Journal of Real-Time Image Processing (2020) 17:41–54
https://doi.org/10.1007/s11554-019-00904-8

SPECIAL ISSUE PAPER

A real-time reversible image authentication method using uniform embedding strategy

Heng Yao[1] · Hongbin Wei[2] · Chuan Qin[1] · Zhenjun Tang[3]

Received: 2 April 2019 / Accepted: 29 July 2019 / Published online: 5 August 2019

Abstract
Reversible image authentication (RIA) is an emerging research field for image tampering operation detection. Tampered regions can be localized precisely by embedding an authentication code (AC) into each divided image block in advance. Once the image is identified as an authentic image, the original image can be recovered without any loss. Under these two preconditions, an efficient RIA scheme is proposed to further improve the detection precision of the final authentication results. Compared with existing methods, a uniform embedding strategy is adopted in this paper, in which one AC bit is embedded into each divided image block to ensure they have the same authentication capability. To improve the forgery localization precision, the block size is adaptively sought according to the embedding capacity of the image. In addition, during the image authentication process, the embedding parameters and location map information are verified to increase the process's rigorousness. The experimental results demonstrate the superiority of the detection precision of the proposed method.

Keywords Image authentication · Tamper detection · Pixel value ordering · Reversible data hiding

1 Introduction

The development of Internet and cloud computing has brought a lot of convenience to our life [1]; however, privacy-preservation is constantly concerned in the current network environment [2, 3]. Image verification techniques are important for many multimedia applications. To identify the authentication of an image, there are two primary methods: active and passive. In an active method, the authentication code (AC) is embedded into the image in advance, and once an attacker alters the content of the image, the forgery operation can be verified by analyzing the extracted AC. This type of method is also referred to as a fragile watermarking technique [4–6]. The AC generation can apply some perceptual and robust image hashing schemes [7–9]. In a passive method, ACs are not embedded into host images in advance. Instead, forgery traces, such as periodical resampling statistical features [10], noise features [11, 12], computer generated image features [13], and JPEG blockiness artifacts [14, 15], can be identified through a statistical analysis on an image. This type of method is also referred to as a digital image forensic technique. Although passive methods can cover more practical circumstances, their detection rates are much lower than those of active methods. Therefore, to obtain satisfactory verification results, the development of active methods is indispensable.

Recently, as an active technique, reversible image authentication (RIA) has become a point of research interest. An RIA technique can be regarded as an advantageous combination of reversible data hiding (RDH) and fragile watermarking. Unlike ordinary fragile watermarking, however, which cannot recover the original image, the host image can be restored with RIA without any loss. Furthermore, unlike traditional RDH

✉ Chuan Qin
qin@usst.edu.cn

Heng Yao
hyao@usst.edu.cn

Hongbin Wei
Whb1350541826@gmail.com

Zhenjun Tang
zjtang@gxnu.edu.cn

[1] School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China

[2] School of Mechanical Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China

[3] Guangxi Key Lab of Multi-source Information Mining and Security, Guangxi Normal University, Guilin 541004, China

methods, RIA can precisely authenticate a host image with a moderate distortion. Because some ACs must be embedded into an image with a reversible manner, existing RIA methods have mainly employed RDH strategies. There are multiple methods to embed information into a host image as reversible, such as adaptive prediction-error expansion (PEE) [16, 17], pairwise PEE [18, 19], multiple histogram modification PEE [20], histogram shift [21], difference expansion [22, 23], and pixel value ordering (PVO) [24]. However, to authenticate each local region of an image, AC embedment should be implemented in blocks; therefore, most RIA methods use the PVO scheme to embed into each block independently. In 2014, Lo and Hu [25] proposed the first RIA method to embed ACs into each block with identical bits, in which an AC is randomly generated by a pseudo-random number generation algorithm. Unlike most PVO-based RIA methods, in 2016, Nguyen et al. [26] proposed an RIA method using an adaptive PEE strategy. Optimal blocks were sought through a local complexity analysis. Motivated by [27], Yin et al. [28] proposed an RIA method using a dynamic pixel block partition scheme to achieve lower distortion for AC embedding. Recently, Hong et al. [29] proposed a novel RIA method using improved PVO (IPVO) [30] and least significant bit (LSB) substitution techniques. Unlike previous methods, in [29], an AC is also embedded into blocks that are not available for reversible data embedding. More detailed introductions of [30] and [29] are presented in Sects. 2.1 and 2.2, respectively. Recently, Gao et al. [31] proposed a blind reversible authentication method using PEE and compressed sensing (CS) techniques. The tampered regions can be well restored based on the reconstructed DCT coefficients.

In this study, we propose an efficient RIA method using the uniform embedding strategy, in which ACs having the same amount of information are embedded into each block. This operation facilitates evaluating the authentication capability of each image. In addition, the optimal block size is sought according to the entire embedding capacity, and the block size in our method is usually lower than in existing methods, thereby resulting in more refined detection results.

The rest of this paper is organized as follows. Section 2 briefly reviews the related works of our method, including IPVO method and Hong et al.'s reversible authentication method. Section 3 presents the proposed method from the both sides of image embedding and authentication. Section 4 presents the experimental results, and Sect. 5 concludes this paper.

2 Related works

In this section, we briefly review related works by Peng et al. [30] and Hong et al. [29], who propose an improved PVO method for ordinary reversible data hiding and a state-of-the-art RIA method, respectively.

2.1 Peng et al.'s IPVO technique

In 2014, Peng et al. introduced an improved PVO method by modifying the difference between the highest (or lowest) and second highest (or second lowest) values. The major improvement over the conventional PVO method is the consideration of the location of those values. Suppose a grayscale image $\mathbf{I}$ is size $M \times N$. First, $\mathbf{I}$ is divided into equal-sized sub-blocks $\mathbf{B}_i$, $i = 1, 2, \ldots, N_B$, where N_B is the total number of sub-blocks, and the total pixels in each sub-block is P. Let $\{B_i^1, B_i^2, \ldots, B_i^P\}$ be the pixel values in the sub-block $\mathbf{B}_i$. The pixels are sorted in ascending order and the rearranged sequence is denoted as $\{B_i^{\sigma(1)}, B_i^{\sigma(2)}, \ldots, B_i^{\sigma(P)}\}$, where $\sigma(j)$ indicates the original location of B_i^j and $j = 1, 2, \ldots, P$. As presented in [30], the embedding process can be conducted along two directions: maximum and minimum. Both directions share similar principles. To embed bits along the maximum direction, first, the two highest values, $B_i^{\sigma(P-1)}$ and $B_i^{\sigma(P)}$, are sought and a unique difference metric between them is defined as

$$d_{\max} = B_i^u - B_i^v, \tag{1}$$

where u and v indicate lowest and highest values of $\sigma(P-1)$ and $\sigma(P)$, respectively. Next, the highest pixel value $B_i^{\sigma(P)}$ is modified as

$$B_i'^{\sigma(P)} = \begin{cases} B_i^{\sigma(P)} + m, & \text{if} \quad d_{\max} = 1 \text{ or } 0, \\ B_i^{\sigma(P)} + 1, & \text{otherwise,} \end{cases} \tag{2}$$

where m is the to-be-embedded message bit. Similarly, for embedding message bits along the minimum direction, first, the difference between the two lowest values, $B_i^{\sigma(1)}$ and $B_i^{\sigma(2)}$, is determined as

$$d_{\min} = B_i^s - B_i^t, \tag{3}$$

where s and t indicate the lowest and highest values of $\sigma(1)$ and $\sigma(2)$, respectively. The minimum value, $B_i^{\sigma(1)}$, is then extended as

$$B_i'^{\sigma(1)} = \begin{cases} B_i^{\sigma(P)} - m, & \text{if} \quad d_{\min} = 1\text{or } 0, \\ B_i^{\sigma(P)} - 1, & \text{otherwise.} \end{cases} \tag{4}$$

At this point, the message bits have been embedded into the cover pixels along both directions. During the message extraction and cover pixel recovery, because the numerical order of pixel values in the sorted order is unchanged, the values of $\{\sigma(1), \sigma(2), \ldots, \sigma(P)\}$ remain unchanged as well. In addition, the values of u, v, s, and t are consistent with their original values appearing in Eqs. (1) and (3). Let $\{B_i'^1, B_i'^2, \ldots, B_i'^P\}$ be the pixel values in the marked sub-block $\mathbf{B}_i'$ and let $\{B_i'^{\sigma(1)}, B_i'^{\sigma(2)}, \ldots, B_i'^{\sigma(P)}\}$ be its corresponding ascending sorted sequence. The difference between $B_i'^{\sigma(P-1)}$ and $B_i'^{\sigma(P)}$ (or $B_i'^{\sigma(1)}$ and $B_i'^{\sigma(2)}$) is calculated by

$$\begin{cases} d'_{\max} = B_i'^u - B_i'^v, \\ d'_{\min} = B_i'^s - B_i'^t. \end{cases} \tag{5}$$

Therefore, if $d'_{\max} = 1$ or 0 (or $d'_{\min} = 1$ or 0), a bit “0” is extracted; if $d'_{\max} = 2$ or -1 (or $d'_{\min} = 2$ or -1), a bit “1” is extracted. For other ranges of $d'_{\max}$ and $d'_{\min}$, message bits are not embedded in advance. After data extraction, the original maximum value $B_i^{\sigma(P)}$ (or minimum value $B_i^{\sigma(1)}$) can be recovered as

$$B_i^{\sigma(P)} = \begin{cases} B_i'^{\sigma(P)}, & \text{if } d'_{\max} = 1 \text{or } 0, \\ B_i'^{\sigma(P)} - 1, & \text{otherwise,} \end{cases} \tag{6}$$

and

$$B_i^{\sigma(1)} = \begin{cases} B_i'^{\sigma(1)}, & \text{if } d'_{\min} = 1 \text{ or } 0, \\ B_i'^{\sigma(1)} + 1, & \text{otherwise.} \end{cases} \tag{7}$$

It is worth noting that there are many works that promote the performance of IPVO, such as [32–36]. However, in our method, we merely use IPVO as an example, which can be substituted by any PVO-based method.

