{p_7 + p_6 + p_{v1} + p_{h2}}{4}\right) \tag{14}$$

$$\hat{x}_4 = round\left(\frac{p_4 + p_5 + p_{v2} + p_{h2}}{4}\right) \tag{15}$$

Here, the $\{\hat{x}_1, \hat{x}_2, \hat{x}_3, \hat{x}_4\}$ represent the predicted values within the $\{x_1, x_2, x_3, x_4\}$. The notation $round(\bullet)$ denotes the rounding function. $P = \{p_1, p_2, p_3, \cdots, p_8\}$.

As an efficient reversible watermarking algorithm, IPVO allows the choice of '0' and '1' as expandable errors, making the ratio of these values crucial for enhancing tamper detection accuracy. The definition of IPVO indicates that prediction errors of +1 or -1 are often influenced by the pixel's position within a block. Prioritizing the collection of larger pixel values can increase the likelihood of obtaining more prediction errors of +1. Consequently, determining the sequence in which pixels within a block are collected is essential for maximizing the number of prediction errors of +1. Ensuring that larger pixel values are gathered first is key. By doing so, and calculating prediction errors according to Equations (16) and (17), the method ensures that larger pixel values are reduced, thus producing more prediction errors of +1. Therefore, pre-ordering the pixel values to prioritize larger ones is recommended.

Pre-sorting the result of ascending order of pixel values by $\{x_1, x_2, x_3, x_4\}$ can get the pre-sorting sequence of $\{\hat{x}_1, \hat{x}_2, \hat{x}_3, \hat{x}_4\}$. Ascending order of $\{\tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \tilde{x}_4\}$ can get $\left\{\tilde{x}_{\sigma(1)}, \tilde{x}_{\sigma(2)}, \tilde{x}_{\sigma(3)}, \tilde{x}_{\sigma(4)}\right\}$. The calculation of the prediction error after ascending order is shown in Equations (16) and (17).

$$e_{\max}(i) = \tilde{x}_u - \tilde{x}_v$$

$$e_{\max}(i) = \tilde{x}_u - \tilde{x}_v$$

$$\begin{cases} u = \max(\sigma(n), \sigma(i)) \\ v = \min(\sigma(n), \sigma(i)) \end{cases} \quad i \in \{n+1, \cdots, 4\}, \quad n \in \{2, 3\} \tag{16}$$

$$e_{\min}(i) = \tilde{x}_r - \tilde{x}_t$$

$$\begin{cases} r = \min(\sigma(n), \sigma(i)) \\ t = \max(\sigma(n), \sigma(i)) \end{cases} \quad i \in \{1, \cdots, n-1\}, \quad n \in \{2, 3\} \tag{17}$$

Where $e_{\max}(i)$ and $e_{\min}(i)$ are the forecast error values $\{\tilde{1}, \tilde{2}, \tilde{3}, \tilde{4}\} \to \{1, 2, 3, 4\}$, $\{\sigma(1), \sigma(2), \sigma(3), \sigma(4)\} \to \{\tilde{1}, \tilde{2}, \tilde{3}, \tilde{4}\}$, They are all one-to-one correspondences.

3. ALGORITHM IMPLEMENTATION

3.1. Generation of the Authentication Watermark

An authentication watermark is generated for each 2×2 sub-block using the hash function {Hash} (block, row, col, key). Here, "block" refers to the pixel information within the current block, "row" and "col" indicate the location of the current block, and "key" is the agreed-upon key shared between the sender and receiver.

3.2. Embedding of Authentication Watermarks

The prediction error values dmax and dmin for each 2×2 embedding unit are computed based on the count of 0s and 1s across the entire sub-block to determine the embedding capacity of the unit. The embedding capacity of each unit is then evaluated. If the capacity of the embedding block is greater than or equal to 2, it is classified as a smoothing block; otherwise, it is classified as a texture block.

3.2.1. Smooth block embedding proces

At the embedding stage, the pre-sorting-IPVO (Improved Pixel Value Ordering) method is employed to embed the authentication code within the smooth blocks. At the authentication stage, the corresponding inverse operation is applied to extract and verify the embedded authentication code. The detailed process of watermark embedding is as follows:

S1: Begin by dividing the image into four main blocks, and further subdivide each main block into 2×2 sub-blocks.

