applied sciences

MDPI

Article

Image Copy-Move Forgery Detection Based on Fused Features and Density Clustering

Guiwei Fu, Yujin Zhang * and Yongqi Wang

School of Electronic and Electrical Engineering, Shanghai University of Engineering Science, Shanghai 201620, China; fgw9295@163.com (G.F.)
* Correspondence: yjzhang@sues.edu.cn

Abstract: Image copy-move forgery is a common simple tampering technique. To address issues such as high time complexity in most copy-move forgery detection algorithms and difficulty detecting forgeries in smooth regions, this paper proposes an image copy-move forgery detection algorithm based on fused features and density clustering. Firstly, the algorithm combines two detection methods, speeded up robust features (SURF) and accelerated KAZE (A-KAZE), to extract descriptive features by setting a low contrast threshold. Then, the density-based spatial clustering of applications with noise (DBSCAN) algorithm removes mismatched pairs and reduces false positives. To improve the accuracy of forgery localization, the algorithm uses the original image and the image transformed by the affine matrix to compare similarities in the same position in order to locate the forged region. The proposed method was tested on two datasets (Ardizzone and CoMoFoD). The experimental results show that the method effectively improved the accuracy of forgery detection in smooth regions, reduced computational complexity, and exhibited strong robustness against post-processing operations such as rotation, scaling, and noise addition.

Keywords: copy-move; A-KAZE; density clustering algorithm; affine matrix transformation; location similarity

Citation: Fu, G.; Zhang, Y.; Wang, Y. Image Copy-Move Forgery Detection Based on Fused Features and Density Clustering. *Appl. Sci.* **2023**, *13*, 7528. https://doi.org/10.3390/app13137528

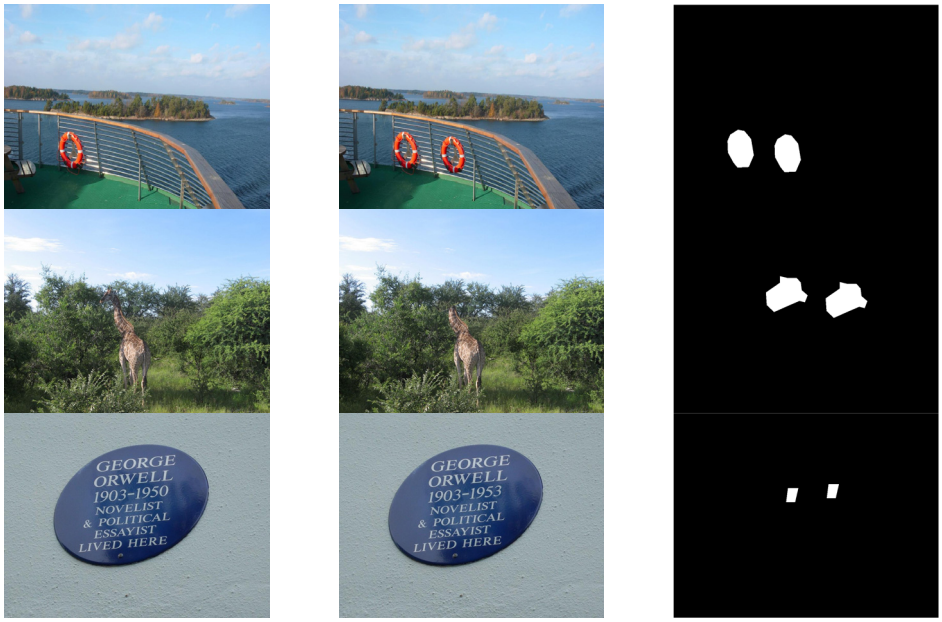
Academic Editors: Héctor Manuel Pérez-Meana and Mariko Nakano-Miyatake

Received: 22 May 2023
Revised: 16 June 2023
Accepted: 24 June 2023
Published: 26 June 2023

1. Introduction

With the rapid development of information technology and the widespread adoption of new computer techniques, digital images have become an important means of obtaining information. People can access high-resolution images and videos through Internet technology, and can capture high-quality images or videos using mobile devices or cameras. Moreover, image processing and editing software such as Adobe Photoshop and Lightroom enable people to manipulate and edit images more easily and quickly. However, not only do these advanced software processing techniques bring convenience, but they also provide criminals with the opportunity to commit crimes. The issue of image tampering and manipulation is common and has caused serious problems in news dissemination, design, social media, and even digital forensic investigation [1]. The credibility and integrity of digital images are especially being questioned in areas such as medical records, scientific publications, celebrity magazines, news reports, political activities, and judicial appraisals. Image copy-move forgery is currently one of the most commonly used methods of tampering [2]. The technique involves copying and pasting a specific region of an image to other regions of the same image for the purpose of emphasizing certain visual information or concealing certain image content. Figure 1 provides an example of image copy-move forgery [3].

In image copy-move forgery, the tampered area may not be identical to the source area, as it often undergoes post-processing operations such as rotation, scaling, edge softening, blurring, noise addition, and JPEG compression. Therefore, the human eye can easily be deceived by tampered images, leading to misdirection [4].



(**a**) (**b**) (**c**)

Figure 1. Image copy-move forgery examples: (**a**) original images; (**b**) forged images; (**c**) binarized truth images.

Many researchers have explored image copy-move forgery detection in detail. Currently, there are two main types of methods to detect copy-move forgery: divide the image into image blocks or extract feature points from the image [5]. It was found that the block-based method, which requires matching a large number of overlapping blocks, is inefficient when it comes to geometric transformation attacks because block features are very sensitive to scale and rotation transformations, which ultimately results in high computational cost. Feature point-based methods are commonly used in copy-move forgery detection because they are unaffected by rotation and scaling, making them highly reliable.

This paper proposes a forgery detection method based on two detection approaches, speeded up robust features (SURF) and accelerated KAZE (A-KAZE), to reduce the computational cost and extract more feature points in smooth regions. The SURF detector has a lower-dimensional feature descriptor and is invariant to rotation and scaling, resulting in faster speed. On the other hand, A-KAZE can extract a large number of useful feature points in smooth regions, allowing for improved accuracy of forgery detection in these regions. After feature matching with g2NN for the two detectors, we adopted the DBSCAN algorithm to eliminate mismatched pairs and locate tampered areas by comparing similarities between the original image and the image transformed by the affine matrix. The experimental results demonstrate that the method proposed in this paper improved the accuracy of smooth region forgery detection by combining the two detection approaches and exhibited strong robustness to geometric transformations such as rotation and scaling.

The rest of this article is arranged as follows: Section 2 provides an overview of the latest research on image copy-move forgery. Section 3 describes SURF and A-KAZE in detail. In Section 4, the proposed image copy-move forgery detection method based on fused features and density clustering is described in detail. Section 5 analyzes the performance of the proposed method and compares it with other algorithms through experiments. Finally, Section 6 summarizes the method proposed in this article.

2. Related Works

At present, copy-move forgery detection (CMFD) is divided into image block-based and feature point-based methods. Using the method of image blocks, the image is typically segmented into overlapping rectangular or circular blocks, from which feature points are

extracted, and then feature matching is conducted. Fridrich et al. [6] first proposed a block-based method that uses CMFD with discrete cosine transform (DCT) in this algorithm. Several features were proposed to describe these blocks. For example, Kumar et al. [7] proposed a method based on principal component analysis (PCA) for image CMFD. This method extracts image features using the Haar transform, simplifies the features through PCA to reduce computational complexity, and then detects, locates, and removes false boundaries. Additionally, the texture features of the input image are analyzed using the gray-level co-occurrence matrix (GLCM). Fattah et al. [8] applied a two-dimensional discrete wavelet transform (DWT) to tampered images, considering only the approximate DWT coefficients for block segmentation, and matching was carried out by computing the distance between pairs of blocks. The results showed that the algorithm was not suitable for forgery detection under geometric transformation attack. Finally, the features were matched using Euclidean distance, and non-matching feature points were identified. Sabeena et al. [9] proposed a method for image CMFD that utilizes local binary patterns (LBP) and Harlick features for feature extraction. In order to verify the authenticity of the image, various supervised machine learning classifiers, such as support vector machines (SVMs), random forest (RF), and gradient boosting classifiers, were used, and the forgery detection performance based on these classifiers was analyzed. In order to address the problem of tampering operations such as rotation and scaling in the tampered area, many local invariant features have been applied in the research of CMFD forensics, such as Zernike moments, Hu moments, and so on [10,11]. Lee et al. utilized statistical features based on a histogram of gradients (HOG) for CMFD [12]. As can be seen from the results, the algorithm's performance in detecting forgeries in the case of rotation and scaling of large regions needs improvement. A perceptual image hashing scheme based on block truncated coding is presented in [13]. In this paper, centrosymmetric local binary patterns are used as image feature descriptors. This method requires a small amount of calculation and is not sensitive to grayscale changes. Wang et al. [14] proposed a CMFD algorithm based on the polar complex exponential transform (PCET). In this algorithm, the suspicious image is first divided into overlapping circular blocks, and PCET is used to extract the geometric invariant features of each block. However, dividing the image into circular blocks results in some loss of image information, which may have a certain impact on the detection performance of the algorithm.

