J. Vis. Commun. Image R. 78 (2021) 103124

Contents lists available at ScienceDirect

Journal of Visual Communication and Image Representation

journal homepage: www.elsevier.com/locate/jvci

Perceptual hash-based coarse-to-fine grained image tampering forensics method☆,☆☆

Xiaofeng Wang [a,*], Qian Zhang [a], Chuntao Jiang [a], Jianru Xue [b]

[a] Xi'an University of Technology, Xi'an, Shaanxi 710048, PR China
[b] Institute of Artificial Intelligence and Robotics at Xi'an Jiaotong University, Xi'an, Shaanxi 710049, PR China

ARTICLE INFO

Keywords:
Perceptual image hash
Simple linear iterative clustering
Image forging detection
Image tempering localization

ABSTRACT

As an active forensic technology, perceptual image hash has important application in image content authenticity detection and integrity authentication. In this paper, we propose a hybrid-feature-based perceptual image hash method that can be used for image tampering detection and tampering localization. In the proposed method, we use the color features of image as global features, use point-based features and block-based features as local features, and combine with the structural features to generate intermediate hash code. Then we encrypt and randomize to generate the final hash code. Using this hash code, we present a coarse-to-fine grained forensics method for image tampering detection. The proposed method can realize object-level tampering localization. Abundant experimental results show that the proposed method is sensitive to content changes caused by malicious attacks, and the tampering localization precision achieves pixel level, and it is robust to a wide range of geometric distortions and content-preserving manipulations. Compared with the state-of-the-art schemes, the proposed scheme yields superior performance.

1. Introduction

With the rapid development of the digital technologies, a variety of excellent image processing software such as Adobe Photoshop, MeiTu, etc. have been extensively applied in various occasions. Convenient and smart image editing tools make it is very easy to modify or falsify image content. Various modified images are likely to be used in scientific researches, news media, political events, medical diagnosis, cultural media, evidence forgery, finance and military, and so on. The existence and spread of the faked photographs seriously reduces the credibility of image information, and have brought a great negative impact in many fields. Therefore, there is an urgent requirement for development powerful technology to detect the authenticity, integrity and primitiveness of image content.

Digital image forensics is one of the effective techniques for image content authentication; it includes active forensics and passive forensics. Active forensics includes image digital watermarking and perceptual image hash. The former need to embed authentication information into image, and thus destroys the primitiveness of image content. Perceptual image hash is a non-intrusive approach for content authentication. Passive forensics is a technique that can detect image content changing without relying on any prior information about the original image. It can detect wide range of image attacks, such as image splicing, copy-move, and so on. However, existing methods can only detect single content tampering attack. In some case, e.g., an image suffers both splicing tampering and copy-move attack; the existing passive detection methods cannot complete two types of detection in a single operation. While, using perceptual image hash, such type of problem can be easily solved. For an ideal perceptual image hash for content authentication, it should have the following properties [1]: compactness, perceptual robustness, one-way, visual fragility, and unpredictability.

In recent years, hash-based image tampering detection methods have emerged endlessly, and many excellent works have emerged. According to the differences of extracted image features, the existing methods can be divided into three categories: the global-feature-based methods, the local-feature-based methods, and the hybrid-feature-based methods.

☆ This paper has been recommended for acceptance by Zicheng Liu.

☆☆ This work was supported by the National Natural Science Foundation of China under Grant No.61772416; the National Major Research and Development Plan Program of China under Grant No.2016YFB1001004; the Key Laboratory Project of the Education Department of Shaanxi Province under Grant No.17JS098; Shaanxi province technology innovation guiding project, No.2018XNCG-G-02.

* Corresponding author.
E-mail address: xfwang66@sina.com.cn (X. Wang).

https://doi.org/10.1016/j.jvcir.2021.103124
Received 23 March 2020; Received in revised form 1 February 2021; Accepted 11 April 2021
Available online 20 May 2021

Early image hash methods were generated from the global features of the image. Representative work such as literature [2], the hash was generated by using the low-frequency coefficients of two-dimensional discrete cosine transform (DCT). Works [3] and [4] were the Non-negative Matrix Factorization (NMF) based image hash methods that were provided with shorter hash length and significant robustness. Khelifi et al. [5] proposed an image hash method based on statistical characteristics. In this work, the image was divided into overlapping blocks, then the mean of the absolute coefficients in each block were calculated to form a feature set. Finally, the Wei-bull model was used to extract statistical parameters of the feature coefficients to generate hash code. Above mentioned methods are all able to detect whether the image content has changed, but they cannot detect tampered image regions.

In order to detect tampered image regions, many local feature-based image hash methods were proposed in last decade. Zhang et al. [6] proposed a method for constructing image hash using the statistical value of DCT coefficients of image blocks. Works [7], [8] and [9] are also block-based image hash methods that used the statistical feature of the discrete wavelet transform (DWT) coefficients. Roy et al. [10] designed an image hash method based on image blocks and Scale Invariant Feature Transform (SIFT) feature points. Lu et al. [11] proposed an image hash method combining the SIFT features and block-based features. The SIFT features were encoded into compact visual words for geometric correction of the image, so that the hash is robust to geometric transformations, and block-based features were used to detect and locate image tampering regions. Wang et al. [12] proposed a perceptual image hash method that combined the SIFT feature points with image-block-based DCT coefficients. This method has good robustness to many image attacks and can detect tampered image regions. The disadvantage of this method is that the hash code is longer. Gorisse et al. [13] proposed a method that the image hash was generated by using the edge information of each image block.

In practical applications, image hash should be robust to content-preserving manipulations and geometric distortion. It should also be sensitive to malicious attacks. These requirements inspired scholars to focus on the research of image hash methods that are based on the combination features. Lu [14] et al. proposed an image hash method based on Radon transform and scale space theory. In this work, authors used the edge information of image that was extracted via Radon transform as the global features, and used edge direction histogram as the local features. The global features could be used to estimate the parameters of geometric transform, and the local features were used to detect tampered image regions. Zhao et al. [15] proposed a method that used Zernike moments of brightness and chromaticity of the image to generate global features, and the position and texture information of the salient regions to form local features. This method can only detect the change of salient image regions and is robust to several attacks such as JPEG compression, Gaussian noise and blurring.

Aiming at the problem that some image hash algorithms cannot detect small tampering region, Khavare et al. [16] proposed an image hash method based on ring segmentation and the non-negative matrix decomposition. The main contribution of this work is to detect small tampered regions, especially the corner regions. Take a general survey of the existing perceptual image hash methods, the main challenge is the lack of the trade-off between geometric distortion robustness and tampering localization ability. To address this problem, Tang et al. [17] presented an image hash method by combining ring segmentation and invariant vector distance in order to enhance both the robustness for rotation operation and discrimination for tampering operation. This algorithm can realize the discrimination for different images; however, it is not satisfactory in tampering detection and localization. Yan et al. [18] proposed a multi-scale image hash algorithm based on adaptive local feature points. In this work, authors used image center as the starting point to segment the image into multiple rings and sector regions according to different radius and angles. Then the adaptive local feature points were combined with the position of the context ring and context angle to generate ring hash and angle hash, respectively. This method is robust to most content-preserving manipulations, but it is difficult to obtain ideal detection results for the case that the local feature points are sparse.

In recent years, scholars have proposed more novel image hash methods. Yan et al. [19] proposed a new image hash algorithm based on the quaternion Fourier Merlin transform moment and the quaternion Fourier coefficients. Since this hash algorithm only uses the global features, it is invalid for small tampering regions or plurality of disconnected regions. Pun et al. [20] proposed a multi-region matching and object-level tampering detection algorithm based on image registration. This method can achieve tampering localization at object-level. Yan et al. [21] put forward a kind of binary ranking hash method based on image block, and a multi-scale difference map fusion method for tampering localization. In this work, the authors first reconstructed image using a quaternion low-pass filter, and decomposed the reconstructed image into blocks, then calculated the local binary code to obtain the binary sort hash. Finally, the authors used multi-scale difference mapping fusion method to locate the tampered image regions. Pun et al. [22] proposed an image hashing algorithm based on progressive feature selection. In this method, SIFT feature points were used as progressive feature, then local features and color features were generated in the specific region of each feature point. Finally, the horizontal and vertical position contexts were used to form hash. This algorithm is robust to content-preserving manipulations and can detect tampered image regions, it also cannot achieve the object-level tampering localization.

In this paper, we propose a multi-feature fusion image hash method and a hash-based image tampering detection and tampering localization method. The main contributions of this paper are as follows: (1) In the proposed method, we define global features, local features and structural features, and combine these features to generate intermediate hash code. (2) We propose a two phase's image tampering localization algorithm. We detect the tampering regions from coarse grained to fine grained, and finally, the locations and shapes of the tampered regions can be detected. (3) The proposed method is sensitive to content changes caused by malicious attacks, and the tampering localization precision achieves pixel level. Furthermore, it is robust to a wide range of geometric distortions and content-preserving manipulations.

The rest of this paper is organized as follows. In Section 2, we describe the proposed method. Experimental results and performance analysis are presented in Section 3. Section 4 concludes the paper with some thoughts on future works.

2. The proposed method

The proposed method includes image hash generation algorithm, and tampering detection and localization algorithm. The hash generation algorithm consists of a pipe process: image preprocessing, global features extraction, local features extraction, structural features extraction, encryption and randomization, and compressing and coding. The role of the global feature is to detect whether an image has undergone content tampering. If the image content has been tampered, appropriate geometric correction and tampering region detection will be performed. The local features include points-based features and block-based features. The points-based local features will be used to perform the geometric correction, the block-based local features will be used to detect suspicious regions that most likely to include the tampering regions, and the structural features will be used to determine the accuracy position and shape of the tampered image regions.

2.1. Image preprocessing

To generate fixed length of hash values from images with different dimensions, for inputted image I with size $M \times N$, we resize it to $M_r \times M_r (M_r = 256)$ using the method of bi-linear interpolation. For

short, we still use I to denote the re-sized image. To avoid the influence of noise, we process the image I using a 2D Gaussian low-pass filter.

$$G(x,y,\sigma)=\frac{1}{2\pi\sigma^2}e^{\frac{-(x^2+y^2)}{2\sigma^2}} \tag{1}$$

where, (x,y) denotes the position coordinate of the pixel and σ is the standard deviation. Let I_o denote the Gaussian low-pass filtered image, then we have:

$$I_o = I \otimes G(x,y,\sigma) \tag{2}$$

where, $\otimes$ denotes convolution operation.