2.2 Hong et al.'s RIA method

Previous RIA methods [25, 28] mainly embedded ACs into blocks that have the reversible capability of embedding data, such as $d_{\max} = 1$ or 0 in Eq. (2) and $d_{\min} = 1$ or 0 in Eq. (4). For other blocks, extra authentication information was not embedded; thus, those unembeddable blocks (UBs) lack content protection. To overcome this deficiency, an improved image authentication method was recently proposed by Hong et al. [29] that embeds ACs in all blocks, both embeddable blocks (EBs) and UBs. First, image $\mathbf{I}$ was partitioned into 4×4 blocks, then each block was further partitioned into four 2×2 sub-blocks. For each sub-block, the embedding capacity was sought according to the IPVO method. Next, the total embedding capacity of each 4×4 block was determined, and all blocks were grouped into categories according to their capacities. Specifically, blocks with zero-bit capacity were defined as UBs and the remaining blocks, with one- to seven-bit capacity, were defined as EBs. One AC bit was extracted from the content features of each UB using a hash algorithm and embedded into the block with a basic LSB replacement method. Because the substituted LSB in the original UB was irreversible, it had to be transferred as auxiliary information (AI) to embed into the EBs. Suppose the capacity of the EB is U; to authenticate this block, $U - 1$ bits hash values were extracted from the block features to generate the authentication bits, and, meanwhile, to recover each UB, one LSB was concatenated with $U - 1$ authentication bits to generate the to-be-embedded U bits, which were then embedded into the block using the IPVO method.

In the image authentication phase, the to-be-authenticated image $\mathbf{I}'$ was divided into 4×4 blocks, which were then further divided into sub-blocks taking the same approach as in the embedding phase. The total embedding capacity of each block was then determined. If a block lacked capacity, the LSB of the first pixel was extracted as the AC, and if the block had embedded reversible bits, these could be extracted through the inverse IPVO method. Finally, all authentication codes were verified with the regenerated hash codes. If all codes matched the hash codes, the image was identified as authentic; however, if there was a discrepancy between codes and hash values, the forgery regions could be located through a differential analysis. If the image was proven to be authentic, to recover the host image, the original LSBs from the UBs could be extracted from the embedded bits of the EBs.

3 Proposed method

In Hong et al.'s method, we notice that the ACs embedded in the UBs and EBs are not balanced. Specifically, for UBs, merely one bit can be used for authentication, whereas for EBs, $U - 1$ bits are used. In other words, the verification capability of EBs is significantly greater than that of UBs. However, it is also worth noting that the local texture complexity of each block is usually inversely proportional to its embedding capacity. Paradoxically, in most cases, the complex regions with low authentication bit rates are more likely to suffer malicious tampering attacks. Therefore, to overcome this contradiction, we propose an improved RIA method with uniform authentication capabilities, in which the same number of authentication bits is embedded into each block for consistent content protection.

Similar to all the existing RIA methods, our improved method will be introduced in two procedures: (1) authentication code embedding, and (2) image authentication and host image recovery. Both procedures are presented separately as follows. For the sake of description clarity, the adopted notation of symbols in the proposed method is summarized in Table 1.

Table 1 Summary of the adopted notations in the proposed method

$\mathbf{I}/\mathbf{I}'$	Original host image/embedded image
M/N	Number of rows/columns in the host image
$\mathfrak{B}_0/\mathfrak{B}_1/\mathfrak{B}_2$	2×2 sub-blocks with the capacity of 0/1/2 bits
C_0	Preliminary sum of capacities of all sub-blocks
$N_0/N_1/N^*$	Preliminary/verified/ultimate division number for each block
N_{SUB}	Total number of divided sub-blocks
T_0	Number of blocks involved in the location map substitution
$\mathbf{A}_{\mathrm{LM}}$	Substituted LSBs of first-row blocks for saving location map
L_{LM}	Length of $\mathbf{A}_{\mathrm{LM}}$
C_1	Sum of capacities of all sub-blocks except for the first T_0 blocks
$\tilde{\mathbf{B}}_i/\tilde{\mathbf{B}}'_i$	Divided blocks in the phases of embedding/authentication
$\tilde{B}_i^k/\tilde{B}_i'^k$	kth element in $\tilde{\mathbf{B}}_i/\tilde{\mathbf{B}}'_i$
$\mathbf{A}_{\mathrm{LSB}}$	Substituted LSBs of $\tilde{B}_i^1$ of each UB
L_{LSB}	Length of $\mathbf{A}_{\mathrm{LSB}}$
$\mathbf{M}_{\mathbf{LM}}$	Compressed location map

3.1 Authentication code embedding

To embed authentication codes into images as reversible, the procedure consists of three steps: image partition and AC embedment for UBs and EBs.

Step 1: Partition image with an appropriate block pattern

Unlike [29], in our method, each image is divided into blocks using a dynamic partition strategy. The $M\times N$-size image $\mathbf{I}$ is first divided into non-overlapped, 2×2 sub-blocks. Next, we evaluate the embedding capacity of each sub-block using the IPVO method, as introduced in Sect. 2.1. According to the properties of IPVO, the probable embedding capacity of each sub-block is zero, one, and two bits, and each sub-block is thereby denoted as $\mathfrak{B}_0$, $\mathfrak{B}_1$, and $\mathfrak{B}_2$, respectively. For sub-blocks having an underflow or overflow, we treat their capacities as zero-bit. Then we use the capacity numbers to determine the preliminary capacity of the entire image, denoted as C_0. Based on C_0, the number of sub-blocks in one authentication block, denoted as N_0, can be preliminarily determined by

$$N_0 = \frac{N_{\mathrm{SUB}}}{C_0}, \tag{8}$$

where N_{SUB} is the total number of divided sub-blocks and $\lceil\cdot\rceil$ denotes the rounded-up operation. In our sample of 1366 uncompressed color image database (UCID) test images, the value of N_0 distributes from one to six, as shown in Fig. 1. Based on this observation, we design six authentication

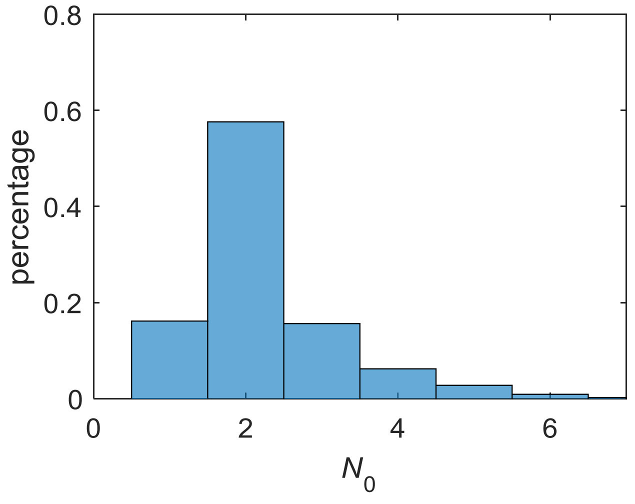


Fig. 1 Distribution of N_0 from UCID images

block patterns, as shown in Fig. 2, where (a)–(f) correspond to the block patterns with one to six sub-blocks, respectively. In some extreme cases, such as when N_0 is greater than 6, the corresponding authentication block patterns can be designed similarly.