S2: Calculate the prediction $\{\hat{x}_1, \hat{x}_2, \hat{x}_3, \hat{x}_4\}$ in the 2×2 subblock $\{x_1, x_2, x_3, x_4\}$

S3: Pre-sorting $\{x_1, x_2, x_3, x_4\}$ according to $\{\hat{x}_1, \hat{x}_2, \hat{x}_3, \hat{x}_4\}$ yields a pre-sorted sequence in ascending order of the original sequence $\{\tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \tilde{x}_4\}$. Pre-sorting $\{\tilde{x}_{\sigma(1)}, \tilde{x}_{\sigma(2)}, \tilde{x}_{\sigma(3)}, \tilde{x}_{\sigma(4)}\}$ in ascending order yields $\{\tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \tilde{x}_4\}$.

S4: Utilize a hash function Hash(block, row, col, key) to generate an authentication code for each 2×2 sub-block. Here, "block" includes the pixel data of the current block, "row" and "col" specify the block's position within the image, and "key" is a pre-established key shared between the sender and receiver.

S5: Using equations $\{\tilde{x}_{\sigma(1)}, \tilde{x}_{\sigma(2)}, \tilde{x}_{\sigma(3)}, \tilde{x}_{\sigma(4)}\}$, (18) and (19), perform data embedding based on the predicted error after pre-sorting. The embedding rule requires that larger values become larger or remain unchanged, and smaller values become smaller or remain unchanged. This ensures reversibility and maintains the original order of the pixel values.

$$\tilde{x}'_{\sigma(i)} = \begin{cases} \tilde{x}_{\sigma(i)} + s & if\ e_{\max}(i) = 0\ or\ e_{\max}(i) = 1 \\ \tilde{x}_{\sigma(i)} + 1 & if\ e_{\max}(i) < 0\ or\ e_{\max}(i) > 1 \end{cases} \tag{18}$$

$$\tilde{x}'_{\sigma(i)} = \begin{cases} \tilde{x}_{\sigma(i)} - s & if\ e_{\min}(i) = 0\ or\ e_{\min}(i) = 1 \\ \tilde{x}_{\sigma(i)} - 1 & if\ e_{\min}(i) < 0\ or\ e_{\min}(i) > 1 \end{cases} \tag{19}$$

Where $\tilde{x}'_{\sigma(1)}$ and $\tilde{x}'_{\sigma(2)}$ denote the pixel values after expansion, $e_{\max}(i)$ and $e_{\min}(i)$ denote the prediction errors corresponding to the modified pixels.

S6:Repeat steps S2 through S5 until the image is fully embedded with the watermark.

3.2.2. Texture Block Embedding Process

At the embedding stage, the authentication code is embedded into the differential block using the differential expansion method. At the authentication stage, the embedded authentication code is extracted and authenticated using the corresponding inverse operation. The detailed watermark embedding process is as follows:

S1: For the pixel pair in the 2 × 2 sub-block (X_i, X_{i+1}), calculate the mean m and difference d of the pixel pair according to Equations (20) and (21).

$$d = X_i - X_{i+1} \tag{20}$$

$$m = \lfloor (X_i + X_{i+1}) / 2 \rfloor \tag{21}$$

S2: The marked difference is obtained by expanding the difference d and embedding the watermark, as illustrated in Equation (22).

$$d' = 2 \times d + s \tag{22}$$

S3: Adjust the pixel values using Equation (23) to obtain the watermarked image.

$$\begin{cases} X_i' = m + \lfloor (d' + 1) / 2 \rfloor \\ X_{i+1}' = m - \lfloor d' / 2 \rfloor \end{cases} \tag{23}$$

S4: Continue repeating steps S1 through S3 until the watermarked image is achieved, as depicted in Figure 2.

(a) Original image (b) Image with watermark (c) Embedded watermark

Figure 2. Image example

3.3. Tampering Detection Process

The extraction of the authentication watermark is the reverse of the embedding process. If the embedding begins with black pixels, then the extraction should start with white pixels, and vice versa.

3.3.1. Extraction of Smooth Block Authentication Code

S1: Begin by dividing the image, which contains the embedded authentication watermark, into four main blocks. Then, further divide these main blocks into 2 × 2 sub-blocks. Organize the sub-blocks into two distinct sets based on a black and white checkerboard pattern: one set comprising black pixels and the other comprising white pixels.

S2: Take the white block as an example, and calculate the prediction error $\{\hat{x}_1, \hat{x}_2, \hat{x}_3, \hat{x}_4\}$ in the 2 × 2 sub-block $\{x_1, x_2, x_3, x_4\}$.