2.2. Feature extraction and generation of the intermediate hash

The feature extraction includes global features extraction, local features extraction, and structural features extraction.

2.2.1. The definition and representation of the global features

According to the visual characteristics of the human eye, the brightness sensitivity is greater than that of chromaticity sensitivity, so we convert the image I_o from *RGB* color space to *YCbCr* color space using formula (3):

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} + \begin{bmatrix} 65.481 & 128.553 & 24.966 \\ -37.797 & -74.203 & 112.000 \\ 112.000 & -93.786 & -18.214 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{3}$$

where R, G and B are the red, green and blue color components of pixels, respectively; Y, Cb and Cr represent the luminance, blue chrominance component, and red chrominance component, respectively.

Then Y, Cb and Cr are normalized as follows.

$$Y'(x,y)=\frac{Y(x,y)}{255}, Cb'(x,y)=\frac{Cb(x,y)}{255}, Cr'(x,y)=\frac{Cr(x,y)}{255} \tag{4}$$

We calculate the mean Y', Cb' and Cr', respectively, and take these mean as the global features of the image.

$$\overline{Y_o}=\sum_{x=1}^{M_r}\sum_{y=1}^{M_r}\frac{Y'(x,y)}{M_r\times M_r},\quad \overline{Cb_o}=\sum_{x=1}^{M_r}\sum_{y=1}^{M_r}\frac{Cb'(x,y)}{M_r\times M_r},\quad \overline{Cr_o}=\sum_{x=1}^{M_r}\sum_{y=1}^{M_r}\frac{Cr'(x,y)}{M_r\times M_r} \tag{5}$$

where, (x,y) represents the position coordinates of pixels in corresponding color space. Then the global features of the image I_o can be denoted as: $H_{o0}=[\overline{Y_o},\ \overline{Cb_o},\ \overline{Cr_o}]$.

2.2.2. The definition and representation of point-based local features

Considering that the SIFT features are invariant to rotation and scaling [23], and are stability to affine transformation, we use the SIFT feature points as point- based local features of the image I_o that will be used for subsequent geometric correction.

Let $S_o=\{S_{o1}(x_{o1},y_{o1}),S_{o2}(x_{o2},y_{o2}),\ldots,S_{on}(x_{on},y_{on})\}$ represent the set of the SIFT feature points extracted from the image, where S_{ov} denote the SIFT feature point, (x_{ov},y_{ov}) denote the position coordinate of S_{ov}, $v=1,\ldots,n$, n is the number of feature points, and each S_{ov} is a 128-dimension feature vector, denoted as P_{ov}. Let $M_{P_o}=[P_{o1},P_{o2},\ldots,P_{on}]^T$ represents the matrix of feature vector.

To reduce the dimension of feature vector, we use Gaussian random matrix G_m to perform compression projection on M_{P_o}, and the compressed matrix M'_{P_o} can be obtained.

$$M'_{P_o}=M_{P_o}\cdot G_m=[P_{o1},P_{o2},\ldots,P_{on}]^T\cdot G_m=[P'_{o1},P'_{o2},\ldots,P'_{on}]^T \tag{6}$$

where, $[\cdot]^T$ indicates the transpose of the matrix, m is the compression projection rate.

We define point-based local features H_{o1} of the image as the compressed feature vector and the coordinates of the corresponding feature points.

$$H_{o1}=(P'_{o1},P'_{o2},\ldots,P'_{on},(x_{o1},y_{o1}),(x_{o2},y_{o2}),\ldots,(x_{on},y_{on})) \tag{7}$$

In the process of image feature matching, it is important to select the appropriate number of SIFT feature points. Too many SIFT feature points will increase the length of the hash code, and too little will affect the performance of geometric correction. According to work [12], we select $21\leqslant n\leqslant 25$ as the number of feature points, and take $m=7.8\%$ as the best compression projection rate of the feature vector.

2.2.3. The definition and representation of the block-based local features

In order to extract block-based local features, we perform Stationary Wavelet Transform (SWT) [24] for image. The SWT also known as redundant wavelet transform. It is different from the conventional two-dimensional discrete wavelet transform. The SWT does not sample the signal, but sampling the filter. Moreover, the SWT also has translation invariance.

For image I_o, we use first-level SWT to obtain approximate/low frequency sub-band *LL*, and detail/high frequency sub-band *LH*, *HL* and *HH*. Since sub-band *LL* has the same size as the image I_o, we divide *LL* into non-overlapping regular blocks $I_i(i=1,2,\ldots,\zeta)$ with the size $p\times p$, where $\zeta=(M_r\times M_r)/(p\times p)$. In this work, we set $p=16$.

For each image block I_i, we use DCT to obtain the coefficient matrix, and scan the coefficient matrix from the second component of the first row according to the zigzag to obtain feature vector $h_{oi}=(h_{oi1},h_{oi2},h_{oi3},h_{oi4},h_{oi5})$. Then the block-based local features of image I_o can be defined as:

$$H_{o2}=[h_{o1},h_{o2},\ldots,h_{o\zeta}] \tag{8}$$

2.2.4. The definition and representation of the structural features

Generally, image tampering will cause the change of visual content. While, simple linear iterative clustering (SLIC) algorithm [25] will segment image into non-overlapping irregular blocks according to visual content, referred to as super-pixel blocks, and the compactness and contour retention of the super-pixel blocks are ideal.

For image I_o, structural features are defined as the position coordinates and the color features of the super-pixel blocks.

Considering that the bilateral filtering [26] can keep the edge of the image as better as possible, which will provide better effect for image segmentation. So, we use bilateral filtering to process image I_o, and obtain the filtered image I_1. Then we adopt SLIC algorithm to segment image I_1 into W (in this work, we set $W=200$) super-pixel blocks, each block is denoted as $B_{o\gamma}(\gamma=1,2,\ldots,W)$, and central coordinate of each block is denoted as $(x''_{o\gamma},y''_{o\gamma})$. For each super-pixel block, we calculate the mean values of three-color channels, respectively, and denoted as $\overline{Y_{o\gamma}}$, $\overline{Cb_{o\gamma}}$ and $\overline{Cr_{o\gamma}}$. We define the feature of $B_{o\gamma}$ as: $l_{o\gamma}=[x''_{o\gamma},y''_{o\gamma},\overline{Y_{o\gamma}},\overline{Cb_{o\gamma}},\overline{Cr_{o\gamma}}]$.

For image I_o, the structural features are defined as:

$$H_{o3}=[l_{o1},l_{o2},\ldots,l_{oW}] \tag{9}$$

2.3. Hash generation

2.3.1. Intermediate hash generation

Combine H_{o0}, H_{o1}, H_{o2} and H_{o3} to generate the intermediate hash. We define the intermediate hash value of the image I_o as follows:

$$H_{o\text{ - }mid}=H\ (H_{o0},\ H_{o1},\ H_{o2},\ H_{o3},\ W,\ M,\ N) \tag{10}$$

Here, $H(\cdot)$ is a cascade function, the length of the hash code is:

$$L=3n+5M_r\times M_r/(p\times p)+5W+6 \tag{11}$$

In this work, the values of W, M_r, and p are fixed, when $n=25$, the length of hash code is 2361 digits.

2.3.2. Final hash generation

To ensure the security of the hash algorithm, it is necessary to encrypt and randomize the intermediate hash. To this end, we generate the encryption key K using the part of the structure features, then encrypt and randomize the intermediate hash to obtain the final hash. The specific process is as follows.

(1) Key generation

Step 1. For different images, the number and position of super-pixel blocks are different. We rank the super-pixel blocks according to their position in the image to form a sequence, then extract the super-pixel block that is locating in the middle of the sequence, denoted as b_s. Then the horizontal/vertical coordinate of b_s are converted into binary, denoted as $b(x)$ and $b(y)$, and then perform bit-xor operation, generating initial seed point k_1.

$$k_1 = b(x) \oplus b(y) \tag{12}$$

where, x and y represent horizontal coordinate and vertical coordinate of b_s, $b(\cdot)$ represents conversion of the decimal to binary, and $\oplus$ represents the bit-xor operation.

Step 2. Considering that the linear congruence algorithm is provided with fast computing speed, we use the linear congruence algorithm to generate a pseudo-random sequence as the encryption key K. The sequence can be obtained by formula (13).

$$k_{n+1} = (ak_n + b)(\ mod\ C) \tag{13}$$

where, a, b are the constants, C is the period.

According to [27], we set $a = 219$, $b = 97$, $C = L$, L is the hash length obtained by Section 2.2.4, $n = 1, 2, \ldots, L\text{-}1$.

According to the formula (13) and parameters a, b, L and k_1, the encryption key sequence can be generated.

$$K = (k_1, k_2, \ldots, k_L) \tag{14}$$

(2) Encryption and randomization

To encrypt the intermediate hash code, we use key sequence $K = (k_1, k_2, \ldots, k_L)$ to encrypt the intermediate hash code. See the following equation:

$$H_c(z) = K(z) \times H_{o\_mid}(z) \tag{15}$$

where, $z = 1, 2, \ldots, L$, " $\times$ " indicates multiplication, $H_{o\_mid}(z)$ is z^{th} of the intermediate hash code.

The cipher hash value H_c is as follows: $H_c = (H_c(1), H_c(2), \ldots, H_c(L))$.

(3) Coding and compressing

Finally, in order to obtain a compact hash code, we use Huffman encoding [33] to compress H_c. Final hash code H_o is generated by adjoining the Huffman codes of leaves in Huffman tree which correspond to each element of H_c.

2.4. Tampering detection and tampering localization

For received image hash H_o, Huffman decoding and decrypting are applied to obtain an intermediate hash code $H_{o\ -\ mid} = H(H_{o0}, H_{o1}, H_{o2}, H_{o3}, W, M, N)$.