Due to the high computational complexity of block-based methods and their limitations in detecting scale displacement tampering, researchers have shown increasing interest in feature point-based methods for image CMFD to address these problems. These methods first extract feature points from high-entropy regions of the image, then calculate and match feature vectors corresponding to each point, and finally identify tampered regions. The advantage of these methods lies in the point extraction, point description, point matching, and positioning methods applied [15–20]. In order to enhance the sensitivity of copy-move detection algorithms to manipulations such as rotation and scaling, Pun et al. [21] proposed an early feature point-based CMFD method. This method utilizes scale-invariant feature transform (SIFT) [22,23] to extract feature points and applies the random sample consensus (RANSAC) algorithm [24] to remove mismatched pairs through affine transformations. However, the time complexity of this algorithm is relatively high, and it cannot detect multiple types of tampering operations. Subsequently, researchers conducted studies on various tampering operations. However, the location accuracy of this algorithm on pixel-level datasets is still slightly insufficient.

Currently, feature-point-based methods are not effective at detecting tampering in small or smooth areas. To address this issue, Liu et al. [25] proposed a detection method based on k-means clustering. This method divides the image into smooth and complex regions and uses SIFT and sector mask features to detect tampering in each region. However, the accuracy of this algorithm needs to be improved. When compared to methods based on blocks, methods based on feature points have numerous advantages [26]. For instance, the most representative key points in the target region can be retrieved using the feature point

method, enabling more precise detection and positioning of the target region. Feature point extraction methods typically have a certain robustness and can withstand post-processing operations such as rotation and scaling [27].

Compared to block-matching methods, the time required for feature point extraction and matching is usually shorter, resulting in lower computational complexity. However, there are still some issues with feature point-based methods. For example, matching time increases accordingly when the extracted key points are dense, and tampering that occurs in smooth areas is difficult to detect accurately [28].

3. SURF and A-KAZE Descriptors

This section mainly summarizes the feature descriptors SURF and A-KAZE [29].

3.1. SURF Descriptor

At present, the image CMFD algorithms based on feature points are mainly the SIFT and SURF algorithms. The SURF algorithm uses the Hessian matrix for efficient calculations, which overcomes the shortcomings of SIFT in terms of too high feature dimension and long calculation time [30]. The SURF algorithm simplifies the Gaussian second-order differential template in the Hessian determinant, i.e., converts the convolution smoothing operation, to an addition and subtraction operation. Furthermore, the SURF algorithm has better robustness and lower time complexity compared to the SIFT algorithm. SURF cannot maintain the scale and rotation invariance of SIFT, but it has strong robustness to changes in lighting and affine distortion [31]. SURF determines the key points by computing the extreme points of the scale space through calculating the relevant Hessian matrix. The Hessian matrix $H(x, \sigma)$ can be represented as:

$$H(x,\sigma) = \begin{bmatrix} C_{xx}(x,\sigma) & C_{xy}(x,\sigma) \\ C_{xy}(x,\sigma) & C_{yy}(x,\sigma) \end{bmatrix} \tag{1}$$

In the formula, $x = (x, y)$, $C_{xx}(x, \sigma)$ is the convolution of the second-order partial derivative $\partial G(x, y, \sigma)/\partial x^2$ of the Gaussian function and the image at the pixel point. The determinant of matrix $H(x, \sigma)$ is expressed as:

$$\det(H) = C_{xx}(x,\sigma) \cdot C_{yy}(x,\sigma) - \left[C_{xy}(x,\sigma)\right]^2 \tag{2}$$

Due to the high time complexity of computing second-order partial derivatives of images, SURF uses rectangular box filters to approximate the second-order partial derivatives of Gaussian functions. The box filter is composed of a simple rectangular template, which speeds up the convolution calculation and reduces the time complexity. Filtering the image with the rectangular box filter can obtain $C_{xx}(x, \sigma)$, $C_{xy}(x, \sigma)$, and $C_{yy}(x, \sigma)$, and the Hessian matrix determinant for each coordinate point can be calculated to obtain the Hessian determinant image. By changing the size of the rectangular box-shaped filter and the scale of the Gaussian function σ, scale changes are achieved, thus generating multiple Hessian determinant images and constructing a pyramid image. Using non-maximum suppression to filter key points, we can compare each point with eight neighboring points in the same scale and 18 neighboring points in adjacent scales. If the value of the point is greater than the value of its 26 neighbors and the preset threshold T_S (subscript $_S$ indicates SURF), the point is determined as a key point.

The direction of SURF feature points depends on the response of a circular region with a radius of 6 centered on the key point. With the key point as the center, a sector region with an angle of $\pi/3$ is scanned within the circular region to calculate the total horizontal and vertical Harr feature points of all points in the sector region. The longest vector direction is selected as the main direction of the key point. After determining the direction of the key point, a SURF descriptor is generated. Specifically, a square region in the neighborhood with a side length of 20σ is selected around and aligned with the

main direction of the key point. The square region is divided into 4×4 sub-regions, and 4-dimensional features are calculated for each of the $5\sigma \times 5\sigma$ grids, including the horizontal and vertical responses of the Harr wavelet in directions $\sum dx$ and $\sum dy$, as well as the absolute values of the responses in directions $\sum |dx|$ and $\sum |dy|$. Finally, the 4-dimensional features of each sub-region are accumulated into a 64-dimensional feature vector descriptor by dividing it into 16 sub-regions.

3.2. A-KAZE Descriptor

A-KAZE is a feature-based image matching algorithm used for tasks such as image feature detection, matching, and object tracking. The A-KAZE algorithm was developed from the KAZE algorithm and employs a new feature detection method called accelerated scale space extrema detection (ASSED), which can detect local extrema points in scale space faster [32]. ASSED searches for local extrema points by comparing the Laplacian of the Gaussian transformation at different scales. In smooth areas, the response of the Gaussian Laplacian transformation is small, so only local extrema points can be detected in larger scale spaces. The design of ASSED enables the A-KAZE algorithm to detect local extrema points in larger scale spaces quickly, thus extracting features in smooth regions. A-KAZE's feature descriptor includes multi-scale and multi-directional features. The feature descriptor, particularly for smooth regions, has distinct response values at different scales, making it robust at various scales and enabling it to extract features in smooth regions. At the same time, the feature descriptor can also calculate the main direction, making them invariant to orientation and allowing them to match the same feature points at multiple rotation angles, enhancing feature stability and discriminability.

In the A-KAZE feature detection method, a non-linear scale space is constructed using non-linear diffusion filtering. Non-linear diffusion filtering considers the changes in pixel intensity at different scales as the divergence of some form of flow function. The classical non-linear diffusion formula is defined as:

$$\frac{\partial L}{\partial t} = div[c(x, y, t) \cdot \nabla L] \tag{3}$$

In the formula, div represents the divergence operator, ∇ represents the gradient operator, L represents the brightness information of the image, and $c(x, y, t)$ represents the transfer function, which can be expressed as:

$$c(x, y, t) = g(|\nabla L_\sigma(x, y, t)|) \tag{4}$$

where ∇L_σ represents the gradient of image L_σ after Gaussian smoothing and time t is used as a scale parameter; the larger its value, the simpler the representation of the image. In this article, the conduction function used to extract key points with A-KAZE detector is [33]:

$$g_2 = \frac{1}{1 + \frac{|\nabla L_\sigma|^2}{\lambda^2}} \tag{5}$$

In the formula, λ is the contrast factor that controls the level of diffusion; the larger its value, the less corresponding edge information will be retained. The fast explicit diffusion (FED) algorithm is used to build an image pyramid and obtain one group of images, each with two layers. Then, group O and layer S are mapped to scale σ through the corresponding formula:

$$\sigma_i(o, s) = 2^{o + \frac{s}{S}}, o = 0, 1, \ldots, O - 1, s = 0, 1, \ldots, S - 1, i = 0, 1, \ldots, M - 1 \tag{6}$$

where M is the number of images in the image pyramid, and $M = O \times S$. The unit of linear filtering is scale σ_i, and the unit of non-linear diffusion filtering is time. The transformation of $\sigma_i \rightarrow t_i$ is realized by the following formula:

$$t_i = \frac{1}{2}\sigma_i^2, i = 0, 1, \ldots, M-1 \tag{7}$$

where t_i is evolutionary time. A set of evolutionary time values is obtained through this mapping, and a non-linear scale space is constructed through these time values.

The key point detection of A-KAZE is similar to SURF; it calculates the Hessian determinant of image L^i in the non-linear scale space as follows:

$$L^i_{Hessian} = \sigma^2_{i.n}\left(L^i_{xx}L^i_{yy} - L^i_{xy}L^i_{xy}\right) \tag{8}$$

In the formula, $\sigma_{i.n}$ represents the scale-normalized scaling factor, and $\sigma_{i.n} = \sigma^i/2^{o^i}$. A-KAZE determines the key points in the same way as SURF.