S3: Utilize the hash function Hash(block, row, col, key) to generate an authentication code for each 2 × 2 sub-block. Here, block contains information about the pixels within the current block, row and col represent the row and column indices of the current block, and key is a pre-agreed key shared between the sender and receiver.

S4: For each $\{\tilde{x}_{\sigma(1)}, \tilde{x}_{\sigma(2)}, \tilde{x}_{\sigma(3)}, \tilde{x}_{\sigma(4)}\}$ modified 2 × 2 sub-block, retrieve the pixel values. If the conditions outlined in Equation (24) are met, the authentication information can be successfully extracted.

$$s = \begin{cases} e'_{\min}(i) - 1 & if \ \ e'_{\min}(i) \in \{1, 2\} \\ -e_{\min}(i) & if \ \ e'_{\min}(i) \in \{-1, 0\} \end{cases} \tag{24}$$

S5: Extract the authentication information for each black pixel by reversing the watermark embedding process.

3.3.2. Extraction of Texture Block Authentication Code

S1: Segment the image into four main blocks, and then further divide these main blocks into 2 × 2 sub-blocks. Organize these sub-blocks into two distinct sets using a black and white checkerboard pattern: one set consisting of black pixels and the other of white pixels.

S2: Classify the image into smooth blocks and texture blocks as detailed in Section 2.2.

S3: Within the texture block, apply the inverse operation of difference expansion to the pixel sequence (X'_i, X'_{i+1}), as described in Equation (25), to restore the original pixel values

$$X'_i = X_i - 2 \times d + s \tag{25}$$

S4: Extract the authentication information using the same method as for black pixels, by reversing the process of watermark embedding.

3.3.3. Authentication Process

During the transmission of a watermarked image from the sender to the receiver, the security of the transmission channel cannot always be guaranteed. Therefore, the receiver must verify the integrity of the watermarked image upon arrival to ascertain whether it has been maliciously altered. In accordance with the watermark generation process outlined in Section 3.1, the regenerated authentication watermark b is compared with the extracted authentication watermark. If both are identical, it indicates that the watermarked image remains intact and unaltered. Conversely, if they differ, it suggests that the image has been tampered with. The specific tampered regions can be identified by examining the watermark pixel values that do not match.

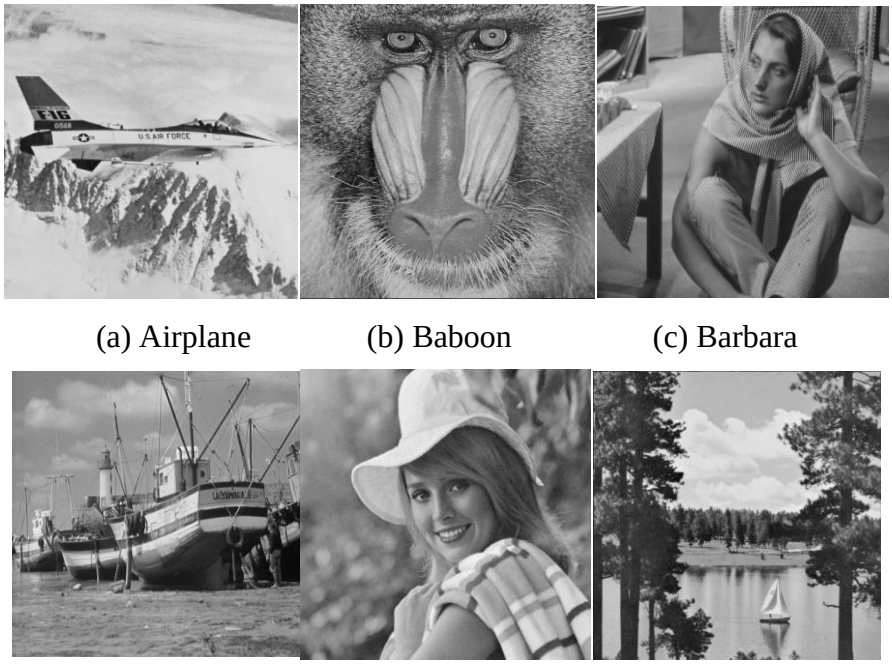
S1: As described in Section 3.2, segment the watermarked image into smooth blocks and texture blocks.

S2: As detailed in Section 3.3, extract the authentication watermark information b'_1 and b'_2 from the smooth blocks and texture blocks, respectively. Simultaneously, regenerate the watermark information b_1 and b_2 for both the smooth and texture blocks.