For tested image I_t, we use the intermediate hash generation algorithm described in Section 2.2 to generate the intermediate hash code $H_{t\ -\ mid} = H(H_{t0}, H_{t1}, H_{t2}, H_{t3}, W_t, M_t, N_t)$. Here, $H_{t0} = [\overline{Y_t}, \overline{Cb_t}, \overline{Cr_t}]$ represents the global feature of I_t; $H_{t1} = (P'_{t1}, P'_{t2}, \ldots, P'_{tn}, (x_{t1}, y_{t1}), (x_{t2}, y_{t2}), \ldots, (x_{tn}, y_{tn}))$ represents the point-based local features of I_t; $H_{t2} = [h_{t1}, h_{t1}, \ldots, h_{tP}]$ represents a block-based local features of I_t, here, $h_{ti} = (h_{ti1}, h_{ti2}, h_{ti3}, h_{ti4}, h_{ti5})$, $i = 1, 2, \ldots, P$; $H_{t3} = [l_{t1}, l_{t2}, \ldots, l_{tW_t}]$ represents structural features of I_t, $l_{t\alpha} = [x''_{t\alpha}, y''_{t\alpha}, \overline{Y_{t\alpha}}, \overline{Cb_{t\alpha}}, \overline{Cr_{t\alpha}}]$, $\alpha = 1, 2, \ldots, W_t$.

2.4.1. Geometric correction

In order to avoid the influence of geometric distortions on the tampering detection results, geometric correction is required. In the work, we use the points-based local features to perform geometric correction.

For received hash H_o, we can obtain the feature vectors H_{o1}, and image size M and N. For the tested image I_t, we can obtain H_{t1}. We use the SIFT feature matching algorithm to carry out similarity matching, and the matched feature point pairs are denoted as (t_i, t'_i). The central coordinates (x_{t_i}, y_{t_i}) and $(x_{t'_i}, y_{t'_i})$ of t_i and t'_i are used to judge whether the tested image has undergone geometric transformation. If $\lfloor x_{t_i} - x_{t'_i} \rfloor \neq 0$ and $\lfloor y_{t_i} - y_{t'_i} \rfloor \neq 0$, it is considered that the tested image has undergone geometric transformation, otherwise, the tested image has not undergone geometric transformation. Here, $\lfloor \cdot \rfloor$ represents an operation of rounding down. If the tested image has undergone geometric transformations, then we use the affine transformation to carry out geometric correction on the image.

$$\text{Let } U_o = \begin{bmatrix} y_{t_1} & y_{t_2} & \ldots & y_{t_n} \\ x_{t_1} & x_{t_2} & \ldots & x_{t_n} \\ 1 & 1 & \ldots & 1 \end{bmatrix},$$

$$U_t = \begin{bmatrix} y_{t'_1} & y_{t'_2} & \ldots & y_{t'_n} \\ x_{t'_1} & x_{t'_2} & \ldots & x_{t'_n} \\ 1 & 1 & \ldots & 1 \end{bmatrix} \tag{16}$$

Using Eq. (17) to calculate affine transformation matrix Q between corresponding matching point pairs.

$$Q \cdot U_o = U_t \tag{17}$$

For tested image I_t, using matrix Q to correct image, then according to image size M and N to perform bi-linear interpolation to obtain corrected image.

2.4.2. Tampering detection

We use the global features to perform tampering detection. To this end, we calculate the Euclidean distance Dis between H_{o0} and H_{t0}, and Dis is used to detect whether the tested image I_t has been tampered. The diagram of the tampering detection algorithm is shown in Fig. 1.

$$Dis = ||H_{o0} - H_{t0}||_2 \tag{18}$$

Here, $||\cdot||_2$ denotes the Euclidean distance. If $Dis < T_2$, it is considered that the tested image is authentic. If $Dis > T_1$, it is considered that the tested image is different from the original image. Otherwise, it is considered that the tested image has been tampered, and the tampering regions can be found by using the method described in Section 2.4.3. The acquisition method of the threshold T_1 and T_2 will be present in the Section 3.1.1.

2.4.3. Tampering localization

In this section, we propose a coarse-to-fine grained image tampering localization algorithm. The algorithm diagram is shown in Fig. 2. The proposed method is consisting of two phases. In the phase of coarse grained tampering localization, we use the block-based local features to detect suspicious regions that are most likely including tampering regions, seeing the red box in Fig. 2. In the phase of fine grained tampering localization, we use the structural features to find suspicious super pixel blocks, seeing the green box in Fig. 2. Then we use both the suspicious regions and suspicious super pixel blocks to estimate the accuracy position and shape of tampered image region.

(1) Coarse-grained tampering localization

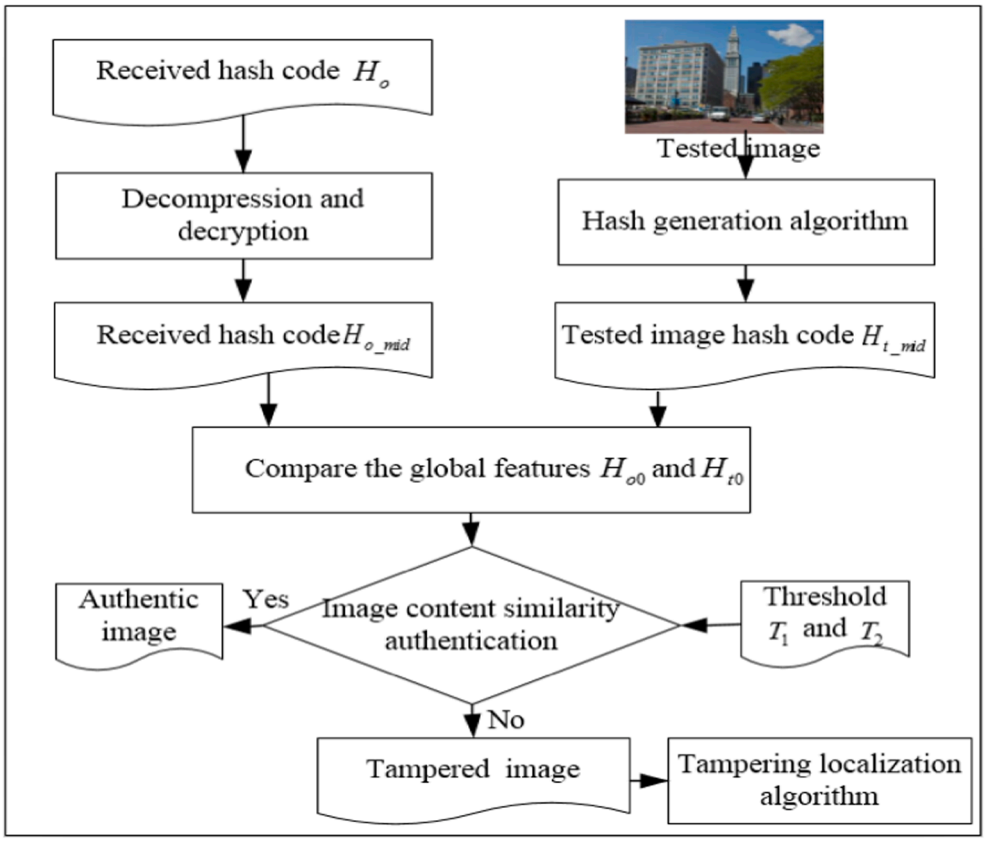


Fig. 1. The frame diagram of tampering detection.

To detect suspicious regions that most likely to include the tampering regions, we investigate the Euclidean distance D_{i1} between H_{o2} and H_{t2}.

$$D_{i1} = \|H_{o2} - H_{t2}\|_2 \tag{19}$$

where, $i = 1,2,\ldots,\zeta$. If $D_{i1} > \tau_1$, it is considered that i^{th} block is tampered block, and τ_1 is the threshold that is determined according to the adaptive threshold algorithm (see details in Section 3.1.2).

We save the tampered blocks into a set A. Searching row and column numbers of each block, let *max_m* and *min_m* denote the largest and smallest row index, respectively, let *max_n* and *min_n* denote the largest and smallest column index, respectively, and the rectangular region enclosed by four indexes is denoted as $Z = \{max\_m, min\_m, max\_n, min\_n\}$. See Fig. 3. We believe that the region Z is suspicious area that may include tampered image regions.

(2) Fine-grained tampering localization

The structural feature is quite important for an image. Once the image content has been changed slightly, the structural feature will be changed significantly. Therefore, if the image content is tampered, even if the number of super-pixel seed point is same, both the number of and the center position of super-pixel blocks will be changed. Therefore, we can check the matching relationship between the structural features from received hash and the structural features from the tested image to determine tampered image regions.

① Super-pixel blocks matching

We estimate the Euclidean distance $D'_{\alpha\gamma}$ between central position coordinate $(x''_{t\alpha}, y''_{t\alpha})$ of each super-pixel block $B_{t\alpha}(\alpha = 1,2,\ldots,W_t)$ in structural feature H_{t3} and central position coordinate $(x''_{o\gamma}, y''_{o\gamma})$ of all super-pixel blocks $B_{o\gamma}(\gamma = 1,2,\ldots,W)$ in structural feature H_{o3}.

$$D'_{\alpha\gamma} = \sqrt{(x''_{t\alpha} - x''_{o\gamma})^2 + (y''_{t\alpha} - y''_{o\gamma})^2} \tag{20}$$

By this way, each super-pixel block $B_{t\alpha}$ of the tested image I_t is associated with W Euclidean distances $D'_{\alpha\gamma}(\gamma = 1,2,\ldots,W)$, denoted as $(D'_{\alpha1}, D'_{\alpha2}, \ldots, D'_{\alpha W})$.

Let $min\_d(\alpha) = min(D'_{\alpha1}, D'_{\alpha2}, \ldots, D'_{\alpha W})$, where, $\alpha = 1,2,\ldots,W_t$.

Finally, using the position information in structural features H_{o3}, we can obtain the super- pixel blocks $B_{o\beta}(\beta \in (1,2,\ldots,W))$ corresponding to the minimum distance $min\_d(\alpha)$. Therefore, $B_{t\alpha}(\alpha = 1,2,\ldots,W_t)$ that matches with $B_{o\beta}(\beta \in (1,2,\ldots,W))$ can be found.