However, unlike SURF, A-KAZE computes all first-order differentials within $\pi/3$ circular sectors and applies Gaussian weighting to make the response near the key points most useful and the response far from the key points contribute the least. The response values within the $\pi/3$ circular sectors are then vector-summed, and the angle of the maximum vector over the entire circular region is the main orientation. A-KAZE uses M-LDB as the descriptor of the key point. LDB divides a $q \times q$ square region centered on the key point into $p \times p$ sub-squares, extracts the mean and gradient information of pixel intensity (grayscale) within each sub-square, and performs binary testing, as follows:

$$\tau\left(F_{function}(i), F_{function}(j)\right) = \begin{cases} 1, \left(F_{function}(i) - F_{function}(j)\right) > 0, i \neq j \\ 0, other \end{cases} \tag{9}$$

where $F_{function}(\cdot)$ represents the function that extracts the mean and gradient information of pixel intensity, defined as:

$$F_{function}(\cdot) = \left\{F_{intensity}(\cdot), F_{dx}(\cdot), F_{dy}(\cdot)\right\} \tag{10}$$

$$F_{intensity}(i) = \frac{1}{m}\sum_{k=1}^{m} Intensity(k), F_{dx}(i) = Gradient_x(i), F_{dy}(i) = Gradient_y(i) \tag{11}$$

In the formula, $F_{intensity}(i)$ represents the average pixel intensity information of the ith sub-box, m represents the number of pixels in the ith sub-box, and $F_{dx}(i)$ and $F_{dy}(i)$ represent the gradient information of the ith sub-box in the x and y directions, respectively.

Later, we rotated the sub-boxes to the main direction and adaptively down-sampled them based on the scale space where the feature points were located in the sub-box. This greatly improved the robustness of M-LDB, because it is no longer necessary to calculate the mean of all pixel intensity information within the sub-box.

4. The Proposed Scheme

The general flow of the method proposed in this article is shown in Figure 2, which includes first using two detection methods, SURF and A-KAZE, to extract feature points from the input image. To solve the problem of insufficient feature point extraction in smooth areas, we lowered the contrast threshold and combined the two detection methods to extract feature points. Then, g2NN is used for feature matching to find similar feature vectors. To improve the accuracy of tampering localization, we used the DBSCAN algorithm to remove false matches. Finally, the tampered area is located by comparing the similarity between the original image and the image after affine transformation.

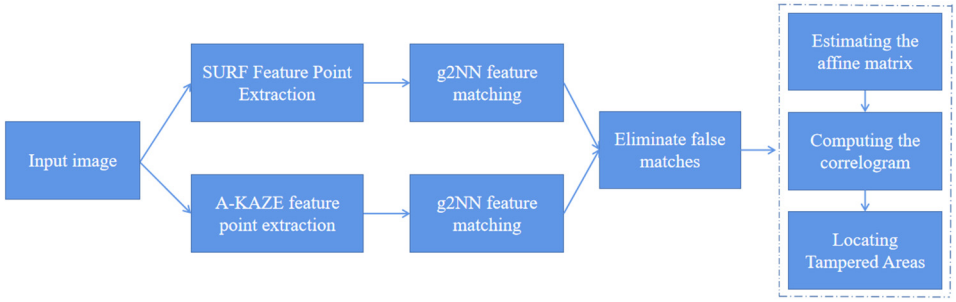


Figure 2. Flowchart of method proposed in this paper.

4.1. SURF and A-KAZE Feature Extraction

To address the issues of difficult detection and high computational complexity that occur in smooth regions with tampering at the present stage, we combined two detection methods, SURF and A-KAZE. Specifically, these methods are used to extract feature points from tampered images, and descriptions of them are provided in Section 3 [34].

When using SURF to extract feature points from an image, we set the contrast to P_s and store the extracted feature points in matrix $x_s = \{x_1, x_2, \ldots, x_n\}$. To better represent the coordinates of each feature point, the coordinates of the ith feature point are represented as $x_s^i = (x_x^i, y_s^i)$. When extracting feature points from an image using A-KAZE, the contrast threshold is set to P_A and the extracted feature points are stored in matrix $x_A = \{x_1, x_2, \ldots, x_m\}$. To better represent the coordinates of each feature point, the coordinates of the ith feature point are represented as $x_A^i = (x_x^i, y_s^i)$. Then, SURF feature matrix $f_s = \{f_1, f_2, \ldots, f_n\}$ and A-KAZE matrix $f_A = \{f_1, f_2, \ldots, f_m\}$ are calculated separately. In this article, contrast threshold P_s is set to 0.1 and contrast threshold P_A is set to 0.0001.

4.2. g2NN Feature Matching

After extracting feature points in the previous stage, g2NN is used to perform feature matching on SURF feature $f_s = \{f_1, f_2, \ldots, f_n\}$ and A-KAZE feature $f_A = \{f_1, f_2, \ldots, f_m\}$ to search for similar feature vectors [35]. Assuming there are n features, $f = \{f_1, f_2, \ldots, f_n\}$, the Euclidean distance is used to determine the similarity between two feature vectors. For the ith key point, its corresponding feature is represented by f_i, and the Euclidean distance between f_i and the remaining features $\{f_1, f_2, \ldots, f_{i-1}, f_{i+1}, \ldots, f_n\}$ is calculated. The results are sorted in ascending order to obtain $D = \{d_1, d_2, \ldots, d_{n-1}\}$. In D, the ratio of d_j and d_{j+1} is calculated and evaluated using threshold V_t, as shown in the following formula:

$$\frac{d_j}{d_{j+1}} < V_t, V_t \in (0,1), j = 1, 2, \ldots, n-1 \tag{12}$$

When the iteration reaches step $k+1$, the process of this iteration will terminate, and key point k will be considered as a candidate similar point matching the ith key point. The value of V_t is 0.6. In the algorithm in this section, the k-d tree and nearest neighbor algorithm are used to search for the N_{kd} points closest to the key point. g2NN feature matching is then used on these points to find similar feature vectors to obtain the matching key point pair matrix x_w. However, there are still some mismatches in the obtained matrix x_w, which requires removing the mismatched pairs.

4.3. Removing Mismatched Pairs

Matching point pairs were obtained in the previous step, but many of them contain errors. At this point, an effective method is needed to eliminate the mismatched pairs in order to improve the accuracy of forgery detection and localization. Currently, the k-means clustering algorithm or the hierarchical clustering algorithm are commonly used to eliminate mismatches, but the problem is that they only consider the positional information of the matching point pairs and ignore the constraints between the point pairs. The number

of clusters needs to be set before clustering. Ester et al. [36] proposed a density-based clustering algorithm, DBSCAN, which considers clusters as sets of maximum density-connected points and partitions the dense areas accordingly. DBSCAN does not require the extraction or setting of clustering parameters and can handle clusters of any shape even in the presence of noise. Therefore, in this paper, DBSCAN is used for feature clustering to eliminate mismatched pairs.

DBSCAN requires setting two important parameters, Eps and MinPts, respectively denoting the radius of density and the threshold of core points, which is the minimum number of points required to form a cluster. In this paper, Eps and MinPts are represented as τ and M, respectively. In DBSCAN, point sets are defined based on the following three categories, as shown in Figure 3:

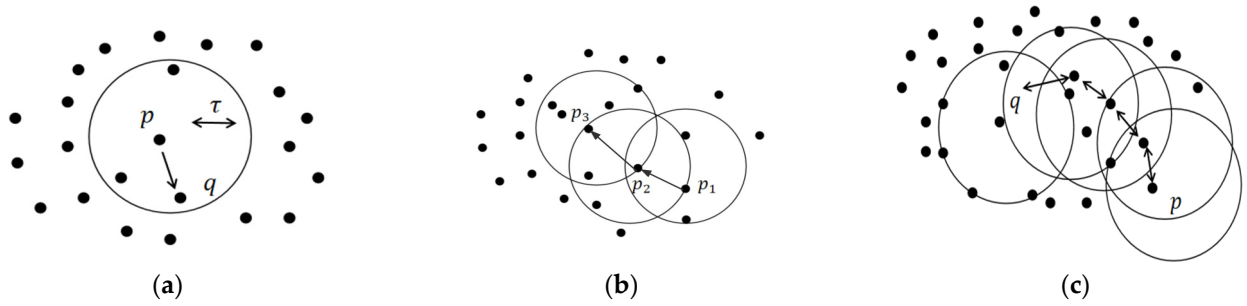


Figure 3. DBSCAN point-to-point relationship diagram: (**a**) density direct; (**b**) density reachable; (**c**) density connected.

1. Core point: If point X_i has at least M points in its neighborhood, then X_i is a core point. We can define the sets of all core points as P_c and the set of non-core points as P_{nc}.
2. Boundary point: If X_i satisfies $X_i \in P_{nc}$ and is in the τ neighborhood of a core point, then X_i is a boundary point. Boundary points can be located within the τ neighborhood of one or more core points at the same time.
3. Outlier: This is neither a core point nor a boundary point.

The relationships between points are classified into three types: density direct, density reachable, and density connected, as shown in Figure 3. The main process of the DBSCAN algorithm is to classify the input points into different categories, iteratively search for density-reachable objects of core points, form a new cluster, and then produce the final clustering result through density connectivity. Figure 3 illustrates the relationships between points, where density reachable is a transitive relationship but not symmetric. Two boundary points in the same cluster may not be density reachable to each other because the core point condition may not be satisfied for both of them. However, there must be at least one core point within a cluster from which all points are density reachable. In Figure 3c, both boundary points are density reachable. Therefore, we introduce the concept of density connectedness, which encompasses this relationship between boundary points. The steps to transition from density reachable to density connected are as follows: First, we calculate the density of each region and label them as density reachable based on a set density threshold. At this stage, we focus on the internal density of each region. Building upon density reachable, we further consider the adjacency relationships between regions. If two density reachable regions have sufficient neighboring regions that can be connected through density reachability, they are considered density connected. In this stage, we emphasize the connectivity between regions.