However, the sub-block number N_0 is not the ultimate division number, due to underflows and overflows and AI substitution. In our scheme, the ultimate division number, denoted by N^*, is determined by comparing the preliminary number N_0 and its verified version N_1. Specifically, we first divide the image into blocks in accordance with the preliminary pattern, and then analyze the characteristics of each block with respect to underflows and overflows. Note that for each block, we evaluate the capacity of each sub-block and sum them to obtain the block's embedding capacity. Therefore, the possible capacity is 0, 1, 2, …, $(2\times N_0)$. It should be noted that for each block, if one sub-block has an underflow or overflow, the entire block is regarded as an issue block and the capacity set at zero. For blocks with a flow issue, we record their positions to generate an underflow/overflow location map, and compress this map using a lossless entropy method, such as arithmetic coding. After generating the compressed location map, we substitute the blocks with the LSBs of pre-defined first-row blocks. Suppose there are T_0 blocks involved in this substitution, and the substituted LSBs are denoted as $\mathbf{A}_{\mathrm{LM}}$ with length L_{LM}. Next, we determine the capacity of all blocks except the first-row T_0 blocks and denote it as C_1. Then the verified block division number N_1 can be determined by

$$N_1 = \frac{(N_{\mathrm{SUB}} - N_0\times T_0)}{(C_1 - L_{\mathrm{LM}})}. \tag{9}$$

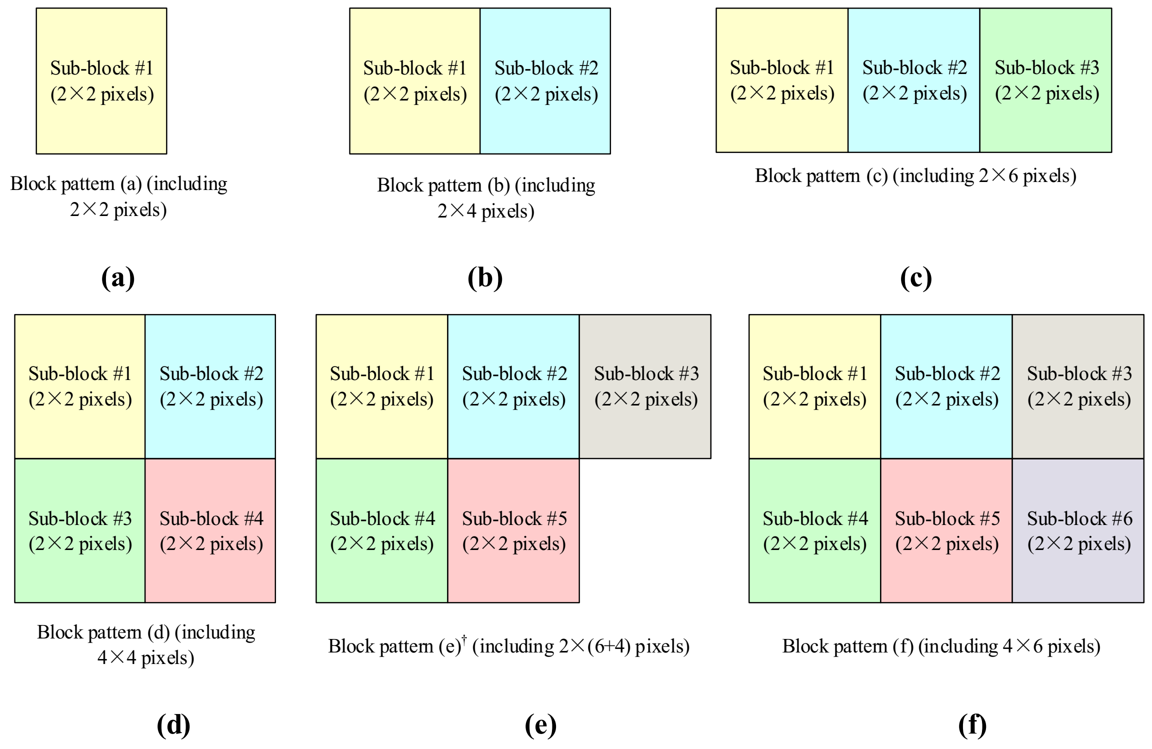

Sub-block #1 (2×2 pixels)
Block pattern (a) (including 2×2 pixels)
(a)
Sub-block #1 (2×2 pixels)
Sub-block #2 (2×2 pixels)
Block pattern (b) (including 2×4 pixels)
(b)
Sub-block #1 (2×2 pixels)
Sub-block #2 (2×2 pixels)
Sub-block #3 (2×2 pixels)
Block pattern (c) (including 2×6 pixels)
(c)
Sub-block #1 (2×2 pixels)
Sub-block #2 (2×2 pixels)
Sub-block #3 (2×2 pixels)
Sub-block #4 (2×2 pixels)
Block pattern (d) (including 4×4 pixels)
(d)
Sub-block #1 (2×2 pixels)
Sub-block #2 (2×2 pixels)
Sub-block #3 (2×2 pixels)
Sub-block #4 (2×2 pixels)
Sub-block #5 (2×2 pixels)
Block pattern (e)† (including 2×(6+4) pixels)
(e)
Sub-block #1 (2×2 pixels)
Sub-block #2 (2×2 pixels)
Sub-block #3 (2×2 pixels)
Sub-block #4 (2×2 pixels)
Sub-block #5 (2×2 pixels)
Sub-block #6 (2×2 pixels)
Block pattern (f) (including 4×6 pixels)
(f)

Fig. 2 Different partition patterns to be applied in the proposed method. (†For pattern (e), the right corner of the vacant block is filled up by another block which is a 180° rotation version of original pattern)

Under the circumstance that $N_1 = N_0$, N_0 is regarded as the ultimate verified division number N^*; otherwise, N_0 will increase by one until the re-verified number N_1 equals N_0, and in this case, the increased N_0 is regarded as N^*.

Step 2: Embed ACs into UBs

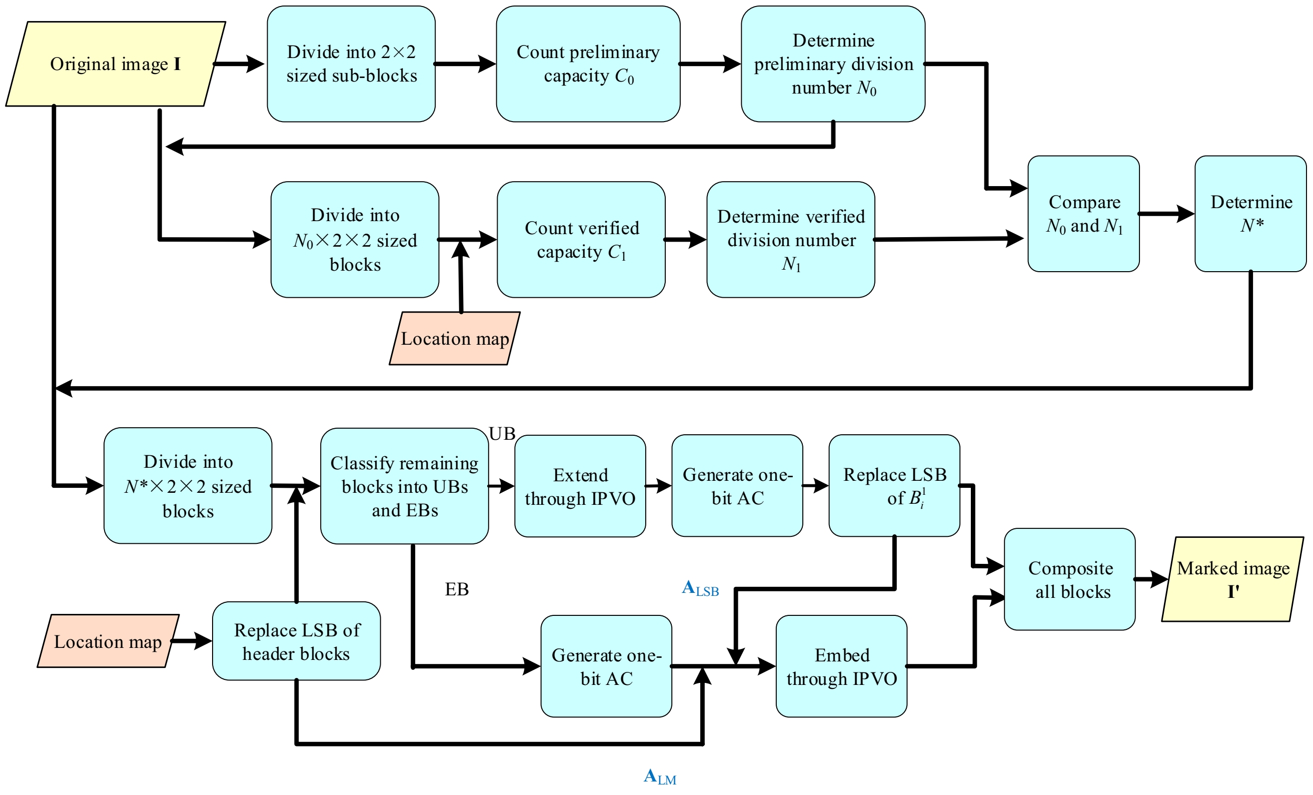
During the AC embedding procedure, the image is divided into blocks with the appropriate parameter N^*. There are some blocks with zero-bit capacity, in which each sub-block is invalid for embedding any information as reversible. However, in this study, we aim for each block to have a uniform authentication capability, in which each block embeds the same amount of AC information. To achieve our goal, we first embed ACs into UBs with an irreversible LSB replacement, and these replaced LSB values are then saved to embed into the UBs as reversible. To embed the ACs into the UBs, the maximum and minimum values of each sub-block in UBs are first extended through IPVO regulation. Denote the extended block as $\tilde{\mathbf{B}}_i$, and its corresponding pixels are $\left\{\tilde{B}_i^j, j = 1, 2, \ldots, 4 \times N^*\right\}$. Next, we hash the pixels $\{\tilde{B}_i^2, \tilde{B}_i^3, \ldots, \tilde{B}_i^{4\times N^*}\}$ (i.e., all pixels in the block except $\tilde{B}_i^1$) using MD5 to generate the 128-bit hash value, then use the parity of the number "1" in the hash sequence to indicate the AC: one if the number is odd, and zero if it is even. Next, the LSB of $\tilde{B}_i^1$ is replaced by the AC through a simple LSB replacement. Note that the replaced LSB values should be saved to be embedded to maintain the reversibility of the method. The saved set of LSBs of $\tilde{B}_i^1$ of all UBs is denoted as $\mathbf{A}_{\text{LSB}} = \left\{A_{\text{LSB}}^1, A_{\text{LSB}}^2, \ldots, A_{\text{LSB}}^{L_{\text{LSB}}}\right\}$, where L_{LSB} is the length of $\mathbf{A}_{\text{LSB}}$.