② Tampering region localization

To find the exact location of tampered image regions, we estimate the Euclidean distance $D_{\alpha2}$ of the color feature between matching super-pixel blocks $B_{o\beta}$ and $B_{t\alpha}$.

$$D_{\alpha2} = \|C_o - C_t\|_2 \tag{21}$$

where, $C_o = (\overline{Y_{o\gamma}}, \overline{Cb_{o\gamma}}, \overline{Cr_{o\gamma}})(\gamma \in (1,2,\ldots,W))$is the color feature come from H_{o3}, $C_t = (\overline{Y_{t\alpha}}, \overline{Cb_{t\alpha}}, \overline{Cr_{t\alpha}})(\alpha \in (1,2,\ldots,W_t))$ is the color feature of the super-pixel block $B_{t\alpha}$ from tested image I_t.

If $D_{\alpha2} > \tau_2$ and the central of the super-pixel block is located in the rectangular region Z, it is considered that α^{th} super-pixel block is tampered, and τ_2 is a threshold that is determined according to the adaptive threshold algorithm (see details in Section 3.1.2).

3. Experimental results and performance analysis

In this section, we evaluate the proposed solution via simulation experiment. We implemented and tested our method using MATLAB

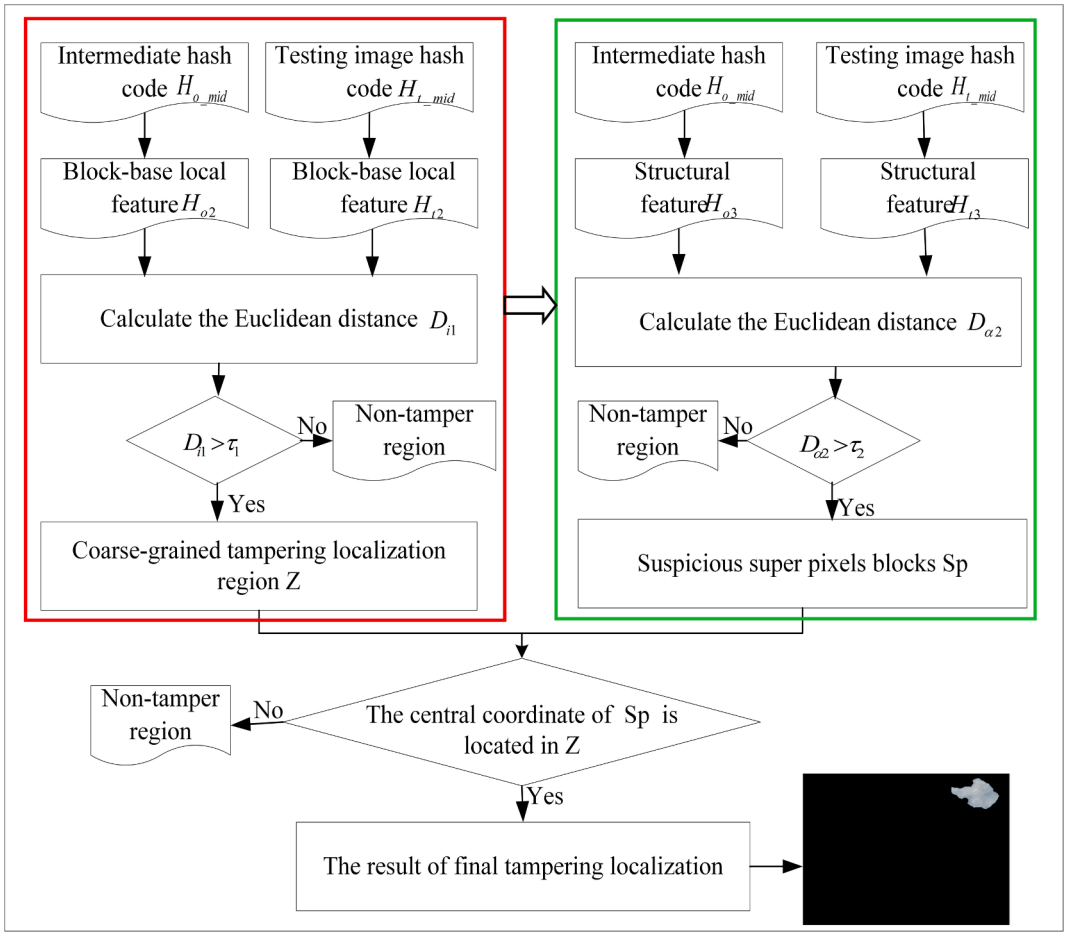


Fig. 2. The frame diagram of tampering localization.

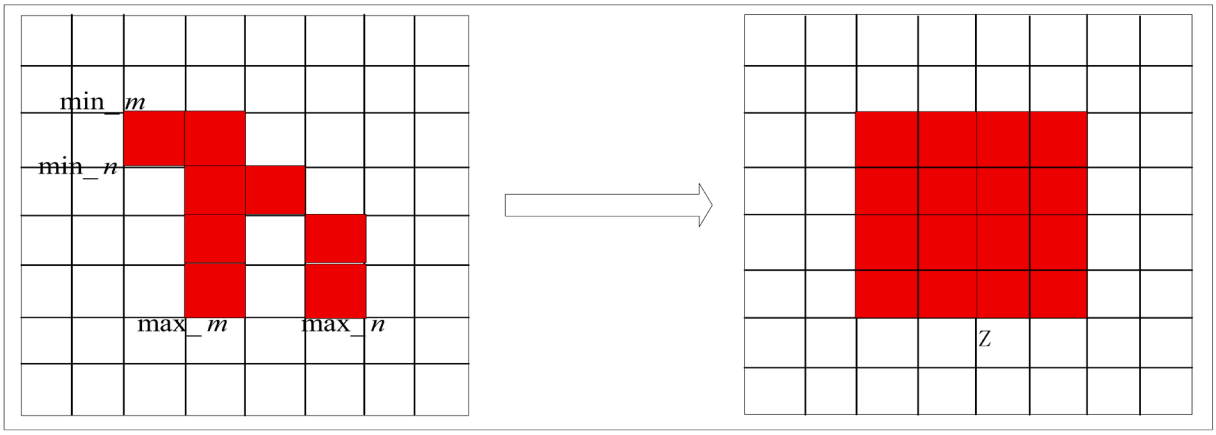


Fig. 3. Suspicious regions.

R2014a. The experiment was run on a computer with an Intel(R) Core (TM) CPU, i5-3470 @ 3.2 GHz, 4.00 GB RAM. The image databases used in this work include the CASIA v1.0 [28], the Ground truth Database [29] from the University of Washington, the UCID Database [30].

3.1. Parameters estimation

For the sake of clarity, the thresholds used in the experiment are shown in Table 1.

Table 1
Thresholds symbol description.

Symbol	Meaning
T_1	The threshold to distinguish tampered images from different images
T_2	The threshold to distinguish tampered images from similar images
τ_1	The threshold for coarse-grained tampering localization
τ_2	The threshold for fine-grained tampering localization
T_c	The threshold for robustness experiment
T_r	The threshold for collision experiment

3.1.1. Parameters estimation for image content authenticity detection

In order to estimate the parameters used in this work, we take randomly 800 original images and corresponding tampered versions from CASIA v1.0 database [28], and 1200 images from Ground truth Databases [29] and UCID [30]. By using benchmark Stirmark software, ten content-preserving manipulations are performed on each original image. These operations include Gaussian blur with standard deviations 6 and 10, JPEG compression with a quality factors 60 and 90, Gaussian noise with means 0, variances 0.001 and 0.005, scaling with factors 0.8 and 1.4, and rotation with angle 5 and 10.

To evaluate threshold values T_1 and T_2 that we are using in the Section 2.4.2, we introduce a definition of 'Detection rate' that was defined in the work [12], as shown in the Eq. (22), that is, the probability of correct detection. Fig. 4 shows the detection rates at different thresholds T_1, where the red and blue curves represent the detection rates for the tampered images and different images, respectively. From the Fig. 4, we can see that the intersection of the red curve and blue curve at approximately $T_1 = 10$. Fig. 5 shows the detection rates under different thresholds T_2. From the Fig. 5, we can see that the curves of the tampered images and similar images intersects at $T_2 = 0.8$. Therefore, in our experiments, we use $T_1 = 10$ and $T_2 = 0.8$ as thresholds to distinguish different images, similar images and tampered images.

$$\text{Detection rate} = \frac{\text{The number of correct detection}}{\text{The total number of tested images}} \tag{22}$$

3.1.2. Parameters estimation for adaptive threshold

We use the OTSU method [31] to estimate the thresholds τ_1 and τ_2. The OTSU method also known as maximum inter-class variance method, is an efficient algorithm for image dichotomy. It was proposed by Japanese scholar Nobuyuki OTSU in 1979. The idea behind the OTSU method is clustering the pixels into two classes according to estimate an appropriate variance of the pixels that make the difference in inter-class is largest, and the difference in intra-class is smallest. Therefore, for two-classification problems, the OTSU algorithm can be used to automatically select threshold that divide the data set into two parts. The algorithm process is as follows.

To estimate the threshold value τ_1, we investigate the Euclidean distance D_{i1} between the block-based local features H_{o2} and H_{t2} according to the Eq. (19), denote $D_{i1}\,(i = 1, 2, ..., \zeta)$ into a set $D = (D_{11}, D_{21}, ..., D_{\zeta 1})$.

Our goal is seeking for a value $e_l \in D$ that divide set D into two subsets W_1 and W_2, such that the difference between W_1 and W_2 is maximum, and the differences within W_1 and W_2 are minimum. Assuming W_1 and W_2 can be denoted as $W_1 = (D_{e_1 1}, D_{e_2 1}, ..., D_{e_l 1})$, $W_2 = (D_{e_{l+1} 1}, ..., D_{\zeta 1})$, then the probability of W_1 and W_2 are:

$$\rho_1 = ||W_1||/\zeta, \quad \rho_2 = 1 - ||W_1||/\zeta \tag{23}$$

where $||\cdot||$ represents the cardinality number of the set.

Then, the mean values of W_1 and W_2 are:

$$m_1 = \frac{1}{||W_1||}\sum_{i=e_1}^{e_l} D_{i1}, \quad m_2 = \frac{1}{\zeta - ||W_1||}\sum_{i=e_{l+1}}^{\zeta} D_{i1} \tag{24}$$

Let

$$m_{total} = \rho_1 m_1 + \rho_2 m_2 \tag{25}$$

The variance between W_1 and W_2 is:

$$v^2(e_l) = \rho_1 (m_1 - m_{total})^2 + \rho_2 (m_2 - m_{total})^2 \tag{26}$$

Our goal is seeking for a value $e_l^* \in D$ such that

$$e_l^* = \underset{e_l}{Argmax}[v^2(e_l)] \tag{27}$$

Then e_l^* is the required threshold value τ_1. Similarly, we can get the threshold τ_2.