S3: Compare the extracted watermark information b' with the regenerated watermark information b. If $b_1 = b'_1$, $b_2 = b'_2$, the image is considered untampered; if they do not match, the image is deemed to have been tampered with.

4. EXPERIMENTAI RESULTS AND ANALYSIS

To validate the performance of the proposed algorithm, we utilized six standard 8-bit grayscale images, each with dimensions of 512 × 512 pixels. These images include Airplane, Baboon, Barbara, Boat, Elaine, and Lake, as depicted in Fig. 3. The experiments were conducted using MATLAB 2016a on a Windows 11 system. The authentication performance of the proposed method was assessed using the correct detection rate (CR). To highlight the effectiveness of our approach, we compared it with the methods proposed by Hong et al. [10] and Yao et al. [11], focusing on image quality and recognition accuracy.



(a) Airplane (b) Baboon (c) Barbara

(d) Boat (e) Elanie (f) Lake

Figure 3. Experimental image

4.1. Comparison of Watermark Image Quality

Table 1 presents the PSNR and SSIM values of the experimental images after embedding the authentication code. As shown, the PSNR values of the watermarked images using our method remain above 38 dB, while the SSIM values are consistently above 0.91. These results indicate that the visual quality of the watermarked images is maintained within an acceptable range, allowing for enhanced authentication performance.

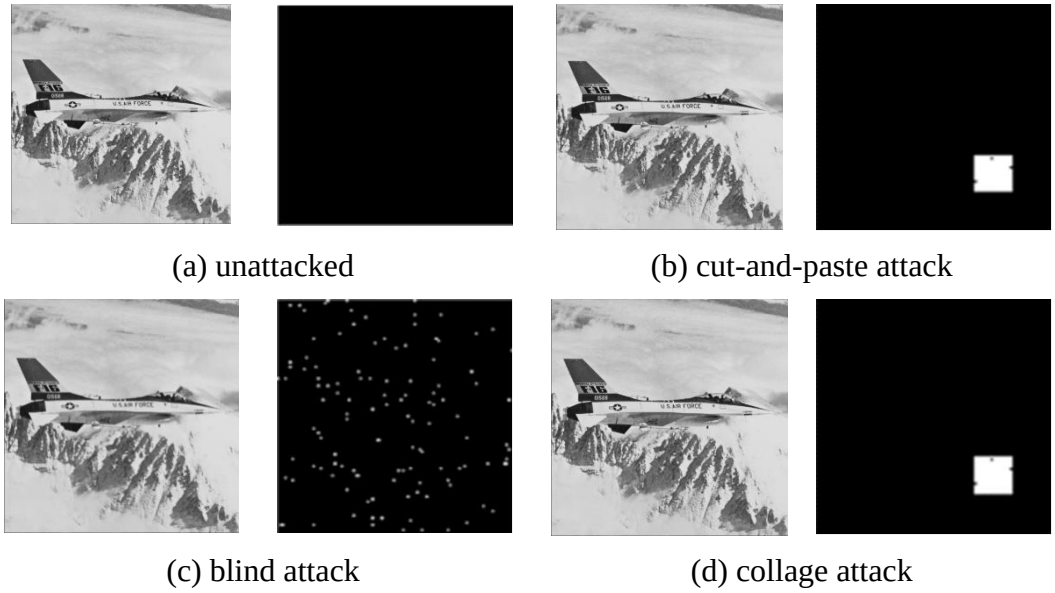
Table 1. Comparison between PSNR and SSIM

Image	Honget al		Yaoet al		This article's method	
	PSNR (dB)	SSIM	PSNR (dB)	SSIM	PSNR (dB)	SSIM
AirPlane	51.04	0.9964	48.84	0.9964	41.60	0.9112
Baboon	51.65	0.9934	-	-	43.23	0.9194
Barbara	50.26	-	49.49	0.9972	43.05	0.9823
Boat	49.93	0.9937	49.79	0.9963	42.03	0.9974
Average value	50.92	0.9954	49.45	0.9964	41.31	0.9726

4.2. Comparison of Correct Detection Rates

Figure 4 illustrates the tamper detection markers obtained under various scenarios: when the watermarked image is not attacked, and when it experiences cut-and-paste attacks, random tampering,

and collage attacks, using a 4×4 block as the tamper detection unit. Figure 5 displays the Airplane image both in its original form and as watermarked images following cut-and-paste, random, and collage attacks. The subsequent images showcase the initial detection markers generated by our method. Each point in the figure corresponds to a 4×4 sub-block.