Step 3: Embed ACs and AI into EBs

For EBs, we embed a one-bit AC into the corresponding block. However, note that in addition to the AC information, there is AI that needs to be embedded, including $\mathbf{A}_{\text{LSB}}$ and $\mathbf{A}_{\text{LM}}$, which were generated during the UB procedure and location map embedding, respectively. Therefore, for a block whose capacity is greater than one, more AI bits are allocated to make the entire process completely reversible. The detailed embedding process was presented in Sect. 2.1. To facilitate understanding, a flow diagram of the embedding process is shown in Fig. 3, in which the image after AC embedding is denoted as $\mathbf{I}'$, and for ease of expression, the issue of underflow/overflow is ignored and discussed in the next section.

3.2 Underflow/overflow issue and its solution

For IPVO-based RDH methods, most maximum/minimum pixel pairs in sub-blocks are expanded by one with higher/lower trends. Hence, once the original pixel value is 255 or 0, there is a significant possibility that the value exceeds the



Original image **I**
Divide into 2×2 sized sub-blocks
Count preliminary capacity C_0
Determine preliminary division number N_0
Divide into N_0×2×2 sized blocks
Count verified capacity C_1
Determine verified division number N_1
Compare N_0 and N_1
Determine N^*
Location map
Divide into N^*×2×2 sized blocks
Classify remaining blocks into UBs and EBs
UB
Extend through IPVO
Generate one-bit AC
Replace LSB of B_i^1
Composite all blocks
Marked image **I'**
EB
$\mathbf{A}_{\text{LSB}}$
Location map
Replace LSB of header blocks
Generate one-bit AC
Embed through IPVO
$\mathbf{A}_{\text{LM}}$

Fig. 3 Flow diagram of AC embedding process

allowable bounds. To avoid this issue, pixel values of 255 or 0 remain unchanged and the location of the blocks they belong to are recorded at the same time to avoid confusion with pixel values of 254 and 1. It is worth noting that, as mentioned in [23], because LSB replacement operations are conducted during the AC embedding for the first pixel $\tilde{B}_i^1$ of UBs, some blocks that were originally UBs may be mistakenly judged as EBs. To overcome this issue, the embedding regulation of the first sub-block of each UB is changed to

$$\begin{cases} B_i^{\prime\sigma(P)} = B_i^{\sigma(P)} + 2, \\ B_i^{\prime\sigma(1)} = B_i^{\sigma(1)} - 2. \end{cases} \tag{10}$$

By doing so, pixel values of 254 or 1 may also cause an underflow/overflow phenomenon. Therefore, for the first sub-blocks of UBs, the pixel values of 0, 1, 254, and 255 are left unchanged and their corresponding positions of blocks they belong to are saved as a location map.

The location map is then compressed and denoted as $\mathbf{M}_{\text{LM}}$. Then we embed $\mathbf{M}_{\text{LM}}$ into the cover image **I** through LSB replacement in the first-row blocks. Note that the possibility still exists that the pixel values of first-row blocks have been tampered with, and once the $\mathbf{M}_{\text{LM}}$ is extracted incorrectly, the remaining authentication will also be affected. To indicate whether the header blocks have been tampered with during transmission, the first two LSBs of each header block are used to make room for authentication bits, which are derived from the hash sequence $\left\{\tilde{B}_i^3, \tilde{B}_i^4, \ldots, \tilde{B}_i^{4\times N^*}\right\}$. Figure 4 illustrates the LSB replacement of header blocks, where parameter N^* is 4 and the length of $\mathbf{M}_{\text{LM}}$ is 42 bits. On the authentication side, the two LSBs of each header block are extracted and compared with the regenerated ACs, which are derived from the hash sequence of the remaining pixels. When an inconsistency occurs in the header blocks, these blocks are marked as forgery blocks and the location map is no longer trustworthy. In other words, the authentication of the remaining blocks, whose pixels include the boundary values, is no longer reliable.

In addition, an example considering the issue of underflow/overflow is illustrated in Fig. 5 for further ease of comprehension of our method, where N^* is 2 in the example (i.e., each block contains two sub-blocks). For Block-A with pixels {64, 70, 26, 30, 42, 11, 124, 128}, whose capacity is 0, the first sub-block is first extended to {64, 72, 42, 9} according to Eq. (10), and the second sub-block is then extended to {25, 30, 124, 129} according to Eqs. (2) and (4). A MD5 hash computing is then conducted on {72, 25, 30, 42, 9, 124, 129} to generate a 128-bit hash sequence. Next the parity of the number of "1" in the hash sequence is extracted to represent the AC, which is 1 in the example. Therefore, the AC is embedded into the first pixel of 64 through an LSB replacement, and the final marked block is {65, 72, 42, 9, 25,