Specifically, the goal of the algorithm is to divide set P into k effective clusters (where k is obtained through the algorithm and does not need to be set in advance) and a cluster of outliers. The calculation of the clustering identification array m_i is as follows:

$$m_i = \begin{cases} j(j > 0), if\ X_i\ belongs\ to\ cluster\ j \\ -1, if\ X_i\ is\ an\ outlier \end{cases} \tag{13}$$

The goal of DBSCAN is transformed into generating clustering identification array m_i, where $i = 1, 2, \ldots, n$ and k is the number of all non-negative values in the $\{m_i\}_{i=1}^{n}$ types (negative values represent outliers). The outliers in clustering identification array m_i are mismatched points. Finally, removing outliers produces an accurate matching sequence.

4.4. Tampered Region Localization

To further improve the accuracy of tampered region localization, in this paper we compute the affine matrix of matching pairs and apply this transformation to the dense regions of the image at the pixel level. The final tampered region is located by comparing the similarity between the transformed image and the original image at the same unknown location.

Estimation of affine transformation: An affine transformation matrix can reveal geometric changes that occur between objects, such as rotation, scaling, and so on. Based on feature point matching, the affine transformation between original and tampered regions can be estimated using the coordinates of the matching pairs. Given two corresponding sets of matching pair coordinates $X = (x, y)$ and $\widetilde{X} = (\widetilde{x}, \widetilde{y})$, their relationship is shown as follows:

$$\begin{pmatrix} \widetilde{x} \\ \widetilde{y} \\ 1 \end{pmatrix} = \begin{pmatrix} a & b & t_x \\ c & d & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = T \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{14}$$

In the formula, t_x and t_y are translation parameters, while a, b, c, and d are scale and rotation transformation parameters. Matrix T can be calculated using at least three non-collinear matching pairs.

The set of matching pairs after removing the mismatched pairs is denoted as $\left\{\left(X_1, \widetilde{X_1}\right), \left(X_2, \widetilde{X_2}\right), \ldots, \left(X_n, \widetilde{X_n}\right)\right\}$. Three points are randomly selected from each cluster, their corresponding points are found in the matching list, and an affine transformation matrix T_i is estimated from the three pairs of points. All matrices are then computed, and an optimal matrix T is found that minimizes the error, as shown in the following formula:

$$\underset{T}{\operatorname{argmin}} \left\| \widetilde{X} - T_i X \right\|^2 \tag{15}$$

Using the obtained optimal affine matrix T, the transformation is applied to all pixels in the entire image. Then, the transformed image is overlaid with the original image, with the original region Y_O overlapping the tampered region Y_D. Similarly, the inverse transformation is applied to region T^{-1}, with the tampered region Y_D overlapping the original region Y_O in the same way.

$$Y_D = TY_O, Y_O = T^{-1}Y_D \tag{16}$$

To locate the tampered regions, we utilize the polar cosine transform (PCT) feature as the similarity measure and compute it for corresponding positions in both the original and transformed images. When the difference between the two features is less than a certain threshold, the location is marked to obtain a region-of-interest map. The PCT feature calculation of the n-order continuous image g for l consecutive times is as follows:

$$f = \left\{ \left| M_{n,l} \right| such\ that\ n + l \leq 3, 0 \leq n, l < 3 \right\} \tag{17}$$

In the formula:

$$M_{n,l} = \Omega_n \int_0^{2\pi} \int_0^1 \left[H_{n,l}(r, \theta) \right] \times g(r, \theta) r dr d\theta \tag{18}$$

$$\Omega_n = \begin{cases} 1/\pi n = 0 \\ 2/\pi\ n \neq 0 \end{cases}, \ H_{n,l}(r, \theta) = \cos\left(\pi n r^2\right) e^{jl\theta} \tag{19}$$

where $g(r, \theta)$ is the image representation in polar coordinates.

Using a threshold value t to compare the feature differences in the overlapping areas, the region is identified, and the final correlation map is obtained. Then, a Gaussian filter is applied with a window size of 7×7 and a standard deviation of 0.5 for denoising and detail processing. Finally, the closing operation is performed to obtain the final detection result.

5. Experimental Results

This section describes the experimental tests conducted on public image datasets, and the comparative analysis performed using evaluation indicators and detection effect diagrams. This section verifies the advantages of the proposed method in terms of accuracy and robustness against geometric transformations and shows that it is comparable to existing research methods.

5.1. Datasets

In our experiments, we used two challenging datasets for evaluation: Ardizzone [37] and CoMoFoD.

5.1.1. Ardizzone

This dataset is composed of medium-sized images, most of them with a size of either 1000×700 or 700×1000 pixels. The Ardizzone dataset is divided into several subsets: Subset D0 contains 50 images with simple tampering that only involves basic translation transformations; subsets D1 and D2 contain tampered images with additional rotations and scaling transformations; and subset D3 contains the corresponding original images of the tampered images in D0.

5.1.2. CoMoFoD

This dataset contains 200 original images with a size of 512×512 pixels that have undergone various geometric transformations, which are divided into five categories: scaling, rotation, translation, deformation, and combination. Each category contains 40 examples of tampered images, with different tampering methods applied to each category, including noise, blur, JPEG compression, rotation, and scaling. The CoMoFoD dataset contains a total of 10,400 images, in which the area of the replicated region ranges from 0.14 to 14.3% of the image area.

Figure 4 displays original images, tampered images, and corresponding binary ground truth maps from both datasets.

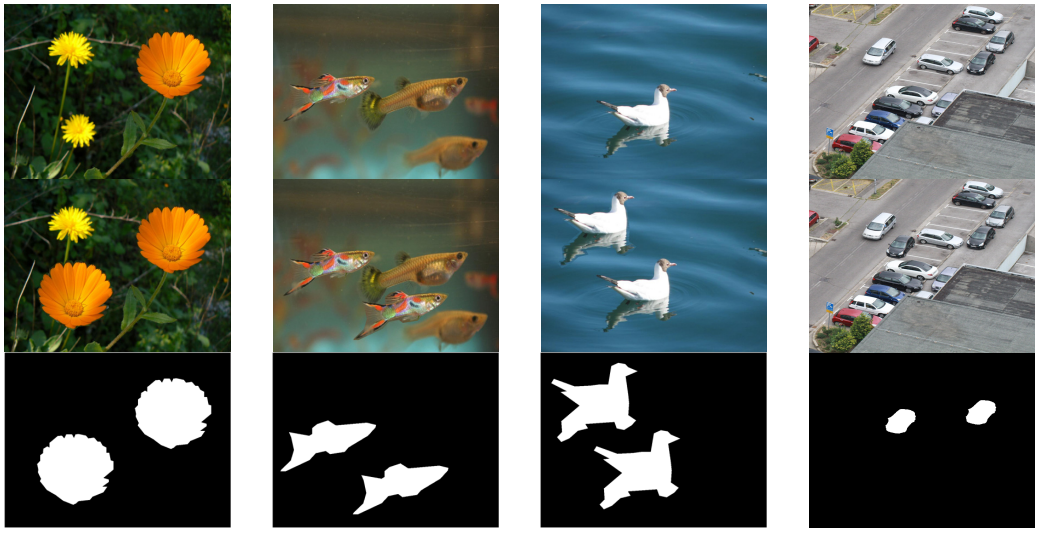


Figure 4. Examples of image copy-move forgery in the dataset. First row: original images; second row: forged images; third row: binarized truth images.

5.2. Evaluation Metrics

The three most commonly used evaluation metrics—precision, recall, and F_1 score—are used in this paper to comprehensively analyze the detection performance of the proposed method. These evaluation metrics are represented by P, R, and F_1, respectively, and their specific formulas are as follows:

$$P = \frac{N_{FF}}{N_{FF} + N_{AF}}, R = \frac{N_{FF}}{N_{FF} + N_{FA}}, F_1 = 2 \times \frac{P \times R}{P + R} \tag{20}$$

where N_{FF} represents the number of tampered images/pixels that are considered to be tampered; N_{AF} represents the number of real images/pixels that are considered to be tampered; and N_{FA} represents the number of tampered images/pixels that are determined to be real. On the basis of comparison with other relevant algorithms, the superiority of the proposed method can be objectively evaluated. This approach not only relies on intuitive detection results, it also enables a comprehensive evaluation of the proposed method through objective data.

5.3. Lab Environment

The experimental environment was Core i5-6200U CPU (2.30 GHz), 8 GB memory, and the software environment was Windows 10 + MATLAB R2019a.

5.4. Copy-Move Forgery Detection

This section presents the experimental detection results. The effectiveness of the algorithm was validated by comparing the detection results with the given binary ground truth image in the dataset. The first row of each detection result represents the tampered image, the second row represents the binary ground truth image in the dataset, which indicates the real tampered region, and the third row represents the experimental detection result of the method in this paper. The blue area represents the correctly detected tampered region, the red area represents the falsely detected region, the white area represents the missed detection region, the dark gray area represents the real tampered region, and the light gray area represents the tampered region after boundary post-processing. In order to verify the effectiveness of the algorithm, two comparison algorithms were used to comprehensively evaluate the detection performance of the proposed method. These were the feature point method combining SIFT and KAZE, proposed by Aydan et al. [38], followed by the feature extraction method using SURF proposed by Badr et al. [17]. The experimental comparison results are presented in tables or figures.