(a) unattacked (b) cut-and-paste attack

(c) blind attack (d) collage attack

Figure 4. Initial Tamper-Detection Mark

Table 2 presents a comparison of the initial correct detection rates achieved using tamper detection units with sub-blocks of sizes 4×4, 8×8, and 16×16, respectively. It is evident that increasing the sub-block size enhances the initial correct detection rate. This improvement is primarily due to the larger sub-blocks having a greater capacity to embed more authentication information. Additionally, the reduction in the number of sub-blocks with limited capacity contributes to a higher detection rate.

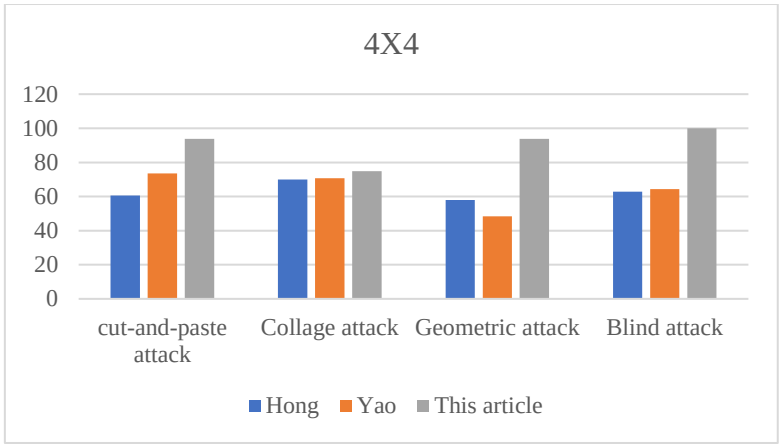
Table 2. Comparison of Correct Detection Rates (%) Between the Algorithm by Hong et al. and the Algorithm Proposed in This Paper.

tampering attack	Honget al			Yaoet al			This article's method		
	4×4	8×8	16×16	4×4	8×8	16×16	4×4	8×8	16×16
cut-and-paste attack	60.67	88.63	89.53	73.60	95.26	97.06	93.81	95.97	98.05
Collage attack	69.92	95.31	96.00	70.70	100.0	100.0	74.83	75.00	75.00
Geometric attack	58.02	82.50	90.00	48.47	57.50	60.00	93.81	95.97	98.05
Blind attack	62.87	88.81	91.84	64.26	84.25	85.68	100.0	100.0	100.0

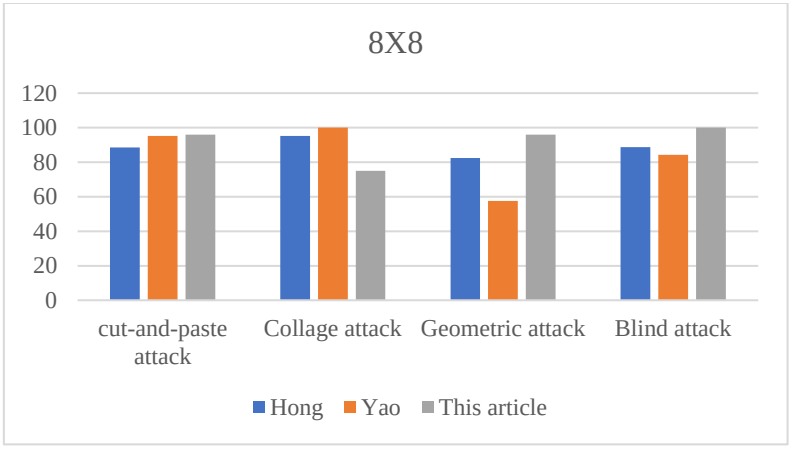
Figure 5 illustrates a comparison of the correct detection rates between the algorithms of Hong et al., Yao et al., and the algorithm proposed in this paper when subjected to clip attacks, random attacks, geometric attacks, and blind attacks, using 4×4, 8×8, and 16×16 blocks as tamper detection units. It is evident that the method introduced in this paper achieves a significantly higher correct detection rate against various attacks. This enhancement is attributed to the use of a pre-sorting method that rearranges the order of pixel collection within a block, thereby increasing the embedding capacity. Moreover, authentication information is embedded not only in smooth pixels but also in textured pixels, further boosting the correct detection rate.