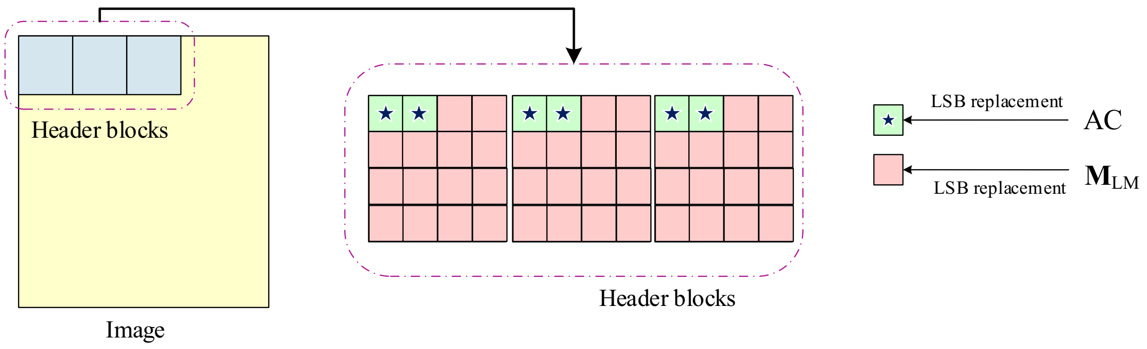


Fig. 4 Schematic diagram of the header blocks' LSB replacement

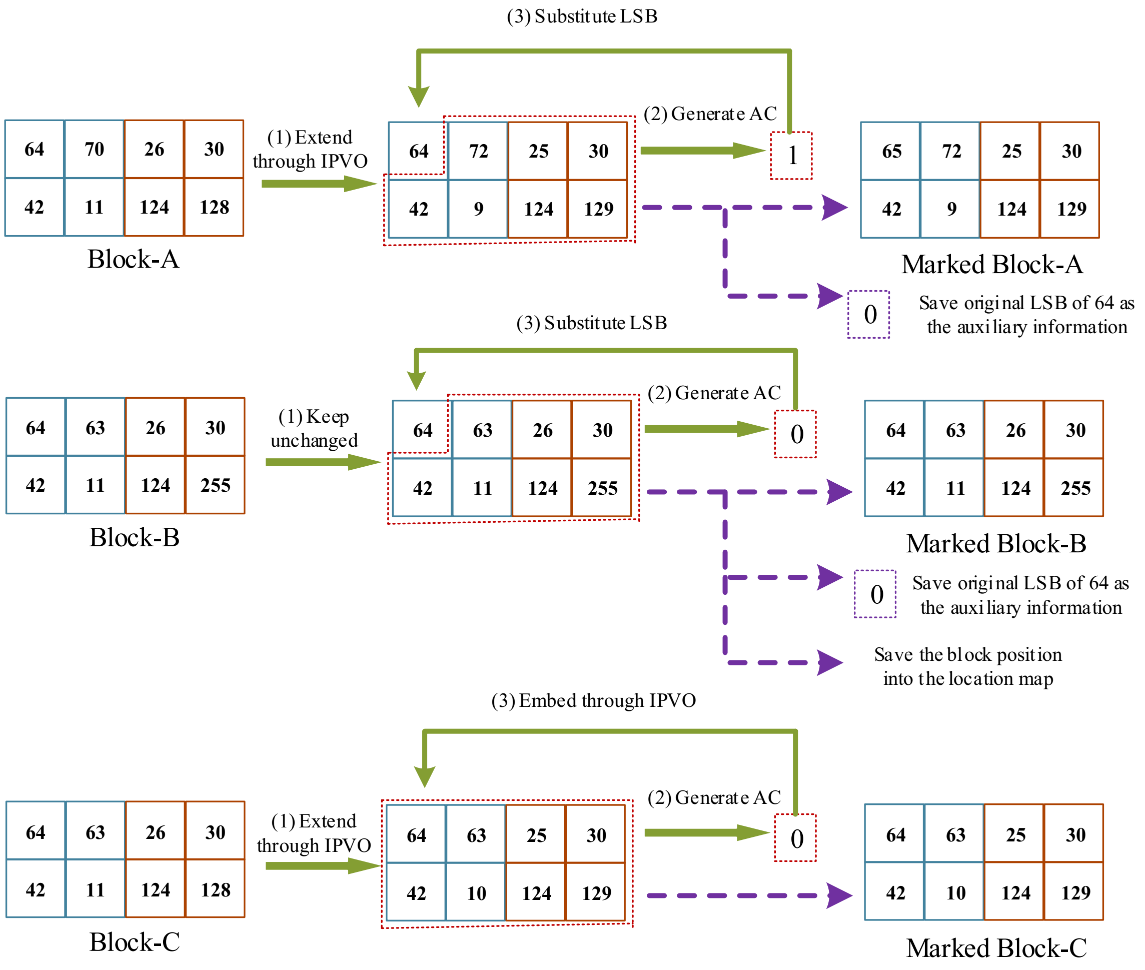


Fig. 5 Example of authentication code embedding considering the issue of underflow/overflow

30, 124, 129}. In addition, the replaced LSB of 64 is 0 and it is saved as auxiliary information for embedded into some EB. For Block-B with pixels {64, 63, 26, 30, 42, 11, 124, 255}, whose original capacity is 1, considering the overflow of the second sub-block, both sub-blocks in Block-B are kept unchanged and the capacity of the entire block is reset to 0. The AC of Block-B is then generated through a MD5 hashing on {63, 26, 30, 42, 11, 124, 255} and a parity count on "1" in the hash sequence, where AC is 0 in this example. Next, similar to Block-A, the AC is embedded into the first pixel value of 64 through an LSB replacement. Therefore, the final marked block is {64, 63, 26, 30, 42, 11, 124, 255}, and the replaced LSB value of 0 is saved as auxiliary information for embedding into some EB, and the position of Block-B is also recorded in the location map. For Block-C with pixels {64, 63, 26, 30, 42, 11, 124, 128}, whose capacity is 1 without the issue of underflow/overflow, the maximum and minimum pixels in each sub-block are first extended through IPVO. After extension, the AC of Block-C is generated based on the MD5 hashing on all extended pixels followed by a parity count, and AC is 0 is this example. Next, 0 is embedded into the sub-block of {64, 63, 42, 10} through an IPVO strategy according to Eq. (2). Therefore, the marked Block-C is {64, 63, 25, 30, 42, 10, 124, 129}.

3.3 Image authentication and host image recovery

During image authentication and host image recovery procedures, we suppose the to-be-authenticated image to be $\mathbf{I}'$. We extract the N^* information first and then divide the image into blocks with the corresponding pattern, as shown in Fig. 2. Next, we extract the $\mathbf{M}_{\mathrm{LM}}$ from the first-row header blocks. Specifically, we first extract the length of the location map and then extract the corresponding length bit for the location map recovery. After the location map $\mathbf{M}_{\mathrm{LM}}$ recovery, the blocks with an underflow/overflow issue are indicated and labeled as unchanged blocks. Next, according to the embedding capacity of each block, the remaining blocks are designated as UBs or EBs. To verify the authentication of the entire image, in the proposed method, we analyze the UBs followed by the EBs.

For a UB $\tilde{\mathbf{B}}'_i$, which consists of the pixels $\left\{\tilde{B}'^1_i, \tilde{B}'^2_i, \ldots, \tilde{B}'^{4\times N^*}_i\right\}$, the current AC is generated by applying the MD5 method to hash the pixels $\left\{\tilde{B}'^2_i, \tilde{B}'^3_i, \ldots, \tilde{B}'^{4\times N^*}_i\right\}$ and counting the parity of the number "1" in the hash sequence, which follows the same method of AC generation during the data-hiding process. Meanwhile, the original AC can be obtained by extracting the LSB of $\tilde{B}'^1_i$. The authentication of the UB $\tilde{\mathbf{B}}'_i$ can be identified by comparing the current and original ACs. If the two fall out of consistency with each other, the block is identified as a forgery block.

After verifying the UBs, the EBs are used to further verify the authentication of the image. For each EB $\tilde{\mathbf{B}}'_i$, the current AC is generated by applying the MD5 method to hash the pixels $\left\{\tilde{B}^1_i, \tilde{B}^2_i, \ldots, \tilde{B}^{4\times N^*}_i\right\}$ and by counting the parity of the number "1" in the hash sequence. The original AC can be extracted using a reverse IPVO, which was introduced in Sect. 2.1. Note that each EB can be further divided into N^* 2×2 sub-blocks, and at least one bit can be extracted from all the sub-blocks. Here, we regard the first extracted bit as the original AC of the block. Similar to UBs, the authentication of the EB $\tilde{\mathbf{B}}_i$ can be identified by comparing the current and original ACs.

When all authentication procedures are finished, if there are some discontiguous tampered regions, they can be regarded as an interim detection result. To further refine the detection result, following [29], we implement the principle that if the horizontal, vertical, or two diagonal neighboring blocks of a block are verified as forgeries, it is re-verified as a forgery, as well.

If all current and original ACs coincide with each other, we identify the image as authentic. To recover the host image $\mathbf{I}$, the EBs and UBs can be restored successively. To restore each EB, after dividing each block into N^* sub-blocks, the embedded bits are extracted based on the IPVO regulation. If the capacity of each block is one, the EB can be restored according to Eqs. (6) and (7), directly. Otherwise, the first extracted bit is used for authentication verification, and the remaining bits belong to the AI which originates from the other UBs. Note that during the AI extraction, the total length of extra bits is also recorded, and all the extra bits are saved as AI until the total length reaches the AI amount. Then each EB can be restored according to Eqs. (6) and (7). After restoring all the EBs, the saved AI is separated as $\mathbf{A}_{\mathrm{LSB}}$ and $\mathbf{A}_{\mathrm{LM}}$, and the UBs can be restored directly by an LSB replacement from $\mathbf{A}_{\mathrm{LSB}}$. Note that the header blocks used for saving $\mathbf{M}_{\mathrm{LM}}$ are not involved in the restoration of EBs and UBs and can be recovered by an LSB replacement from $\mathbf{A}_{\mathrm{LM}}$.

4 Experimental results

To demonstrate the efficacy of our method, we first conduct our experiments on several standard test images. Figure 6 shows six examples of the images we tested, which are Lena, Boat, Barbara, Peppers, Goldhill, and Airplane. In this study, the peak signal-to-noise ratio (PSNR) is applied to evaluate the quality of the marked image. Table 2 lists the PSNR values of each test image and they are compared with the state-of-the-art methods proposed by Hong et al. [29] and Yin et al. [28]. Observed from Table 2, method [28] performs the best results on the aspect of image quality, and the PSNR values of both methods of ours and [29] are around 1.5 dB lower than that of [28]. This is because the embedding strategy designed in [28] is only conducted in EBs; while no extra AC bit is embedded into UBs. Therefore, in [29] and ours, more distortion is inevitable due to the embedment of AC bits in UBs and AI bits in EBs. Although the PSNR performance is not remarkable, it is worth pointing out that the detection block size of our method is sought optimally (i.e., the parameter N^* is set as low as possible to realize a more refined detection resolution). In [28, 29], however, the corresponding division parameter N^* is constantly set at 4. Table 3 lists the comparisons of the minimum detection block size between the proposed method, [28, 29]. Table 3 indicates that the minimum detection sizes of the proposed method on the six images are all lower than those of [28,