Compared with block-based methods, the feature point extraction stage is extremely important in feature point-based methods, especially the number of feature points extracted in smooth areas. In this experiment, we tested SIFT, SURF, A-KAZE, and BRISK. For traditional methods, a contrast threshold does not need to be set in advance during feature extraction, and feature matching can be completed using functions in the OpenCV database. After wrong matches have been eliminated, the set of matched pairs is then counted, as shown in Table 1. We selected photos from the dataset for testing; the images in the first and second columns came from the Ardizzone dataset, and those in the third and fourth columns came from the CoMoFoD dataset. Among them, the first and third columns show occlusion tampering on the original image, and the second and fourth columns show addition tampering on the original image. Specific test photos are shown in Figure 5.

Table 1 presents the experimental results for the four columns of images in Figure 5, clearly showing that the proposed SURF and A-KAZE feature points have significant advantages compared to other feature points. Compared to some traditional feature points that do not require setting a contrast threshold, a small contrast threshold was set in the method proposed in this paper and SURF and A-KAZE were fused to obtain more matching feature points in the feature extraction stage. Especially for the test image in the first column of Figure 5, where tampering occurs in the smooth area of the sky, only the proposed method detects the correct matching pairs, while other feature extraction

methods do not detect any tampering. This leads to the tampering operations in the smooth area being ignored, resulting in reduced detection effectiveness.

Table 1. Number of matching points of different feature points in tampered area.

Feature \ Row	First Row	Second Row	Third Row	Fourth Row
SIFT	0	16	215	16
SURF	0	2	124	5
A-KAZE	0	23	175	5
BRISK	0	17	96	4
SURF + A-KAZE	98	145	325	18

(**a**) (**b**) (**c**) (**d**)

Figure 5. Experimental photos for detecting the number of matching points of different key point features in the tampered area: (**a**) obscured images; (**b**) tampered images; (**c**) obscured images; (**d**) tampered images.

5.4.1. Geometric Transformation Forgery Detection

This paper mainly focuses on the experimentation and testing of geometric transformation forgery, including simple translation forgery detection, rotation forgery detection and scaling forgery detection. A certain number of photos were selected from the dataset for each type of tampering, and the experimental detection results are presented in the form of detection effect images.

Simple translation forgery detection: We conducted experimental tests on geometric transformation forgery, including simple translation forgery detection, rotation forgery detection, and scaling forgery detection. For each type of tampering, a certain number of photos were selected from the dataset for experiments, and the results of the experimental detection are displayed. Partial experimental results are given below, and are shown in Figure 6. The first row of images in the figure represent the original image, the second row comprises the binarized truth images provided in the dataset (used for comparison with the detection results), and the third row presents the effect images. The third row shows the detection results of the proposed algorithm, which was mainly used to test for simple translation forgery operations. From Figure 6, it can be intuitively seen that the experimental effect image of the proposed method almost coincides with the binary image given in the dataset, and there are no missed detection areas. For images that have not undergone complex tampering, the proposed method can more accurately locate tampered areas, and there is no occurrence of false detection or missed detection in simple tampering.

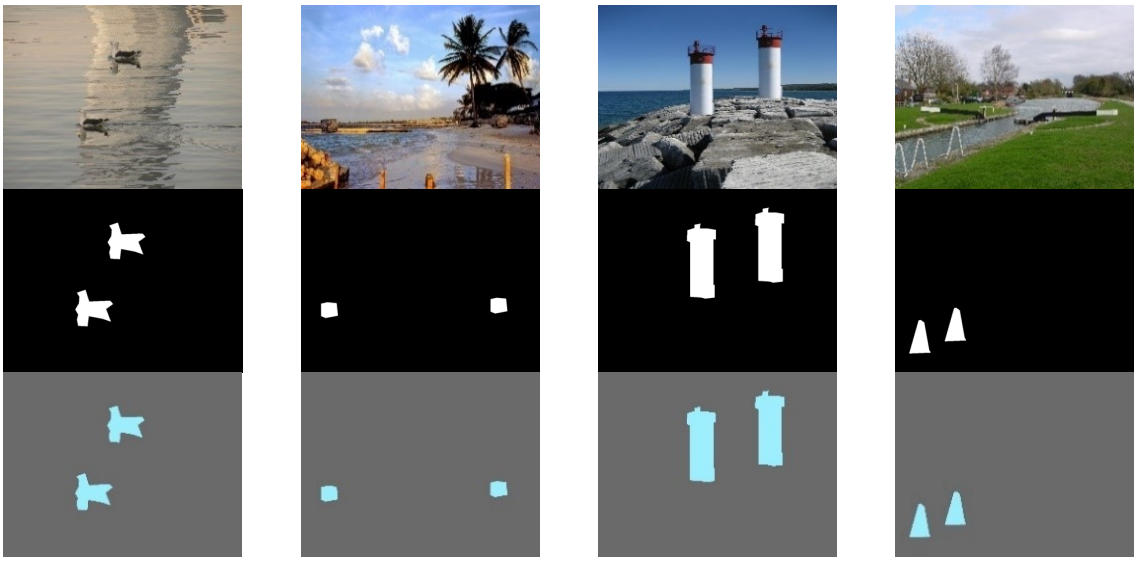


Figure 6. Experimental results of simple translational forgery detection. **First row**: original images; **second row**: binarized truth images; **third row**: effect images of forgery detection.

Rotation transform forgery detection: We conducted forgery detection experiments on images from two publicly available datasets, with the images subjected to rotation-based tampering at angles of $\pm 1^{\circ}$, $\pm 2^{\circ}$, $\pm 3^{\circ}$, $\pm 4^{\circ}$, $\pm 5^{\circ}$, and $\pm 10^{\circ}$. The images in Figure 7 demonstrate the detection results for rotation angles of 1°, 3°, 5°, and 10°. The first row of the figure shows the original images, the second row shows the binarized truth images provided in the dataset (used for comparison with the detection algorithm), and the third row shows the detection results of the method proposed in this paper; the detection results are shown at different rotation angles. The comparison with the binarized truth images from the dataset indicates that the proposed method maintains high detection accuracy even at different rotation angles. The comparison with the binarized truth images in the second row shows that there are almost no missed detections. This advantage is particularly evident in small-scale rotations, and the proposed method has good detection performance.

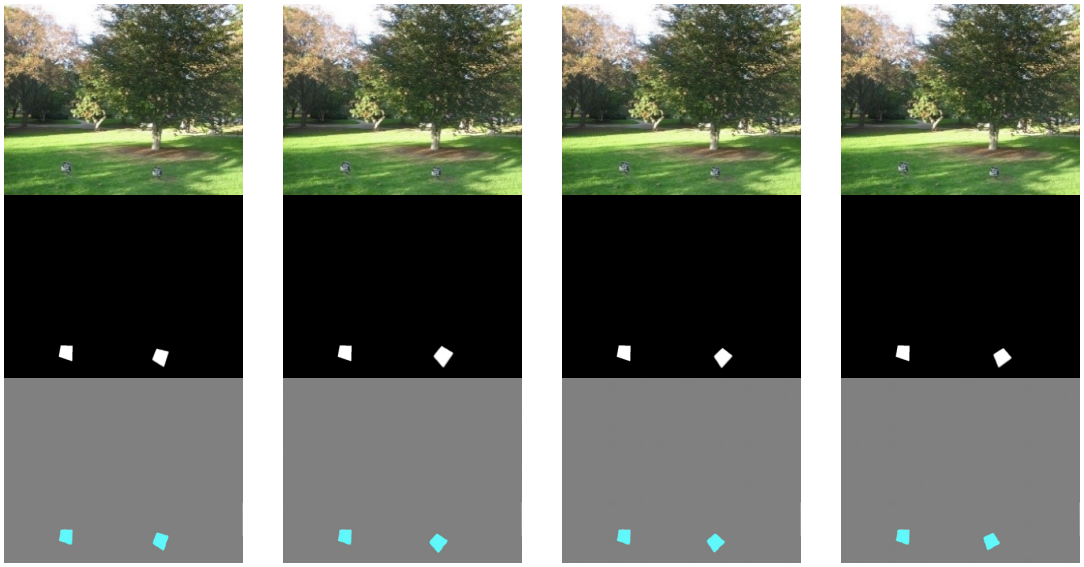


Figure 7. Rotation angle forgery detection experiment results. **First row**: original images; **second row**: binarized truth images; **third row**: effect images of forgery detection.

Scale transform forgery detection: We conducted scale manipulation experiments on images from two publicly available datasets. The scaling factors used were 0.9, 0.95, 1.05, 1.25, and 1.5. Figure 8 presents the experimental results for these scaling factors, from left to right. The first row represents the original image, the second row shows the binary ground truth provided in the dataset, and the third row displays the experimental results of the proposed method.