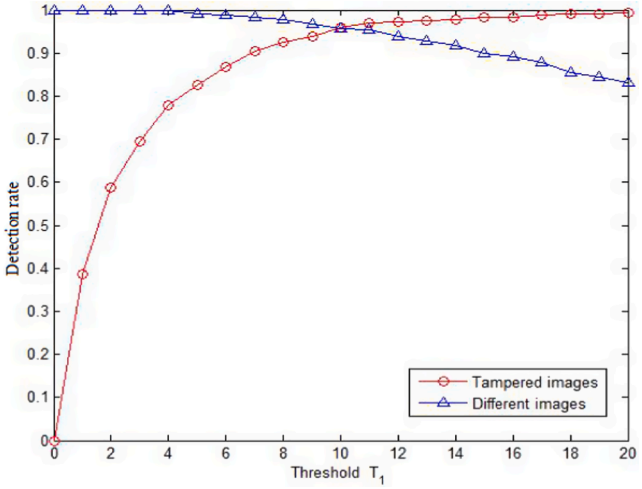


Fig. 4. The estimation of threshold T_1

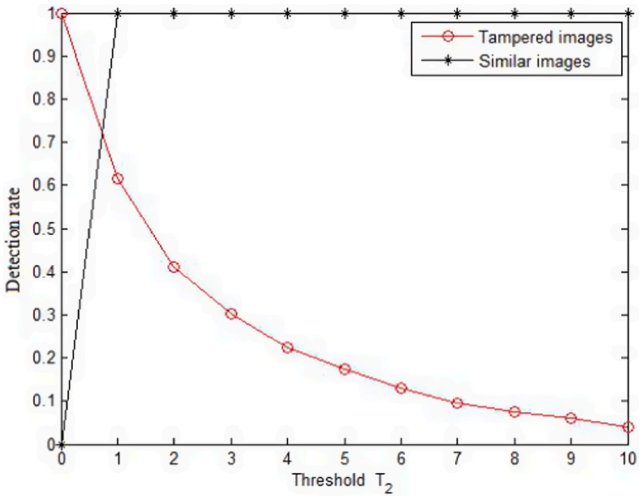


Fig. 5. The estimation of threshold T_2.

3.2. Robustness analysis and comparison

The purpose of this section is to test whether the proposed method is robust to incidental changes caused by content-preserving manipulations. In our experiment, we use 1600 different images that come from Ground truth Database [29] as tested images. We carry out content-preserving manipulations for these images: Gaussian noise, Gaussian blurring, JPEG compression, salt and pepper noise, motion blurring, out-

Table 2
Content-preserving manipulations and parameter settings.

Type	Parameter Settings				
Motion blurring(motion pixels)	1.000	2.000	3.000	4.000	5.000
Out-of-focus blurring(radius)	0.200	0.600	0.800	1.400	1.800
Gaussian blurring(standard deviation)	2.000	4.000	6.000	8.000	10.000
JPEG compression(quality factor)	20.000	40.000	60.000	80.000	90.000
Salt and pepper noise(noise factor)	0.001	0.005	0.009	0.030	0.050
Gaussian noise(noise factor)	0.001	0.003	0.005	0.007	0.009
Rotation(angle)	5.000	10.000	15.000	20.000	25.000
Scaling(scaling scale)	0.500	0.800	1.200	1.500	2.000

of-focus blurring, rotation and scaling. The specific parameter settings are shown in Table 2. We calculate the hash value of the original image and the corresponding manipulated image. Then, we compare these hash values pairwise and measure the hash distance between the original image and the manipulated image.

Fig. 6. shows the test results of the detection rate (as shown in the Eq. (22)) under different content-preserving manipulations.

As shown in the Fig. 6, the detection rate is satisfactory when threshold $T_c > 2$. This means that the proposed method is robust to content-preserving manipulations and geometric distortions. In addition, with the increase of the threshold T_c, the detection rate is increasing gradually and finally tends to stable, reaching 100% at a certain point. This means that the robustness is enhancing with the increase of the threshold T_c.

In order to analyze the performance in detail, we compare the detection rate of the proposed method with that of the existing methods from the perspective of content-preserving manipulations. Table 3 shows the detection rate of different methods under content-preserving manipulations. As shown in Table 3, compared with the image block-based method [12], the feature point-based method [22] and the local feature-based methods [18] and [20], the proposed method is provided with high detection rate under content-preserving manipulations, this means that the proposed method has good perceptual robustness for content-preserving manipulations.

3.3. Sensitivity analysis and performance comparison

For image hashing, perceptual robustness and perceptual sensitivity are contradictory. Perceptual robustness requires good stability under slight disturbance, while perceptual sensitivity requires sensitivity to minor malicious modification. An idea image hash method requires the trade-off between perceived robustness and perceived sensitivity. To analyze these characteristics quantitatively, we use definition of the false negative rate (R_{FN}) and the false positive rate (R_{FP}) that are described in work [12].

In this experiment, 1600 tested images came from Ground truth Database [29]. Forgery attacks include image splicing, copy-move, object replacement attacks and object removal attacks. The content-preserving manipulations include JPEG compression, adding noise, blurring, filtering, rotation, and scaling. The parameters are shown in the Table 4.

To assessment the robustness and sensitivity of the proposed method, we investigate and compare the receiver operating characteristic curve (ROC). For each tested image I_t, we calculate R_{FN} and R_{FP}, respectively. We repeat this process at different thresholds and finally got the ROC (Fig. 7).

We compare the proposed method with methods [12] and [20] in term of ROC curve, and experimental results are shown in Fig. 8. As can be seen from Fig. 8, In the case that false negative rate is same, the detection rate of the proposed method is higher than that of the comparative methods.

3.4. Comprehensive performance comparison

To evaluate the overall performance of the proposed method, we compare it with some state-of-the-art image hashing schemes. The comparison results are shown in Table 5. Here, "Yes(No)" means that the method is providing with (is not providing with) the corresponding functionality. TLC refers to the tampering localization; OLTC refers to object-level tampering localization; JPEG refers to JPEG compression; AN refers to adding noise; MB refers to motion blurring; OFB refers to out-of-focus blurring.

Additional, in the last column of the Table 5, n_p is the number of feature points in works [1] and [12], ε_1, ε_2 are the projection rates of Gaussian random matrices in work [12], $M \times N$ are the size of the testing image in work [12], p is the size of the image block in work [12], and N_G is the dimension of the global features in work [20].

The results in Table 5 show that the performance of the proposed method is more comprehensive, through the size of hash code is a bit longer. Longer hash codes, on the other hand, carry more image detail information, is providing with stronger object-level tampering localization functionality (we will discuss in the next section), which is exactly a very important evaluation index of a hash scheme.

3.5. The precision of tampering localization

The purpose of this experiment is to evaluate the detection performance of the proposed method in quantitatively means. To this end, we estimate the true positive rate (R_r), the false positive rate (R_f) and the F_1 score that takes into account detection precision (R_p) and the recall rate (that the true positive rate, R_r). The F_1 score is an indicator to measure the accuracy of the detection model in statistics. Their definitions are as follows:

$$F_1 = 2 \cdot (R_p \cdot R_r)/(R_p + R_r) \tag{28}$$

$$R_p = TP/(TP + FP) \tag{29}$$

$$R_r = TP/(TP + FN) \tag{30}$$

$$R_f = FP/(TN + FP) \tag{31}$$

where, TP represents numbers of pixels are correctly detected in the tampering regions, FP represents number of pixels are incorrectly detected in the tampering regions, and FN represents number of pixels that are in the tampering regions but are not detected.

In this experiment, we take color images from UCID database [30] and randomly tailor square blocks of 25×25, 50×50, 75×75, 100×100, 125×125, 150×150, 175×175 and 200×200, then splice these blocks into 1200 images from the Ground truth Database [29], respectively, to generate 9600 spliced images. We use the proposed method to detect these tampered images, and estimate the F_1 score, R_r and R_f. The average values of the F_1 score, R_r, R_f are shown in Table 6.

As can be seen from the Table 6, with the increases of the size of the tampering regions, the value of F_1 score and R_r will be larger. This means that the size of the tampering region is large, the more accurate the tampering localization will be. When the size of the tampering regions is 25×25, the F_1 score and R_r are smaller. This means that if the size of the tampering region is very small, the detection ability of the algorithm would decline.

3.6. The visual effect of tampering localization

For image hash scheme, tampering localization functionality is very significant. Aiming at different types of tampering attacks, we investigate the tampering localization effects of the proposed method from visual effects and pixel-level detection rate.

In the experiments, we take testing images from Ground truth Database [29], and operate these images using object replacement attack, add object, copy-move attacks, multiple object insertion attacks, and combine with the operations such as rotation and scaling, then test these images using the proposed method. Figs. 9–14 show the examples of detection results.

As can be seen from Figs. 9–14, using the proposed method, the shape of tampering regions can be detected, even if tested image has undergone geometric distortions.