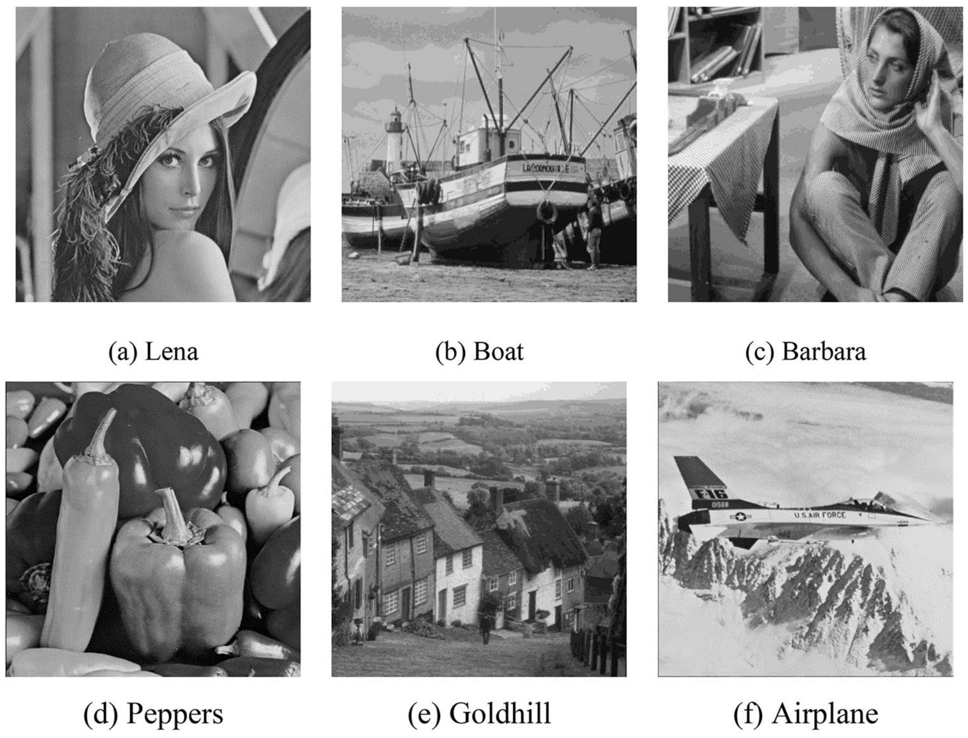


Fig. 6 Six standard test images

Table 2 The PSNR values of test images for the proposed method, Hong et al.'s method [29], and Yin et al.'s method [28] (dB)

	Lena	Boat	Barbara	Peppers	Goldhill	Airplane	Average
Ours	49.92	49.93	50.26	50.87	50.44	50.70	50.35
Hong et al. [29]	50.48	49.79	49.49	50.24	49.95	50.88	50.26
Yin et al. [28]	**51.83**	**51.61**	**51.84**	**51.68**	**51.92**	**52.11**	**51.83**

The best result for each image is shown by boldface

Table 3 The minimum detection block size of the proposed method, Hong et al.'s method [29], and Yin et al.'s method [28] (pixels)

	Lena	Boat	Barbara	Peppers	Goldhill	Airplane
Ours	2×4	2×4	2×6	2×6	2×6	2×4
Hong et al. [29]	4×4	4×4	4×4	4×4	4×4	4×4
Yin et al. [28]	4×4	4×4	4×4	4×4	4×4	4×4

Table 4 The PSNR values of test images for the proposed method with a fixed parameter ($N^* = 4$) (dB)

	Lena	Boat	Barbara	Peppers	Goldhill	Airplane	Average
Ours	**54.66**	**54.66**	**53.86**	**54.40**	**56.08**	**55.37**	**54.84**
Hong et al. [29]	50.56	50.44	49.49	50.34	49.95	50.88	50.26
Yin et al. [28]	51.83	51.61	51.84	51.68	51.92	52.11	51.83

The best result for each image is shown by boldface

29]. Furthermore, to make a fair comparison, Table 4 lists the PSNR values of our method if N^* is fixed at 4. In this case, the average PSNR value of our method improves to 54.84 dB.

The authentication capability of the proposed method is demonstrated in the left-most column of Fig. 7 which shows four problematic images that have been tampered with by splicing one part of an image into another image. We extracted the embedded ACs and compared them with the hash values generated by the image content. The interim detection results and further refined results are shown in the middle and right-most columns of Fig. 7, respectively,

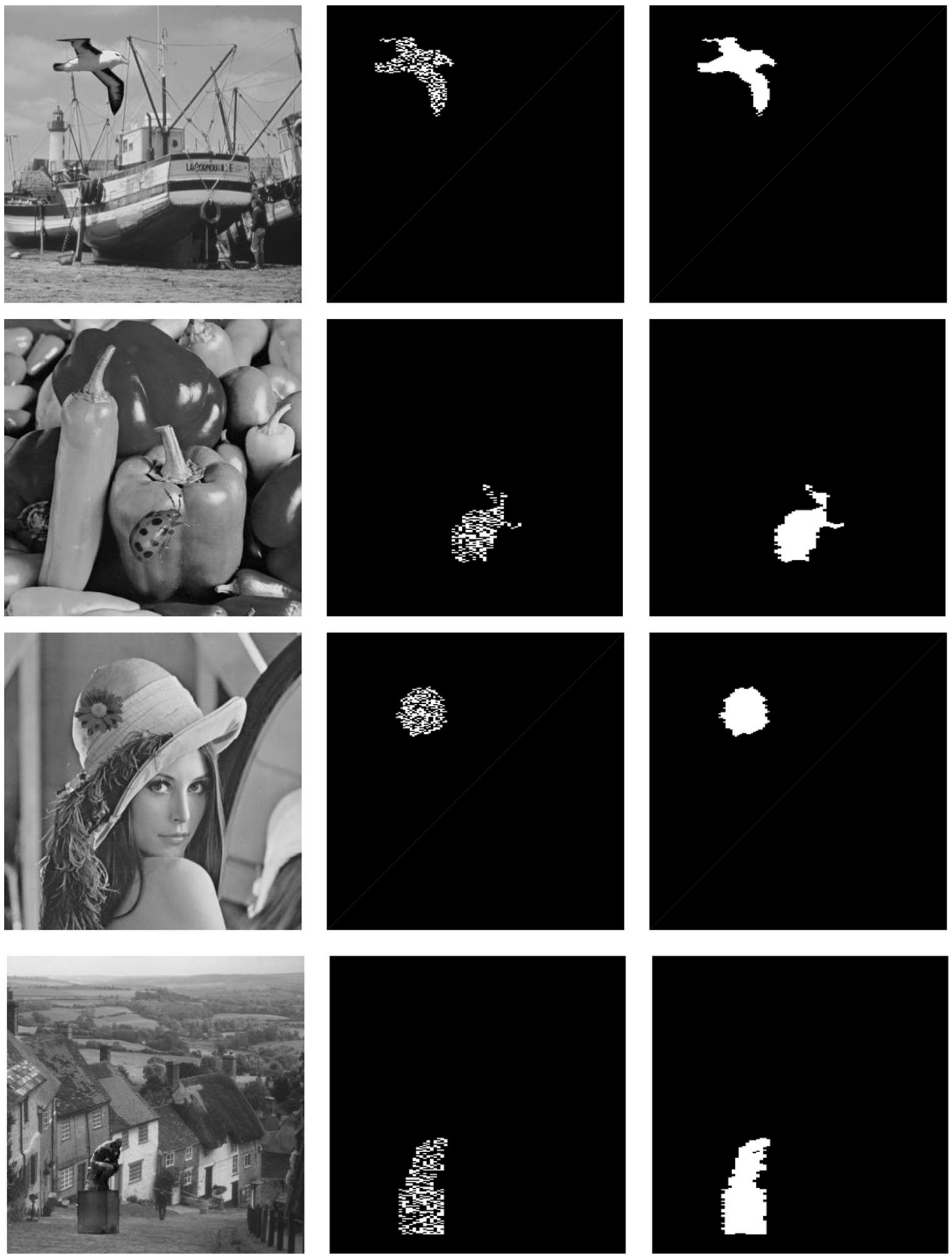


Fig. 7 The authentication results of the proposed method. The left column: forgery images. The middle column: the interim authentication results. The right column: the refined authentication results

where the white areas indicate the detected tampered regions. Compared with the real tampered regions, the proposed method can effectively detect and localize the splicing manipulations.