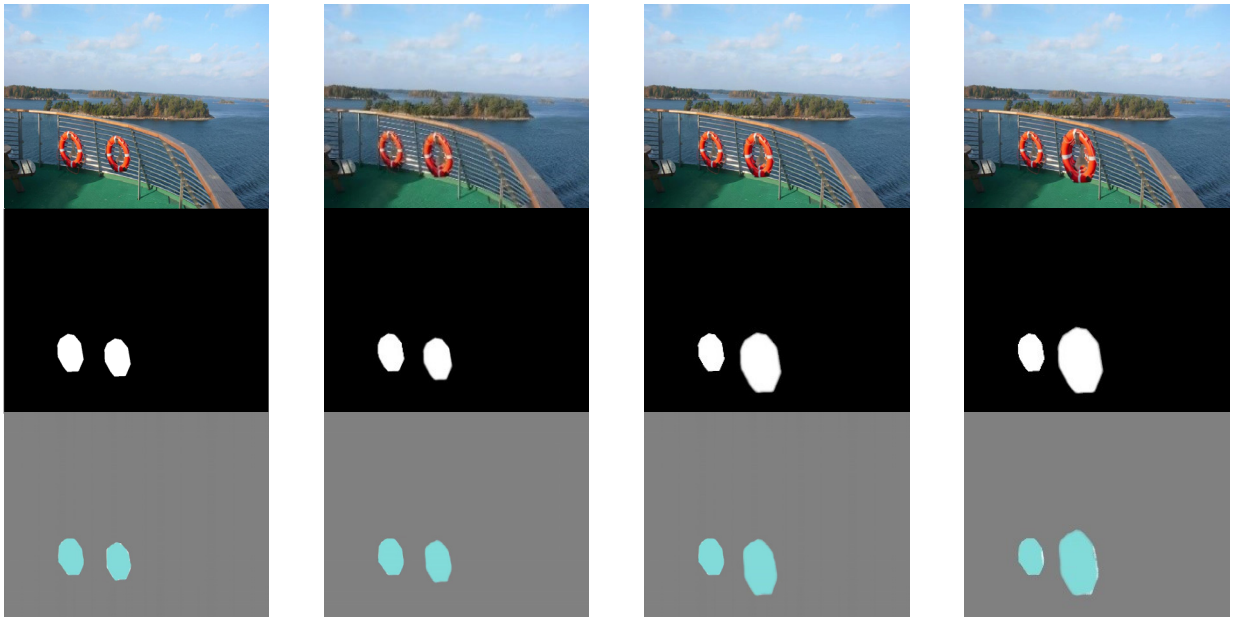


Figure 8. Different scaling factor transformations falsify experimental results. **First row**: original images; **second row**: binarized truth images; **third row**: effect images of forgery detection.

By comparing the detection results in the figure (third row) with the binary images provided in the dataset (second row), it is clear that the proposed method still has good detection performance in scaling forgery detection. Especially for small compression factors, the detection results almost have no missed detection areas. However, as the scaling factor increases, for example, at a scaling factor of 1.5, it can be observed that there are missed detection areas in the boundary of the detection results.

In summary, after conducting experiments on simple translation forgery, rotation forgery at different angles, and compression factor forgery, it can be seen by comparison with the binary images provided in the dataset that the method proposed in this paper is an effective detection algorithm. For geometric transformation forgery, the proposed method can accurately locate tampered regions, especially in smooth areas, where the detection performance still shows good results. In addition, it also performs well in small-scale rotation and compression tampering scenarios.

In addition to using intuitive effect diagrams to demonstrate the effectiveness of the proposed method, we also conducted experiments at the image level. All images from the Ardizzone and CoMoFoD datasets were tested to calculate the number of tampered images correctly identified as tampered, the number of tampered images incorrectly identified as real, and the number of real images incorrectly identified as tampered. In comparison with other methods in the literature, precision, recall, and F_1 were calculated, and the experimental results are listed in Tables 2 and 3. The standard deviations of the three evaluation metrics were also calculated, and the data are presented in Table 4.

Table 2. Experimental results of precision (P), recall (R), and F_1 evaluation metrics for images in the Ardizzone dataset.

Method	P (%)	R (%)	F_1 (%)
Badr et al. [17]	89.52	91.23	90.37
Aydin et al. [38]	90.23	90.56	90.39
Proposed method	92.75	92.43	92.89

Table 3. Experimental results of precision (P), recall (R), and F_1 evaluation metrics for images in the CoMoFoD dataset.

Method	P (%)	R (%)	F_1 (%)
Badr et al. [17]	88.56	90.54	89.58
Aydin et al. [38]	89.46	91.56	90.50
Proposed method	95.23	93.78	95.12

Table 4. Standard deviation of three evaluation indicators on the two datasets.

Image Dataset	Precision	Recall	F_1
Ardizzone	±0.14	±0.20	±0.21
CoMoFoD	±0.15	±0.24	±0.28

We comprehensively analyzed the performance of the proposed algorithm through three evaluation metrics and compared it with the algorithms proposed in [17,38]. Table 2 presents the experimental data obtained by testing the proposed method and the comparison algorithms on the Ardizzone dataset [37], where it can be clearly seen that the method has certain advantages. Compared with the other algorithms, the precision of our proposed algorithm is higher. Although the recall rate is slightly lower than that of the algorithm in [17], the F_1 value of the proposed algorithm indicates a significant advantage. This is mainly because in the feature extraction stage, we used two detection methods, and the A-KAZE detector extracted more features in smooth areas, resulting in better detection performance in the tampering localization stage.

Table 3 shows the results of experimental testing on the CoMoFoD dataset. The proposed algorithm has significant advantages in precision, which is higher than that of the two comparison algorithms. The proposed method also has significant advantages in recall and overall F_1 value. The main reason for this is that the method proposed in [17] only uses SURF for feature point extraction, determines suspicious areas by replacing matching feature points with corresponding superpixel blocks, and then merges adjacent blocks based on similar local color features. This method is less robust to rotation and scale invariance and does not perform well in geometric transformation forgery detection. In this paper, we combined SURF and A-KAZE in the feature extraction stage to obtain enough feature points in smooth areas, and removed mismatches through a density-based clustering method, thereby effectively and accurately locating tampered areas.

To further test the superiority of the proposed method in terms of time complexity, we conducted tests on images in the Ardizzone and CoMoFoD datasets, and recorded the time required for the proposed method and the comparison algorithms to detect a single image. Table 5 presents the data. The proposed method required an average of 2.54 s for feature extraction on the Ardizzone dataset, and 61.58 s on average for feature matching and detection of mismatched pairs. On the CoMoFoD dataset, the average time for feature extraction was 2.24 s, and the time required for other detections was 51.99 s. The method proposed in [38] required a longer time for feature extraction due to the greater number of iterations and redundant feature points extracted, resulting in longer detection time. The average detection time was 122.12 s on the Ardizzone dataset and 99.25 s on the CoMoFoD dataset. The method proposed in [17] segmented the image equally and

without overlapping, and used a segmentation algorithm to divide the entire image into superpixel blocks, which undoubtedly increased the computational complexity of the algorithm. Consequently, the average time required for this method was 184.23 s on the Ardizzone dataset and 145.23 s on the CoMoFoD dataset. In summary, as a result of combining SURF and A-KAZE, which has the advantage of two detectors complementing one another, the proposed method outperformed the two comparison algorithms in terms of time complexity.

Table 5. Time complexity comparison.

Image Dataset	Badr et al. [17]	Aydin et al. [38]	Proposed Method
Ardizzone	184.23 s	122.12 s	65.12 s
CoMoFoD	145.23 s	99.25 s	54.23 s

5.4.2. Post-Processing Forgery Detection

This paper focuses primarily on post-processing operations for detecting tampering in images using two methods: noise and Gaussian blur. The detection results were compared and analyzed by comparing the visual detection effect with the binary image given in the dataset.

Gaussian blur forgery detection: To evaluate the detection performance of the proposed method on Gaussian blur forgery, we selected two blur factors, $\sigma = 0.5$ and $\sigma = 2$. The detection results are shown in Figure 9. The first row in the figure shows the original images, the second row shows the binarized truth images provided in the dataset, and the third row shows the detection results of the proposed method. When the Gaussian blur factor is set to $\sigma = 2$, it can be observed from the comparison between the detection results of the proposed method (Figure 9, third row) and the binary images from the dataset (Figure 9, second row) that there was no missed detection with the proposed method, and it accurately located the tampered regions in the dataset. This indicates that the proposed method has good detection performance on Gaussian blur forgery.