Additionally, while both the Hong et al. and Yao et al. algorithms achieve correct detection rates of over 90% when all 16×16 sub-blocks are tampered with, the effectiveness diminishes as the sub-

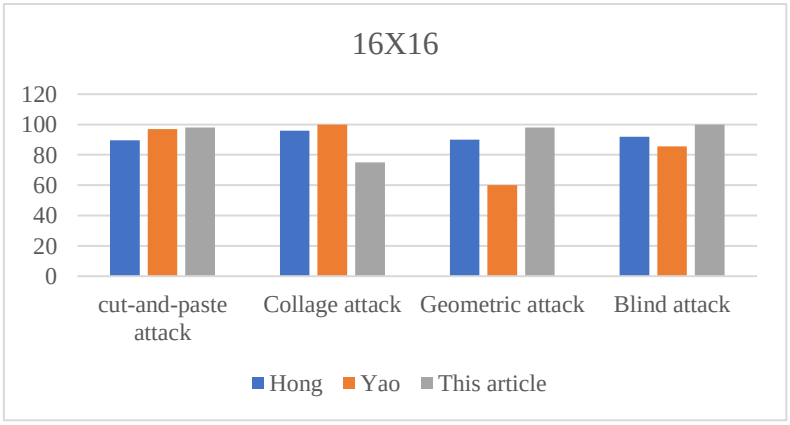
block size decreases, with lower tampering detection rates for 4×4 or 8×8 sub-blocks. This is primarily because the methods of Hong et al. and Yao et al. typically embed only 1-bit authentication codes per block, whereas the method described in this paper embeds at least 2-bit authentication codes per block, including in texture blocks. This reduces the likelihood that the authentication code extracted from a tampered block matches the regenerated authentication code, allowing for high tamper detection rates even with 4×4 or 8×8 block sizes.



(a) Comparison of the correct detection rates for 4×4 blocks



(b) Comparison of the correct detection rates for 8×8 blocks



(c) Comparison of the correct detection rates for 16 × 16 blocks

Figure 5. Comparison of the correct detection rates

5. SUMMARY

To enhance the accuracy of image tampering detection, this paper introduces a reversible image authentication method based on pre-sorting-IPVO. As an advancement over the traditional IPVO algorithm, this approach not only effectively detects image tampering but also significantly increases detection accuracy. Notably, the algorithm is adept at protecting images with complex textures or intricate textured regions while ensuring that the visual quality of the image remains high.

REFERENCES

[1] Zhang XP, Yin ZX. Data hiding in multimedia [J]. Chinese Journal of Nature, 2017, 39(2): 87–95.

[2] Li XL. A review on image reversible data hiding [J]. Journal of Information Security Research, 2016, 2(8): 729-734.

[3] Lee SK, Suh YH, Ho YS. Reversible image authentication based on watermarking [J]. In: Proc. of the 2006 IEEE Int'l Conf. on Multimedia and Expo. Toronto: IEEE, 2006. 1321–1324.

[4] Wang H, Huang FJ. Attack and improvement of an authentication scheme based on reversible data hiding [J]. Journal of Cyber Security, 2022, 7(1): 56-65 (in Chinese with English abstract).

[5] Lo CC, Hu YC. A novel reversible image authentication scheme for digital images [J]. Signal Processing, 2014, 98: 174–185.

[6] X. Li, J. Li, B. Li, et al. High-fidelity reversible data hiding scheme based on pixel-value-ordering and prediction-error expansion [J]. Signal Process, 2013, 93 (1): 198–205.

[7] F. Peng, X. Li, B. Yang. Improved pvo-based reversible data hiding [J]. Digital Signal Process, 2014, 25 (12): 255–265 .

[8] Lin, P. L., Hsieh, et al. A hierarchical digital watermarking method for image tamper detection and recovery [J]. Pattern Recognition, 2005, 38(12), 2519-2529.

[9] Wu HR, Li XL, Zhao Y, Ni RR. Improved PPVO-based high-fidelity reversible data hiding [J]. Signal Processing, 2020, 167: 107264.

[10] Fan GJ, Pan ZB, Gao ED, Gao XY, Zhang XR. Reversible data hiding method based on combiningIPVO with bias-addedrhombuspredictor by multi-predictor mechanism [J]. Signal Processing, 2021, 180: 107888.

[11] Yin ZX, Niu XJ, Zhou ZL, Tang J, Luo B. Improved reversible image authentication scheme [J]. Cognitive Computation, 2016, 8(5): 890–899.

[12] Hong, W., Chen, M., T. S., et al. An efficient reversible image authentication method using improved PVO and LSB substitution techniquesn [J]. Signal Processing: Image Communication, 2017, 58, 111-122.