In order to assess the detection performance quantitatively, we introduce the tampering rate (r_1) and the detection rate (r_2) at pixel level that were defined in work [12]:

$$r_1 = \frac{\text{The size of tampered regions}}{\text{The size of tested image}} \times 100\% \tag{32}$$

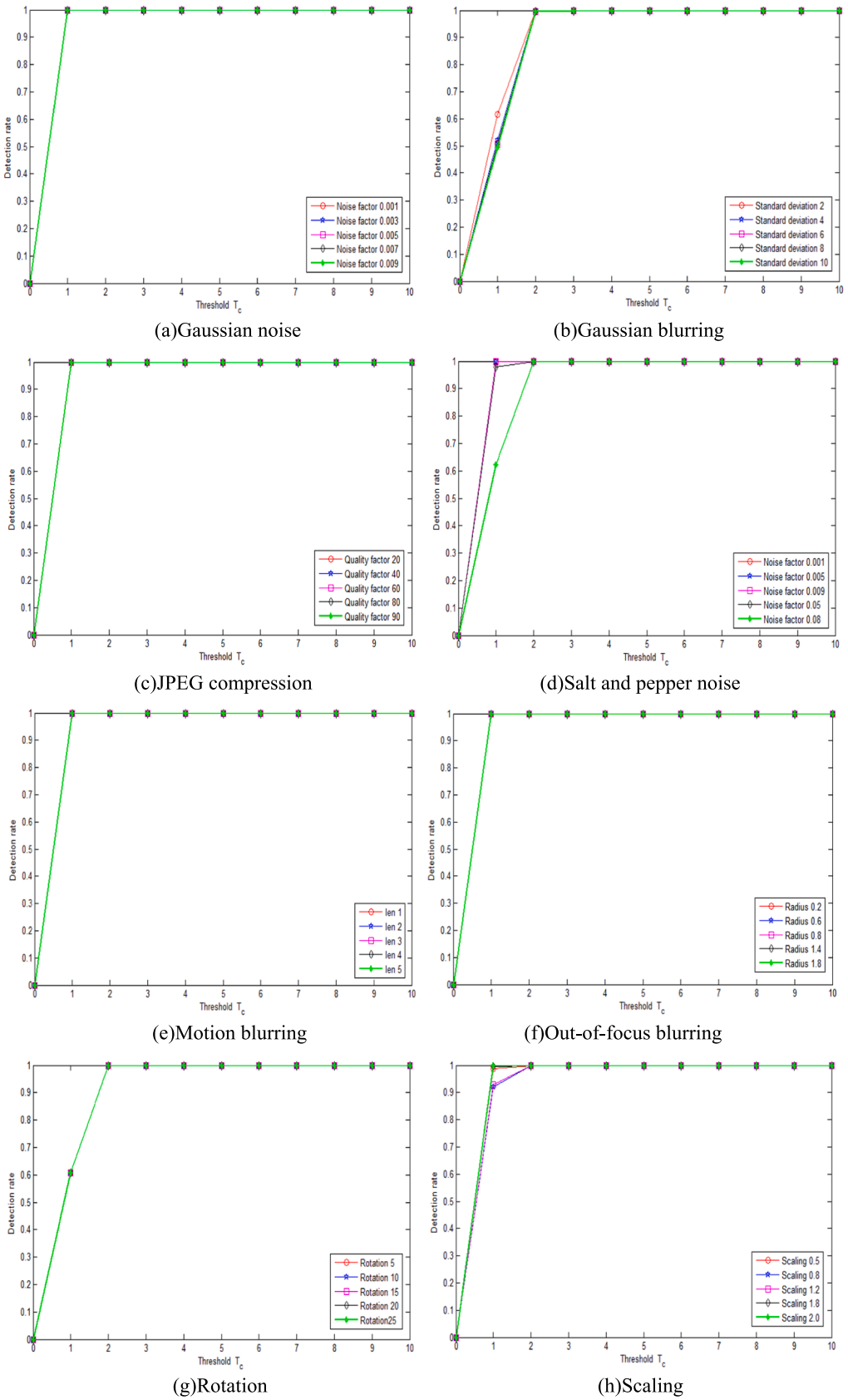


Fig. 6. Robustness tests for content-preserving manipulations.

Table 3
The comparison results of the detection rate (%).

Works	Method	JPEG quality factor				Gaussian blurring				Salt and pepper noise			
		20	60	80	90	4	6	8	10	0.005	0.009	0.050	0.080
Wang [12]	Image Block	98.500	99.350	99.500	99.800	97.000	97.200	98.100	98.500	**100.000**	98.800	99.000	97.900
Yan [18]	Local feature	98.500	99.000	99.250	**100.000**	98.000	99.000	99.500	99.500	99.000	97.000	97.100	97.000
Pun[20]	Local feature	96.000	99.000	99.000	98.000	**99.600**	**99.800**	**100.000**	97.000	99.500	99.700	98.200	98.000
Pun [22]	Feature Point	99.980	**100.000**	99.500	99.870	98.000	97.000	98.000	97.500	98.850	99.000	98.100	97.700
Our method T_c = 1.8	Hybrid Feature	**100.000**	**100.000**	**100.000**	**100.000**	99.300	98.540	99.270	**99.800**	**100.000**	**99.750**	**99.270**	**98.500**

Table 4
Content-preserving manipulations and corresponding parameters.

Operations	Parameters	Number of experiment
Motion blur	Pixel: 1–10	10
Out-of-focus blur	Radius: 1–10	10
Gaussian blur	Size: 3–21	10
JPEG compression	Quality factor: 10–90	10
Salt and pepper noise	Noise factor: 0–0.05	10
Gaussian noise	Noise factor: 0–0.05	10
Rotation	Angle: 5–60	10
Scaling	Scale: 20–200	10

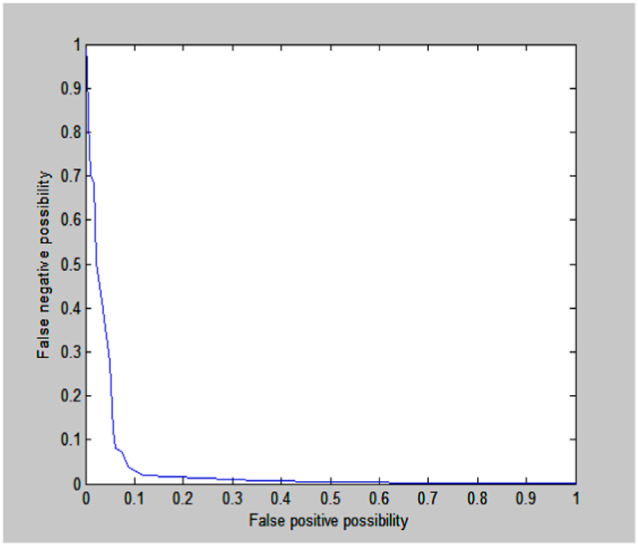


Fig. 7. ROC curve of the proposed method.

$$r_2 = \frac{\text{The size of the correctly detected region}}{\text{The size of tested image}} \times 100\% \quad (33)$$

We estimate the tampering rates and detection rates of the tested images in Figs. 10–17. The results are shown in Table 7. As can be seen from Table 7, the detection rates are close to the tampering rates. This means that the proposed method has better detection precision. The "size" in the formula (32) and formula (33) refers to the number of pixels in the image.

To demonstrate the advantages of the proposed method in detection performance, we compare the visual effect of the proposed method with that of the methods [12], [18], [20] and [22]. In experiments, the tested images come from CASIA database [28]. Fig. 15 shows some examples of the testing results. In the Fig. 15, (c1)-(c5) represent the detection results of the proposed method, (d1)-(d5), (e1)-(e5), (f1)-(f5), (g1)-(g5)

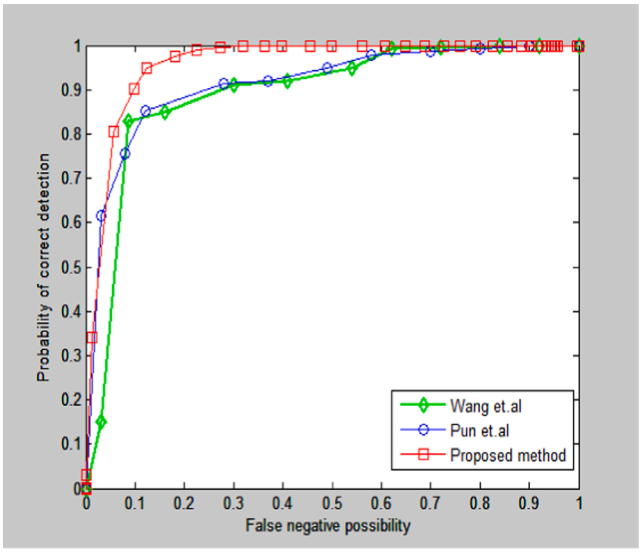


Fig. 8. Comparing result of ROC.

represent the detection results of literature [12], [18], [20] and [22], respectively.

As can be seen from the Fig. 15, the proposed method has better visual effect than that of the compared works. Although works [12,18,22] can achieve tampering localization, the tampering localization results are not accurate, there are large false positives and false negatives. The reason is that these methods are all using block-based image features, and involve sector block and regular block. The work [20] is based on the SLIC algorithm, and can achieve tampering localization at the object-level. However, this method has large false positives. In the proposed method, we use the hybrid image features and a coarse-to-fine grained tampering localization strategy, it can achieve more accurate localization effect.

3.7. Computational complexity analysis

In this section, we estimate the computational time and compare with literatures [12] and [21], including the calculation time of hash generation, tampering detection and tampering localization. In the experiments, we test 1,200 images with different sizes, include 300 images in size 384 × 384, 300 images in size 512 × 512, 300 images in size 832 × 832, and 300 images in size 1200 × 1200. We count the computational time cost on hash generation, tampering detection and tampering localization for each image, then calculate the average time cost for each image according to different sizes. Table 8 shows the statistical average values of the time cost, the unit is second. As can be seen from Table 8, the larger the image size is, the longer the time cost will be. Comparing with the comparative methods, the proposed method is

Table 5
The results of comprehensive performance comparison.

Works	Method	TLC	OLTC	JPEG (Quality factor)	AN (Noise factor)	MB (Pixel)	OFB (Radius)	Rotation (Angle)	Scaling (%)	Hash size (bits)
Lv [1]	Feature point& Shape context	Yes	No	>10	0.01	3	2	30	0.5–1.5	$4np$ (Variable-length)
Wang [12]	Feature point& Image block	Yes	No	>10	0.01	3	2	25	0.5–2.0	$np\cdot\varepsilon 1 + [(M \times N)/p^2]\cdot\varepsilon 2 + 2\,np + 2$ (Variable-length)
Zhao [15]	Zernike moments &Salient region	Yes	No	>30	0.01	3	2	5	0.5–1.5	560
Yan [19]	Image block & QFMMs	Yes	No	>10	0.02	4	2	180	0.6–2	688
Pun [20]	Segmentation&MR	Yes	Yes	>10	0.02	5	2	18	0.6–2	$5(M \times N)/p^2 + 3NG + 3$ (Variable-length)
Pun [22]	Feature point	Yes	No	>10	0.02	5	2	180	0.6–2	2096
Davarzani [32]	Image block& CSLBP	Yes	No	>70	0.05	3	2	5	0.5–1.5	2048
Our scheme	Hybrid feature	Yes	Yes	>20	0.05	5	1.8	25	0.5–2	2631

Table 6
The average values of R_r, R_f and the F_1 score of different size of tampering regions.