To evaluate the detection precision of the proposed method, we selected four test images and spliced some partial regions with external resources as shown in the left-most column of Fig. 8. The refined results of our method and [29] are shown in the second and third columns of Fig. 8, respectively. To make the comparison clearer, the corresponding difference maps between the detection results and the actual splicing maps are shown in the right-most two columns of Fig. 8, in which the fourth and fifth columns correspond to the proposed method and [29], respectively. In addition, we use the detection error percentage η to indicate the detection accuracy of the proposed method, where η is defined as

$$\eta = \frac{E}{M \times N} \times 100\%, \tag{11}$$

where E denotes the number of error-detected pixels. Table 5 shows a comparison of the detection percentages of the images from Fig. 8 between the proposed method and [29]. Figure 8 and Table 5 indicate that the proposed method and [29] both have a high detection accuracy for most circumstances; however, the proposed method has better localization capabilities in the splicing border regions. This is

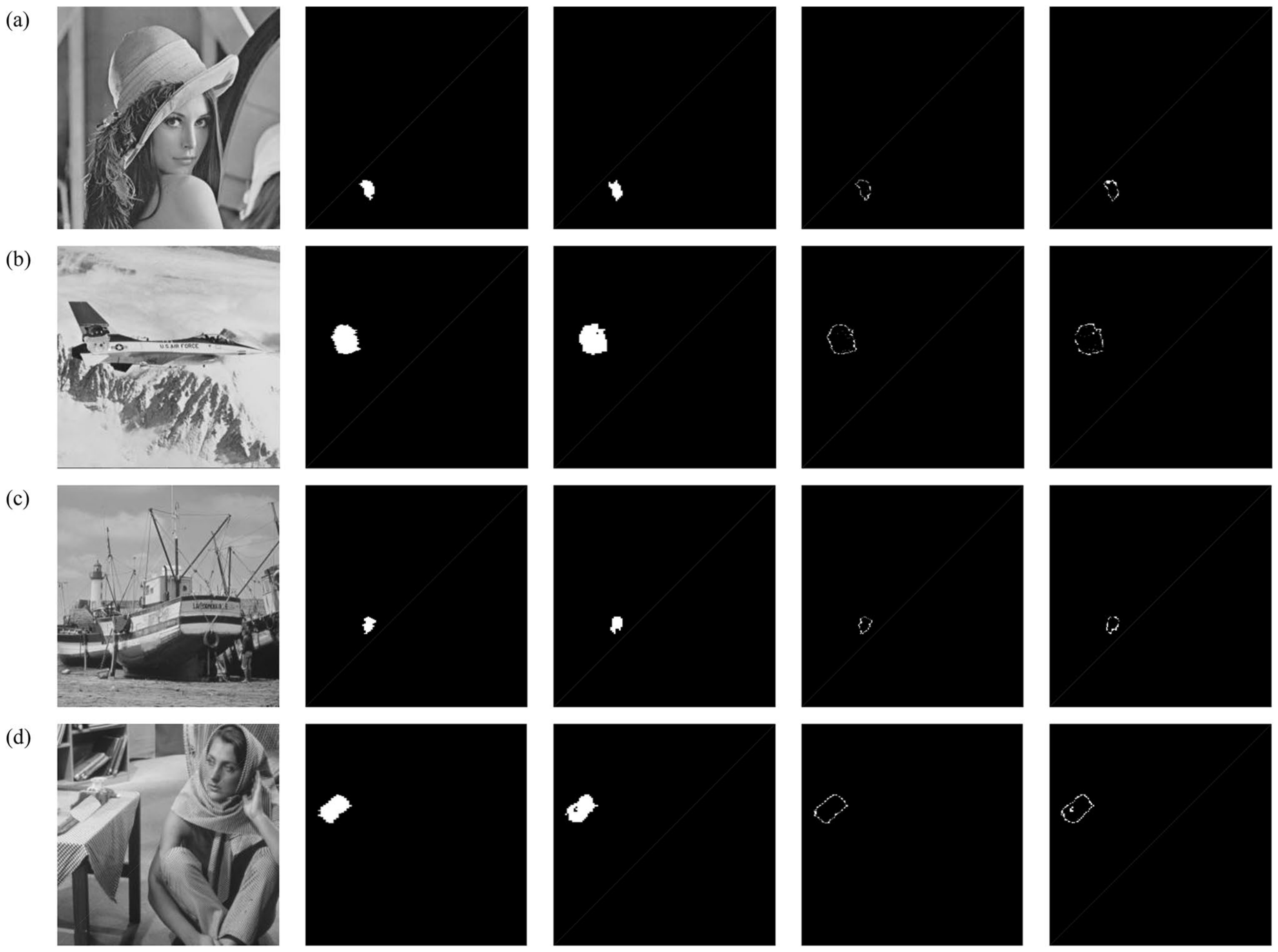


Fig. 8 Comparisons of detection results of the test images. The first column: spliced images; second and third columns: detection results of the proposed method and [29], respectively; fourth and fifth columns: the error detection results of the proposed method and [29], respectively

Table 5 Comparisons of the detection error percentage η for the proposed method and [29] (%)

	Figure 8a	Figure 8b	Figure 8c	Figure 8d	Average
Ours	**0.06**	**0.13**	**0.06**	**0.14**	**0.10**
Hong et al. [29]	0.09	0.14	0.08	0.18	0.12

The best result for each image is shown by boldface

perhaps because a dynamic selection strategy using block size is applied in the proposed method.

For real-time efficiency, Table 6 shows the execution time of the proposed method on six test images during the embedding and authentication procedures. All the experiments were implemented with the Matlab platform and tested on a PC with Intel Core i5-8300H 2.3 GHz CPU and 24 GB memory; note that our program code is not optimized.

Table 6 Execution time of the proposed method on six standard images

	Lena	Boat	Barbara	Peppers	Goldhill	Airplane	Average
Embedding process time (s)	12.76	12.58	11.25	14.53	10.52	12.76	12.40
Authentication process time (s)	17.96	19.17	14.20	14.66	13.22	18.11	16.22

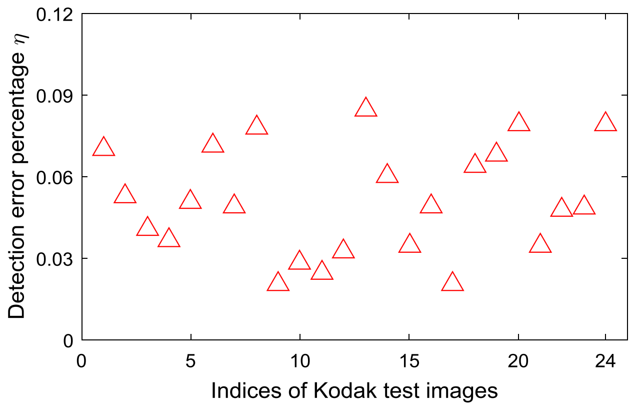


Fig. 9 Performance of detection error percentage on Kodak standard test images

Compared with [29], during the embedding process, the optimal block size is sought with an updated procedure, which takes a significant portion of the total time, and the authentication of the embedded location map is also performed to further improve the preciseness of the entire procedure. However, the execution complexity of the proposed method is sufficient to deal with most image authentication circumstances.

In addition, to demonstrate the general applicability of the proposed method, we conducted our strategy on Kodak test image dataset, which contains 24 standard test images. For each test image, we first embedded the ACs into all UBs and EBs sequentially and then randomly selected a 32×32 block from each marked image followed by a same-size block replacement for tampering. Figure 9 shows the corresponding detection error percentages of all test images according to Eq. (11), and the average detection error percentage is around 0.05%, which maintains at a satisfactory level.

5 Conclusions

In this study, an efficient reversible image authentication method is proposed to improve forgery detection precision. First, the block size is sought according to the embedding capacity of the entire image, and the image is divided into non-overlapping blocks. Each block is then classified into two categories: unembeddable and embeddable. Next, considering the issue of underflow and overflow, the optimal block size is ultimately determined through a verification and updating process. For each unembeddable and embeddable block, the corresponding one AC bit is embedded via a direct LSB replacement and IPVO data-hiding process, respectively. The experimental results demonstrate the efficacy of the proposed method and the detection results show that it has a more refined forgery localization ability than the state-of-the-art method, while keeping satisfactory visual quality. However, it should be pointed out that the capability of our method is limited to forgery localization, and for damaged region restoration, our method is invalid. An efficient RIA with the ability to restore images is our future research direction. In addition, to better meet the actual application requirements, an extension from the spatial to JPEG domain is worth future consideration.

Acknowledgements We would like to thank Drs Wien Hong and Zhaoxia Yin for kindly sharing the codes of their works with us. This work was supported in part by the National Natural Science Foundation of China (61702332, 61672354, 61562007) and Guangxi Natural Science Foundation (2017GXNSFAA198222). The authors would like to thank the anonymous reviewers for their valuable suggestions.