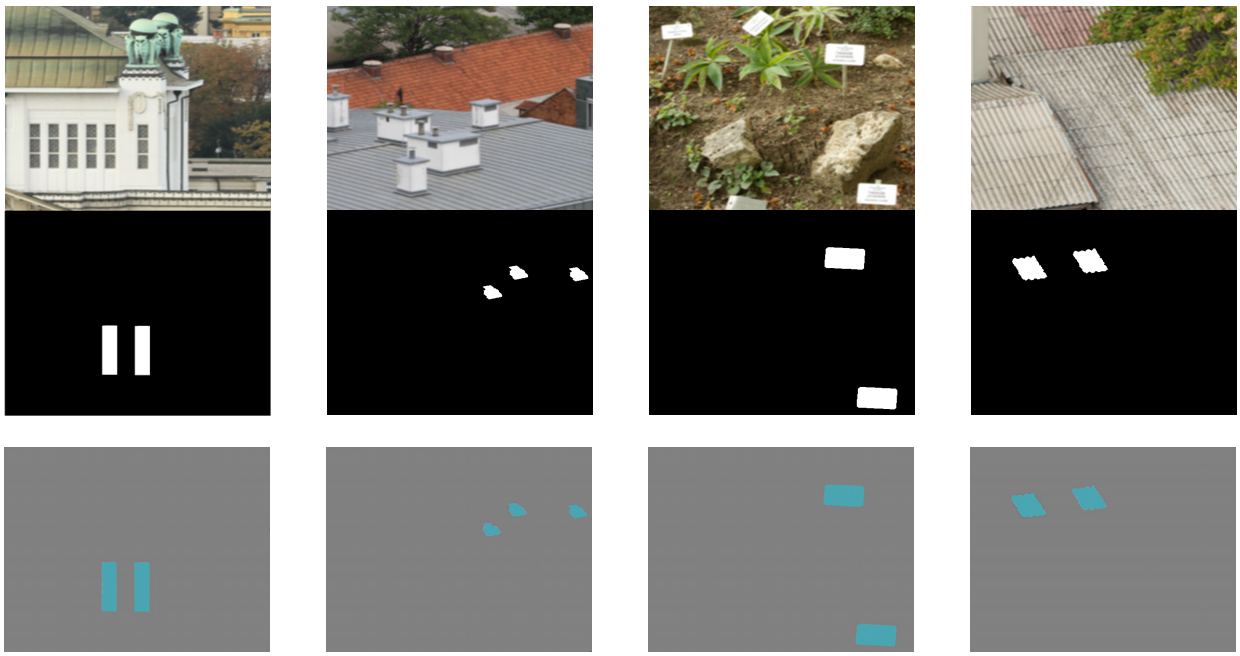


Figure 9. Rendering of Gaussian blur forgery detection. **First row**: original images; **second row**: binarized truth images; **third row**: effect images of forgery detection.

Gaussian white noise forgery detection: To verify the detection performance of the proposed method on Gaussian white noise tampering, we also tested the images in the dataset by adding Gaussian white noise with mean $m = 0$ and variance $v = 0.001$ and $v = 0.0005$. The detection results with Gaussian white noise forgery detection are shown in Figure 10. The first row shows the original images, the second row shows the binary images provided in the dataset, and the third row shows the experimental test results obtained in this study. It can be observed from the comparison between the experimental detection results (Figure 10, third row) and the binary images (Figure 10, second row) that the proposed method had no missed detection and accurately located the tampered regions with Gaussian white noise, demonstrating its effectiveness in detecting Gaussian white noise forgery.

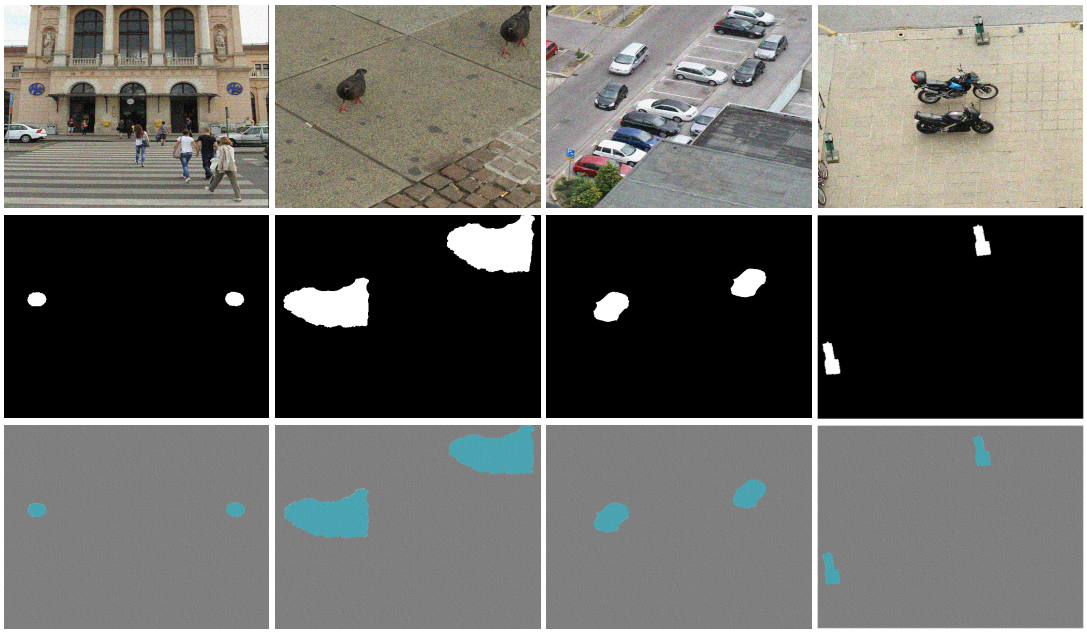


Figure 10. Effect images of Gaussian white noise forgery detection. **First row**: original images; **second row**: binarized truth images; **third row**: effect images of forgery detection.

The comparison between binarized truth images and detection results in Figures 9 and 10 shows that there are almost no missed detection regions with the proposed method. Whether in noise forgery or Gaussian blur forgery detection, the detection results of this paper are almost identical to the binary images provided in the dataset. This indicates that the proposed method demonstrates excellent performance in post-processing operations, such as Gaussian blur forgery and Gaussian white noise. Overall, the proposed method can accurately locate tampered regions, especially when the selected blur factor and noise variance are small, and the detection results are more accurate.

The results of the F_1 score for the proposed method and the two comparison algorithms are shown in Figure 11. From the figure, it can be observed that the proposed method has a significant advantage in detecting Gaussian noise and blur forgery.

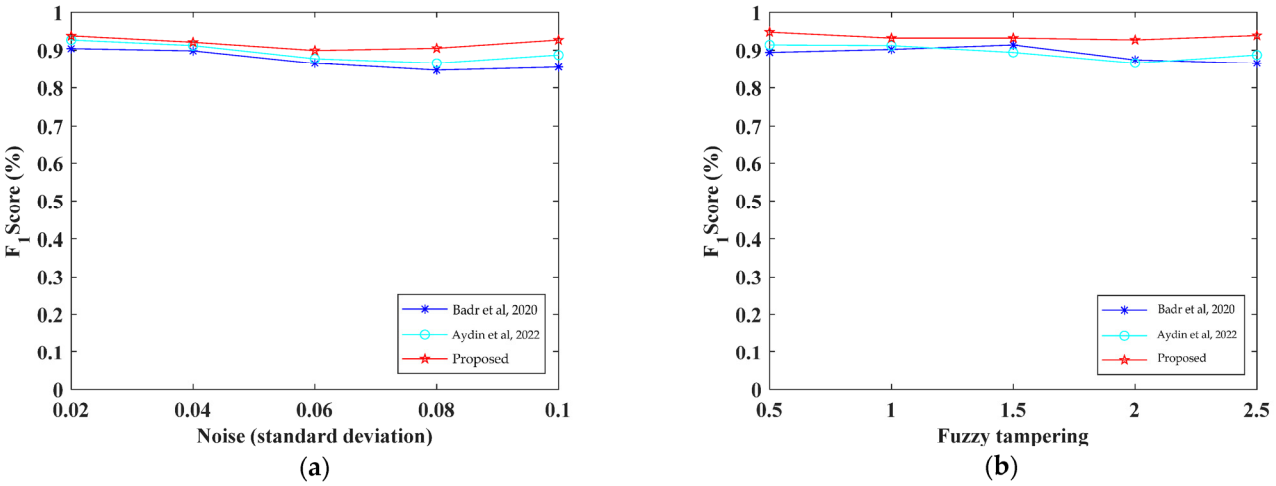


Figure 11. Score for different approaches against (**a**) noise addition attack and (**b**) fuzzy addition attack. Badr et al., 2020 [17]. Aydin et al., 2022 [38].

6. Conclusions

In this paper, we combined SURF and A-KAZE detectors and used g2NN for feature matching. To improve the localization accuracy, the DBSCAN algorithm was applied to remove erroneous matches and reduce false positives. The experimental results show that the proposed method has good detection performance. SURF and A-KAZE can extract effective feature points in smooth areas, and the DBSCAN algorithm can remove incorrect matchings, thus reducing false alarms. The proposed method demonstrates strong robustness against various tampering types such as rotation and scaling, and achieves improved detection accuracy in smooth areas. When small blur factors and noise variances are used in tampering operations, this method exhibits good detection performance.

Author Contributions: Conceptualization, G.F.; data curation, G.F.; investigation, Y.W.; methodology, G.F.; software, G.F.; validation, Y.W.; visualization, Y.Z.; writing—original draft, G.F.; writing—review and editing, Y.Z. All authors have read and agreed to the published version of the manuscript.

Funding: Industry-University-Research Innovation Fund of the Chinese Ministry of Education, Item Number: 2021ZYB01003; Shanghai Natural Science Foundation Project, Item Number: 17ZR1411900; Shanghai Science and Technology Commission Key Project, Item Number: 18511101600.