The size of the tampering regions	F_1 scores	The size of the tampering regions	F_1 scores
25 × 25	0.3731	125 × 125	0.9107
50 × 50	0.8469	150 × 150	0.9132
75 × 75	0.8890	175 × 175	0.9169
100 × 100	0.9049	200 × 200	0.9223

obviously superior in computing time, and also, the change of image size has little influence to the computing time. This demonstrates that the proposed method is efficient.

3.8. Security analysis

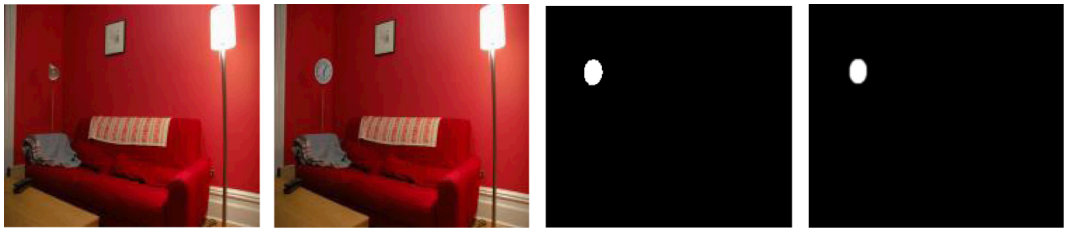
The security of the image hash method mainly refer to content-based security, which means that it can resist forgery attacks, according to the work [12], including: (1) Unauthorized user cannot forge the hash code even if he knows the contents of image; (2) It is impossible to deduce image content or forge image content from the known hash code; (3) Every one cannot use a hash conflict to replace image content with other content. These properties put forward three requirements for hash algorithm. (1) The hash codes should hide the relevance between itself and it associated image content, or, it is difficult to find the relevance between hash code and it associated image content. (2) The hash code should be sufficiently random; (3) The hash code should be collision resistant.

3.8.1. Randomness

We analyze the statistical distribution of the final hash code generated by using the proposed method. According to the literature [27], the linear congruence algorithm can generate a pseudo-random sequences, so the key sequence in our encryption algorithm is a pseudo-random sequence and subject to uniform distribution, and the average value of the key sequence is about 0.5.

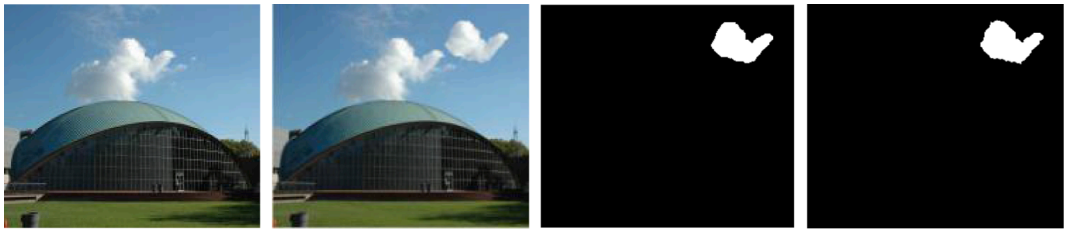
Assume that the probability of each plain text bit (0 or 1) is an independent event. The statistical distribution of the encrypted data is as follows:

Let d_j denote j^{th} plain text bit, and q be the probability that d_j is equal to 1, that is, $P(d_j = 1) = q$. Since the key sequence in our encryption algorithm is subject to uniform distribution, the probability of a bit being equal to 0 or 1 is the same. That is, $P(k_j = 1) = P(k_j = 0) = 0.5$.



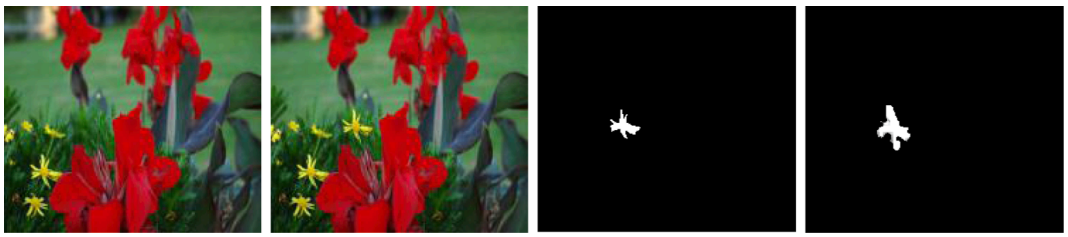
(a1)Original image (a2)Tampered image (a3)Detected result (a4)Ground truth

Fig. 9. Detection results for object replacement attacks.



(b1)Original image (b2)Tampered image (b3)Detected result (b4)Ground truth

Fig. 10. Detection results for adding image object attacks.



(c1)Original image (c2)Tampered image (c3)Detected result (c4)Ground truth

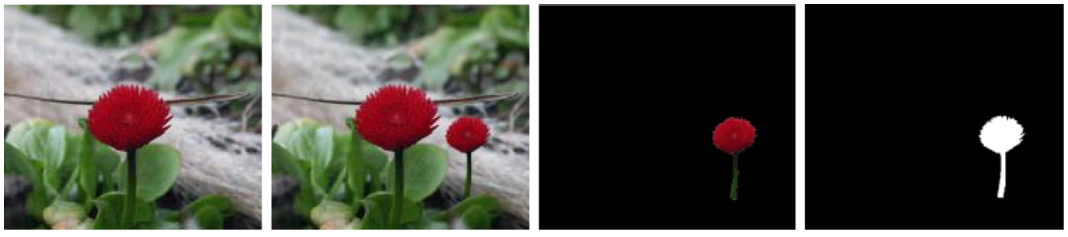
Fig. 11. Detection results for copy-move attacks.

(d1)Original image (d2)Tampered image (d3)Detected result (d4)Ground truth

Fig. 12. Detection results for multiple object insertion attacks.

(e1)Original image (e2)Tampered image (e3)Detected result (e4)Ground truth

Fig. 13. Detection results for adding object and rotating 10 degrees.



(f1)Original image (f2)Tampered image (f3)Detected result (f4)Ground truth

Fig. 14. Detection results for adding image object and scaling 80%.

Therefore, for cipher text, the probability that j^{th} bit is equal to 1 is as follows [12]:

$$P(h_j = 1) = P((d_j = 1)\Lambda(k_j = 0)) + P((d_j = 0)\Lambda(k_j = 1)) = q \times 0.5 + (1 - q) \times 0.5 = 0.5 \tag{34}$$

Here, $k_j(d_j \text{ or } h_j) = 1(\text{or } 0)$ indicates the j^{th} bit of the key sequence is equal to 1(or 0), and $P(h_j = 1)$ represents the probability of $h_j = 1$.

The above analysis shows that after encryption and randomization, the final hash can finally be considered to obey same probability distribution as the key sequence.

3.8.2. Hash collision probability

We evaluate the collision resistance of the proposed method by using the probability of hash collision. In this experiment, we selected 2000 images from the Ground truth Database [29] and the UCID database

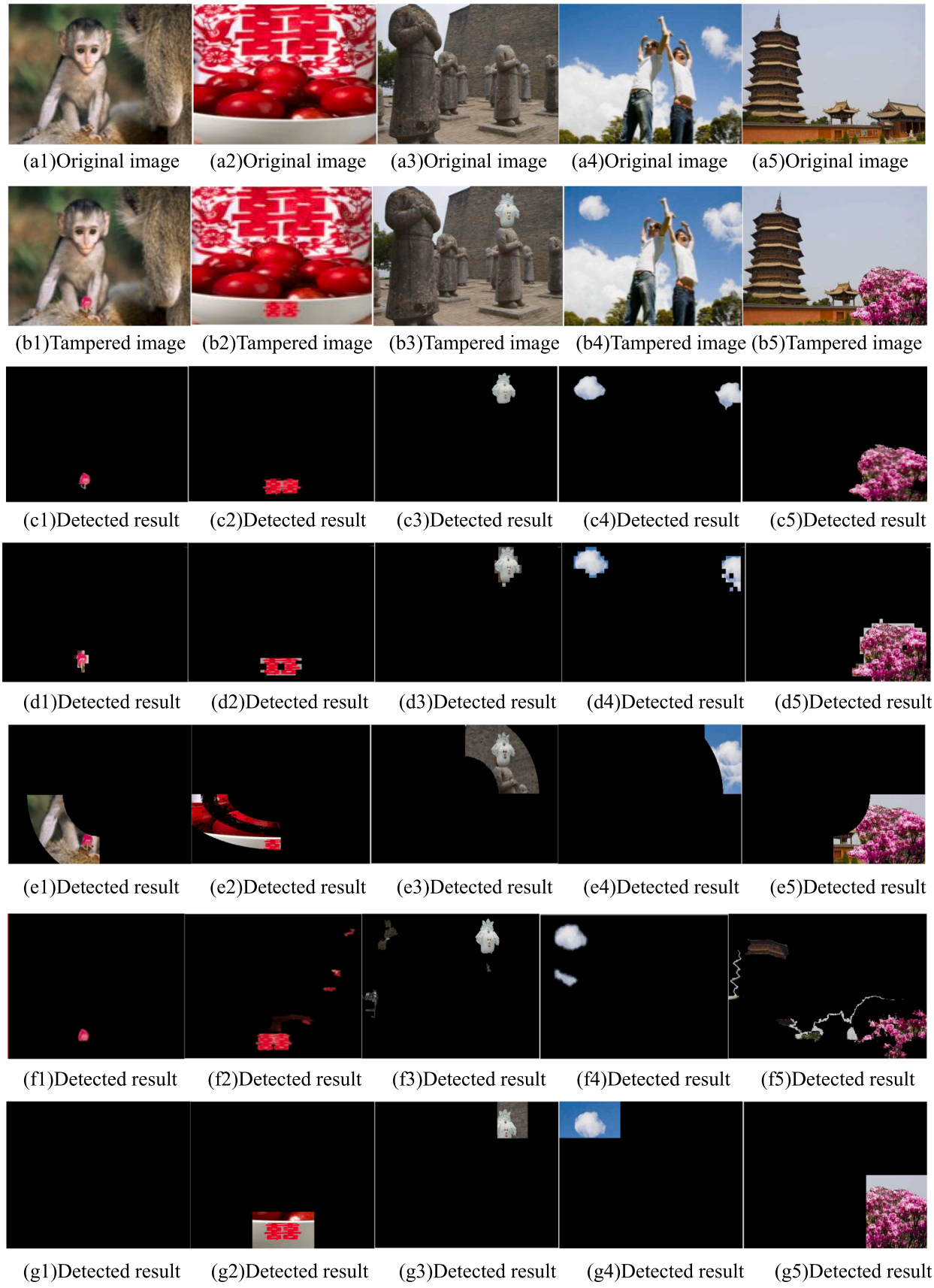


Fig. 15. The comparison of the detecting results.