References

1. Zhou, Z., Wu, Q.M.J., Sun, X.: Multiple distances-based coding: toward scalable feature matching for large-scale web image search. IEEE Trans. Big Data (2019). https://doi.org/10.1109/tbdata.2019.2919570)
2. Qi, L., Wang, R., Li, S., He, Q., Xu, X., Hu, C.: Time-aware distributed service recommendation with privacy-preservation. Inf. Sci. **480**, 354–364 (2019)
3. Qi, L., Zhang, X., Dou, W., Hu, C., Yang, C., Chen, J.: A two-stage locality-sensitive hashing based approach for privacy-preserving mobile service recommendation in cross-platform edge environment. Futur. Gener. Comp. Syst. **88**, 636–643 (2018)
4. Qin, C., Ji, P., Zhang, X.P., Dong, J., Wang, J.W.: Fragile image watermarking with pixel-wise recovery based on overlapping embedding strategy. Signal Process. **138**, 280–293 (2017)
5. Qin, C., Ji, P., Chang, C.C., Dong, J., Sun, X.M.: Non-uniform watermark sharing based on optimal iterative BTC for image tampering recovery. IEEE Multimed. **25**(3), 36–48 (2018)
6. 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. **73**, 83–92 (2018)
7. Qin, C., Chen, X., Ye, D., Wang, J., Sun, X.: A novel image hashing scheme with perceptual robustness using block truncation coding. Inf. Sci. **361**, 84–99 (2016)
8. Qin, C., Hu, Y., Yao, H., Duan, X., Gao, L.: Perceptual image hashing based on a Weber local binary pattern and color angle representation. IEEE Access. **7**, 45460–45471 (2019)
9. Tang, Z., Chen, L., Zhang, X., Zhang, S.: Robust image hashing with tensor decomposition. IEEE Trans. Knowl. Data Eng. **31**(3), 549–560 (2019)
10. Qiao, T., Shi, R., Luo, X.Y., Xu, M., Zheng, N., Wu, Y.M.: Statistical model-based detector via texture weight map: application in re-sampling authentication. IEEE Trans. Multimed. **21**(5), 1077–1092 (2019)
11. Yao, H., Wang, S.Z., Zhang, X.P., Qin, C., Wang, J.W.: Detecting image splicing based on noise level inconsistency. Multimed. Tools Appl. **76**(10), 12457–12479 (2017)
12. Yao, H., Cao, F., Tang, Z.J., Wang, J.W., Qiao, T.: Expose noise level inconsistency incorporating the inhomogeneity scoring strategy. Multimed. Tools Appl. **77**(14), 18139–18161 (2018)
13. Wang, J., Li, T., Li, T., Luo, X., Shi, Y.Q., Jha, S.K.: Identifying computer generated images based on quaternion central moments in color quaternion wavelet domain. IEEE Trans. Circuits Syst. Video Technol. (2018). https://doi.org/10.1109/tcsvt.2018.2867786

14. Wang, J., Wang, H., Li, J., Luo, X., Shi, Y.Q., Jha, S.K.: Detecting double JPEG compressed color images with the same quantization matrix in spherical coordinate. IEEE Trans. Circuits Syst. Video Technol. (2019). https://doi.org/10.1109/tcsvt.2019.2922309
15. Iakovidou, C., Zampoglou, M., Papadopoulos, S., Kompatsiaris, Y.: Content-aware detection of JPEG grid inconsistencies for intuitive image forensics. J. Vis. Commun. Image Represent. **54**, 155–170 (2018)
16. Yao, H., Qin, C., Tang, Z.J., Tian, Y.: Guided filtering based color image reversible data hiding. J. Vis. Commun. Image Represent. **43**, 152–163 (2017)
17. Weng, S.W., Pan, J.S., Li, L.D.: Reversible data hiding based on an adaptive pixel-embedding strategy and two-layer embedding. Inf. Sci. **369**, 144–159 (2016)
18. Ou, B., Li, X.L., Zhao, Y., Ni, R.R., Shi, Y.Q.: Pairwise prediction-error expansion for efficient reversible data hiding. IEEE Trans. Image Process. **22**(12), 5010–5021 (2013)
19. Ou, B., Li, X.L., Wang, J.W., Peng, F.: High-fidelity reversible data hiding based on geodesic path and pairwise prediction-error expansion. Neurocomputing. **226**, 23–34 (2017)
20. Li, X.L., Zhang, W.M., Gui, X.L., Yang, B.: Efficient reversible data hiding based on multiple histograms modification. IEEE Trans. Inf. Forensic Secur. **10**(9), 2016–2027 (2015)
21. Qin, C., Chang, C.C., Huang, Y.H., Liao, L.T.: An inpainting-assisted reversible steganographic scheme using a histogram shifting mechanism. IEEE Trans. Circuits Syst. Video Technol. **23**(7), 1109–1118 (2013)
22. Dragoi, I.C., Coltuc, D.: Local-prediction-based difference expansion reversible watermarking. IEEE Trans. Image Process. **23**(4), 1779–1790 (2014)
23. Coltuc, D.: Low distortion transform for reversible watermarking. IEEE Trans. Image Process. **21**(1), 412–417 (2012)
24. Li, X.L., Li, J., Li, B., Yang, B.: High-fidelity reversible data hiding scheme based on pixel-value-ordering and prediction-error expansion. Signal Process. **93**(1), 198–205 (2013)
25. Lo, C.C., Hu, Y.C.: A novel reversible image authentication scheme for digital images. Signal Process. **98**, 174–185 (2014)
26. Nguyen, T.S., Chang, C.C., Shih, T.H.: A high-quality reversible image authentication scheme based on adaptive PEE for digital images. KSII Trans. Internet Inf. Syst. **10**(1), 395–413 (2016)
27. Wang, X., Ding, J., Pei, Q.Q.: A novel reversible image data hiding scheme based on pixel value ordering and dynamic pixel block partition. Inf. Sci. **310**, 16–35 (2015)
28. Yin, Z.X., Niu, X.J., Zhou, Z.L., Tang, J., Luo, B.: Improved reversible image authentication scheme. Cogn. Comput. **8**(5), 890–899 (2016)
29. Hong, W.E., Chen, M.J., Chen, T.S.: An efficient reversible image authentication method using improved PVO and LSB substitution techniques. Signal Process. Image Commun. **58**, 111–122 (2017)
30. Peng, F., Li, X.L., Yang, B.: Improved PVO-based reversible data hiding. Digit. Signal Prog. **25**, 255–265 (2014)
31. Gao, G., Cui, Z., Zhou, C.: Blind reversible authentication based on PEE and CS reconstruction. IEEE Signal Process. Lett. **25**(7), 1099–1103 (2018)
32. Ou, B., Li, X.L., Wang, J.W.: Improved PVO-based reversible data hiding: a new implementation based on multiple histograms modification. J. Vis. Commun. Image Represent. **38**, 328–339 (2016)
33. Ou, B., Li, X.L., Wang, J.W.: High-fidelity reversible data hiding based on pixel-value-ordering and pairwise prediction-error expansion. J. Vis. Commun. Image Represent. **39**, 12–23 (2016)
34. He, W.G., Cai, J., Zhou, K., Xiong, G.Q.: Efficient PVO-based reversible data hiding using multistage blocking and prediction accuracy matrix. J. Vis. Commun. Image Represent. **46**, 58–69 (2017)
35. He, W.G., Xiong, G.Q., Weng, S.W., Cai, Z.C., Wang, Y.M.: Reversible data hiding using multi-pass pixel-value-ordering and pairwise prediction-error expansion. Inf. Sci. **467**, 784–799 (2018)
36. Lee, C.F., Shen, J.J., Kao, Y.C., Agrawal, S.: Overlapping pixel value ordering predictor for high-capacity reversible data hiding. J Real-Time Image Proc (2019). https://doi.org/10.1007/s11554-019-00872-z

Heng Yao received the B. Sc. degree from Hefei University of Technology, China, in 2004, the M. Eng. degree from Shanghai Normal University, China, in 2008, and the Ph. D. degree in signal and information processing from Shanghai University, China, in 2012. Since 2012, he has been with the faculty of the School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, where he is currently an Associate Professor. His research interests include multimedia security, image processing, and pattern recognition. He has contributed more than 25 international peer-reviewed journal papers.

Hongbin Wei received B.S. degree in Electrical Engineering from East China Jiaotong University, Nanchang, Jiangxi, China, in 2017. He is currently pursuing the M.S. degree in electrical engineering from University of Shanghai for Science and Technology, China. His research interests include image processing and multimedia security.

Chuan Qin received the B.S. degree in electronic engineering and the M.S. degree in signal and information processing from Hefei University of Technology, Anhui, China, in 2002 and 2005, respectively, and the Ph.D. degree in signal and information processing from Shanghai University, Shanghai, China, in 2008. Since 2008, he has been with the faculty of the School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, where he is currently a Professor. He was with Feng Chia University at Taiwan as a

Postdoctoral Researcher from July 2010 to July 2012. His research interests include image processing and multimedia security. He has published over 120 papers in these research areas.

Zhenjun Tang received the B. S. and M. Eng. degrees from Guangxi Normal University, Guilin, P. R. China, in 2003 and 2006, respectively, and the PhD degree from Shanghai University, Shanghai, P. R. China, in 2010. He is now a professor with the Department of Computer Science, Guangxi Normal University. His research interests include image processing and multimedia security. He has contributed more than 50 international journal papers. He holds six China patents. He is a senior member of China Computer Federation (CCF) and also a reviewer of more than 20 SCI-indexed journals, such as IEEE journals, IET journals, Elsevier journals, Springer journals, and Taylor & Francis journals.