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

References

1. Bensaad, A.; Loukhaoukha, K.; Sadoudi, S. Keypoint-based copy-move forgery detection in digital images: A survey. In Proceedings of the 2022 7th International Conference on Image and Signal Processing and Their Applications (ISPA), Mostaganem, Algeria, 8–9 May 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 1–6.
2. Paul, S.; Pal, A.K. A fast copy-move image forgery detection approach on a reduced search space. *Multimed. Tools Appl.* **2023**, *82*, 25917–25944. [CrossRef]
3. Anushree, R.; Vinay Kumar, S.B.; Sachin, B.M. A Survey on Copy Move Forgery Detection (CMFD) Technique. In Proceedings of the 2023 International Conference on Intelligent and Innovative Technologies in Computing, Electrical and Electronics (IITCEE), Bengaluru, India, 27–28 January 2023; IEEE: Piscataway, NJ, USA, 2023; pp. 439–443.
4. Venugopalan, A.K.; Gopakumar, G. Copy-Move Forgery Detection-A Study and the Survey. In Proceedings of the 2022 Third International Conference on Intelligent Computing Instrumentation and Control Technologies (ICICICT), Kannur, India, 11–12 August 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 1327–1334.
5. Zhang, Z.; Wang, C.; Zhou, X. A survey on passive image copy-move forgery detection. *J. Inf. Process. Syst.* **2018**, *14*, 6–31.
6. Fridrich, J. Detection of copy-move forgery in digital images. In Proceedings of the Digital Forensic Research Workshop, Cleveland, OH, USA, 6–8 August 2003; pp. 19–23.
7. Kumar, A.; Singh, K.U.; Swarup, C.; Singh, T.; Raja, L.; Kumar, A. Detection of Copy-Move Forgery Using Euclidean Distance and Texture Features. *Traitement Signal* **2022**, *39*, 781–788. [CrossRef]

8. Fattah, S.A.; Ullah, M.M.I.; Ahmed, M.; Ahmmed, I.; Shahnaz, C. A scheme for copy-move forgery detection in digital images based on 2D-DWT. In Proceedings of the 2014 IEEE 57th International Midwest Symposium on Circuits and Systems (MWSCAS), Station, TX, USA, 3–6 August 2014; IEEE: Piscataway, NJ, USA, 2014; pp. 801–804.
9. Sabeena, M.; Abraham, L.; Varghese, A. Digital Image Forgery Detection Using Local Binary Pattern (LBP) and Harlick Transform with classification. In Proceedings of the 2021 IEEE International Power and Renewable Energy Conference (IPRECON), Kollam, India, 24–26 September 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1–6.
10. Rahma, F.I.; Utami, E.; Al-Fatta, H. Gaussian Pyramid Decomposition in Copy-Move Image Forgery Detection with SIFT and Zernike Moment Algorithms. *Telematika* **2022**, *15*, 1–13. [CrossRef]
11. Ye, W.; Zeng, Q.; Peng, Y.; Liu, Y.; Chang, C.C. A two-stage detection method of copy-move forgery based on parallel feature fusion. *EURASIP J. Wirel. Commun. Netw.* **2022**, *2022*, 30. [CrossRef]
12. Lee, J.C.; Chang, C.P.; Chen, W.K. Detection of copy–move image forgery using histogram of orientated gradients. *Inf. Sci.* **2015**, *321*, 250–262. [CrossRef]
13. Qin, C.; Chen, X.; Ye, D.; Wang, J.; Sun, X. A novel image hashing scheme with perceptual robustness using block truncation coding. *Inf. Sci.* **2016**, *361*, 84–99. [CrossRef]
14. Wang, C.; Huang, Z.; Qi, S.; Yu, Y.; Shen, G.; Zhang, Y. Shrinking the Semantic Gap: Spatial Pooling of Local Moment Invariants for Copy-Move Forgery Detection. *IEEE Trans. Inf. Forensics Secur.* **2023**, *18*, 1064–1079. [CrossRef]
15. Kumar, S.; Mukherjee, S.; Pal, A.K. An improved reduced feature-based copy-move forgery detection technique. *Multimed. Tools Appl.* **2023**, *82*, 1431–1456. [CrossRef]
16. Salman, M.; Uhl, A. Countering Anti-forensics of SIFT-based Copy-Move Detection. In Proceedings of the 2020 25th International Conference on Pattern Recognition (ICPR), Milan, Italy, 10–15 January 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 2701–2707.
17. Badr, A.; Youssif, A.; Wafi, M. A robust copy-move forgery detection in digital image forensics using SURF. In Proceedings of the 2020 8th International Symposium on Digital Forensics and Security (ISDFS), Beirut, Lebanon, 1–2 June 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–6.
18. Kumar, N.; Meenpal, T. Salient keypoint-based copy–move image forgery detection. *Aust. J. Forensic Sci.* **2023**, *55*, 331–354. [CrossRef]
19. Samel, M.; Reddy, A.M. An Empirical Study on Copy-Move Forgery Detection Techniques in Images. *Math. Stat. Eng. Appl.* **2022**, *71*, 183–193.
20. Benhamza, H.; Djeffal, A.; Cheddad, A. Image forgery detection review. In Proceedings of the 2021 International Conference on Information Systems and Advanced Technologies (ICISAT), Tebessa, Algeria, 27–28 December 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1–7.
21. Pan, X.; Lyu, S. Region duplication detection using image feature matching. *IEEE Trans. Inf. Forensics Secur.* **2010**, *5*, 857–867. [CrossRef]
22. Lowe, D.G. Distinctive image features from scale-invariant key points. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
23. Zheng, J.; Zhang, K. Copy-Move Forgery Detection Algorithm based on Feature Point Clustering. In Proceedings of the 2022 IEEE 6th Information Technology and Mechatronics Engineering Conference (ITOEC), Chongqing, China, 4–6 March 2022; IEEE: Piscataway, NJ, USA, 2022; Volume 6, pp. 775–780.
24. Fischler, M.A.; Bolles, R.C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* **1981**, *24*, 381–395. [CrossRef]
25. Liu, Y.; Wang, H.; Chen, Y.; Wu, H.; Wang, H. A passive forensic scheme for copy-move forgery based on superpixel segmentation and K-means clustering. *Multimed. Tools Appl.* **2020**, *79*, 477–500. [CrossRef]
26. Wang, X.Y.; Wang, C.; Wang, L.; Jiao, L.X.; Yang, H.Y.; Niu, P.P. A fast and high accurate image copy-move forgery detection approach. *Multidimens. Syst. Signal Process.* **2020**, *31*, 857–883. [CrossRef]
27. Wang, X.; Chen, W.; Niu, P.; Yang, H. Image copy-move forgery detection based on dynamic threshold with dense points. *J. Vis. Commun. Image Represent.* **2022**, *89*, 103658. [CrossRef]
28. Fatima, B.; Ghafoor, A.; Ali, S.S.; Riaz, M.M. FAST, BRIEF and SIFT based image copy-move forgery detection technique. *Multimed. Tools Appl.* **2022**, *81*, 43805–43819. [CrossRef]
29. Orhei, C.; Radu, L.; Mocofan, M.; Vert, S.; Vasiu, R. Urban landmark detection using A-KAZE features and vector of aggregated local descriptors. In Proceedings of the 2022 International Symposium on Electronics and Telecommunications (ISETC), Timisoara, Romania, 10–11 November 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 1–4.
30. Ouyang, T.; Shen, X. Online structural clustering based on DBSCAN extension with granular descriptors. *Inf. Sci.* **2022**, *607*, 688–704. [CrossRef]
31. Sujin, J.S.; Sophia, S. Copy-Move Geometric Tampering Estimation Through Enhanced SIFT Detector Method. *Comput. Syst. Sci. Eng.* **2023**, *44*, 157–171. [CrossRef]
32. Bay, H.; Ess, A.; Tuytelaars, T.; Van Gool, L. Speeded-up robust features (SURF). *Comput. Vis. Image Underst.* **2008**, *110*, 346–359. [CrossRef]
33. Perona, P.; Malik, J. Scale-space and edge detection using anisotropic diffusion. *IEEE Trans. Pattern Anal. Mach. Intell.* **1990**, *12*, 629–639. [CrossRef]
34. Alcantarilla, P.F.; Solutions, T. Fast explicit diffusion for accelerated features in nonlinear scale spaces. *IEEE Trans. Patt. Anal. Mach. Intell.* **2011**, *34*, 1281–1298.

35. Amerini, I.; Ballan, L.; Caldelli, R.; Del Bimbo, A.; Serra, G. A sift-based forensic method for copy–move attack detection and transformation recovery. *IEEE Trans. Inf. Forensics Secur.* **2011**, *6*, 1099–1110. [CrossRef]
36. Borah, B.; Bhattacharyya, D.K. An improved sampling-based DBSCAN for large spatial databases. In Proceedings of the International Conference on Intelligent Sensing and Information Processing, Chennai, India, 4–7 January 2004; IEEE: Piscataway, NJ, USA, 2004; pp. 92–96.
37. Ardizzone, E.; Bruno, A.; Mazzola, G. Copy–move forgery detection by matching triangles of key points. *IEEE Trans. Inf. Forensics Secur.* **2015**, *10*, 2084–2094. [CrossRef]
38. Aydin, Y.I. Comparison of color features on copy-move forgery detection problem using HSV color space. *Aust. J. Forensic Sci.* **2022**, 1–17. [CrossRef]