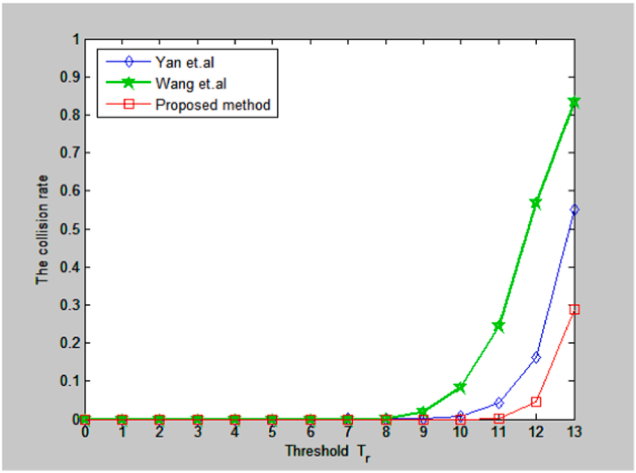


Fig. 16. Comparison of Hash collision rates.

Table 7
The tampering rates and detection rates.

Image Index	a	b	c	d	e	f
r_1(100%)	0.62	2.58	0.87	2.73	9.04	2.50
$r_{2(100\%)}$	0.61	2.72	0.80	3.14	10.04	2.40

Table 8
The computational time and comparison (second).

Image size	384 × 384	512 × 512	832 × 832	1200 × 1200
Work [12]	2.78	4.79	12.56	20.95
Work [21]	10.75	24.36	41.98	52.64
The proposed method	4.98	5.23	5.30	5.47

[30] for testing and comparison. We calculate the Euclidean distance and compare the hash values pairwise. For image I^1, I^2, and their hash values $H(I^1)$ and $H(I^2)$, a hash collisions occurs if and only if the Euclidean distance:

$$D = ||H(I^1) - H(I^2)||_2 \leqslant T_r$$

According to literature [12], the probability of hash collision is defined as follows.

$$\text{The collision rate} = \frac{\text{Image pairs numbers of distance } D \leqslant T_r}{\text{Total pairs numbers of tested images}} \times 100\% \quad (35)$$

We tested the hash collision rates under a series of different thresholds T_r and compared it with the existing methods [12] and [21]. The experiment results are shown in Fig. 16. It can be seen that the proposed method is superior to the comparison methods and achieves the optimal collision rate when $T_r \leqslant 11$.

4. Conclusion

Image hash is an effective technology to solve the problem of image content authentication. It has become one of the hot topics in the field of information security and multimedia application. In this work, we present an image hash method and a hash-based image tampering detection and localization method. In this paper, we have carried out a large number of experiments, and experimental results show that tampering detection precision of the proposed method can reach 92%. The main advantages of the proposed method are as follows: (1) The proposed method is providing with higher detection precision, it can realize object-level tampering detection and localization, and tampering localization precision achieves pixel level. (2) The OTSU method is used to generate system parameters, rather than experimental results or man-made setting. The disadvantage of the proposed method is that the hash code is a bit longer and hash generation time is a bit longer since three types of image features must be extracted.

The proposed method is robust to JPEG compression, noise, filtering and other content-preserving manipulations and geometric deformation. It is very sensitive to changes caused by malicious attacks and achieves a compromise between robustness again geometric distortion and tampering localization. Comparing with the state-of-the-art methods, the proposed method is provided with satisfactory results.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] X.D. Lv, Z.J. Wang, Perceptual image hashing based on shape contexts and local feature points, IEEE Trans. Inf. Forensics and Security 7 (3) (2012) 1081–1093.
[2] J. Fridrich, M. Goljan. Robust hash functions for digital watermarking, in: IEEE Int. Conf. Inf. Coding and Computing, 2000, pp. 178-183.
[3] Z. Tang, S. Wang, X. Zhang, Robust image hashing for tamper detection using non-negative matrix factorization, J. Ubiquitous Converg. Technol 2 (1) (2008) 18–26.
[4] Z. Tang, X. Zhang, S. Zhang, Robust perceptual image hashing based on ring partition and NMF, IEEE Trans. Knowledge and Data Engineering 26 (3) (2014) 711–724.
[5] F. Khelifi, J. Jiang, Perceptual image hashing based on virtual watermark detection, IEEE Trans. Image Processing 19 (4) (2010) 981–994.
[6] Y. Zhang, S. Tang, J. Li, Secure and incidental distortion tolerant digital signature for image authentication, Comput. Sci. Technol. 22 (4) (2007) 618–625.
[7] F. Ahmed, M.Y. Siyal, V.U. Abbas, A secure and robust hash-based scheme for image authentication, Signal Processing, 90 (5) (2010) 1456-1470.
[8] C.S. Lu, H.Y. Liao, Structural digital signature for image authentication: an incidental distortion resistant scheme, IEEE Trans. Multimedia 5 (2) (2003) 2003.
[9] L. Sumalatha, V. Venkata Krishna, V. Vijaya Kumar, Local content based image authentication for tamper localization, Int. J. Image, Graphics and Signal Processing 4 (9) (2012) 30–36.
[10] S. Roy, Q. Sun, Robust Hash for Detecting and Localizing Image Tampering, in: Proc. IEEE Int. Conf. Image Processing, 2007, pp. 117-120.
[11] W. Lu, M. Wu, Multimedia forensic hash based on visual words, in: Proc. IEEE Int. Conf. Image Processing, 2010, pp. 989-992.
[12] X. Wang, K. Pang, X. Zhou, Y. Zhou, L. Li, J. Xue, A visual model based perceptual image hash for content authentication, IEEE Trans. Inf. Forensics and Security 10 (7) (2015) 1336–1349.
[13] D. Gorisse, M. Cord, F. Precioso, Locality-sensitive hashing for Chi2 Distance, IEEE Trans Pattern Anal. Mach. Intell. 34 (2) (2012) 402–409.
[14] W. Lu, A. Varna, M. Wu, Forensic hash for multimedia information, Proc. Media Forensics and Security I I (2010) 18–20.
[15] Y. Zhao, S. Wang, X. Zhang, H. Yao, Robust hashing for image authentication using Zernike moments and local features, IEEE Trans. Information Forensics and Security 8 (1) (2013) 55–63.
[16] S.A. Khavare, A.A. Manjrekar, Robust image hashing algorithm for detecting and localizing image tamper in small region, IEEE Int. Conf. Image Processing 16 (2015) 289–294.
[17] Z. Tang, X. Zhang, X. Li, Robust Image Hashing With Ring Partition and Invariant Vector Distance, IEEE Trans. Inf. Forensics and Security 11 (1) (2015) 200–214.
[18] C.P. Yan, C.M. Pun, X.C. Yuan, Multi-scale image hashing using adaptive local feature extraction for robust tampering detection, Signal Process. 121 (2016) 1–16.
[19] C.P. Yan, C.M. Pun, X.C. Yuan, Quaternion-based image hashing for adaptive tampering localization, IEEE Trans. Inf. Forensics Security 11 (12) (2016) 2664–2677.
[20] C.M. Pun, C.P. Yan, X.C. Yuan, Image alignment based multi-region matching for object-level tampering detection, IEEE Trans. Inf. Forensics Security 12 (2) (2017) 377–391.
[21] C.P. Yan, C.M. Pun, Multi-scale difference map fusion for tamper localization using binary ranking hashing, IEEE Trans. Inf. Forensics and Security 12 (9) (2017) 2144–2158.
[22] C.M. Pun, C.P. Yan, X.C. Yuan, Robust image hashing using progressive feature selection for tampering detection, Multimedia Tools Applications 77 (10) (2018) 11609–11633.
[23] D.G. Lowe, Object recognition from local scale-invariant features, IEEE Computer Society 99 (2) (1999) 1150–1157.
[24] G. P. Nason, B. W. Silverman. The stationary wavelet transform and some statistical applications, Springer, New York, NY, 103 (1995) 281-299.
[25] O. Veksler, Y. Boykov, P. Mehrani, Superpixels and supervoxels in an energy optimization framework, Eur. Conf. Computer vision 6315 (2010) 211–224.

[26] C. Tomasi, R. Manduchi. Bilateral filtering for gray and color images, in IEEE Int. Conf. Computer Vision, 98 (1) (1998).
[27] P. L'ecuyer, Tables of linear congruential generators of different sizes and good lattice structure, Math. Computation Am. Math. Soc. 68 (225) (1999) 249–260.
[28] CASIA, accessed on Mar. 1, 2016. [Online]. Available: http://forensics.idealtest.org/.
[29] http://www.cs.washington.edu/research/imagedatabase/.
[30] G. Schaefer, M. Stich, UCID: an uncompressed color image database, Storage Retrieval Methods Appl. Multimedia 5307 (2004) 472–480.
[31] N. Otsu, A threshold selection method from Gray-level histograms, IEEE Trans Syst., Man, Cybernetics 9 (1) (2007) 62–66.
[32] R. Davarzani, S. Mozaffari, K. Yaghmaie, Perceptual image hashing using center-symmetric local binary patterns, Multimedia Tools Appl. 75 (8) (2016) 4639–4667.
[33] D.A. Huffman, A method for the construction of minimum redundancy codes, Proc. Inst. Radio Eng. 40 (10) (1952) 1098–1101.