frontiers | Frontiers in Signal Processing

ORIGINAL RESEARCH
published: 24 June 2022
doi: 10.3389/frsip.2022.906304

A Very Fast Copy-Move Forgery Detection Method for 4K Ultra HD Images

Laura Bertojo, Christophe Néraud and William Puech *

Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier, Centre National de la Recherche Scientifique, University of Montpellier, Montpellier, France

Copy-move forgery detection is a challenging task in digital image forensics. Keypoint-based detection methods have proven to be very efficient to detect copied-moved forged areas in images. Although these methods are effective, the keypoint matching phase has a high complexity, which takes a long time to detect forgeries, especially for very large images such as 4K Ultra HD images. In this paper, we propose a new keypoint-based method with a new fast feature matching algorithm, based on the generalized two nearest-neighbor (g2NN) algorithm allowing us to greatly reduce the complexity and thus the computation time. First, we extract keypoints from the input image. After ordering them, we perform a match search restricted to a window around the current keypoint. To detect the keypoints, we propose not to use a threshold, which allows low intensity keypoint matching and a very efficient detection of copy-move forgery, even in very uniform or weakly textured areas. Then, we apply a new matching algorithm, and finally we compute the cluster thanks to the DBSCAN algorithm. Our experimental results show that the method we propose can detect copied-moved areas in forged images very accurately and with a very short computation time which allows for the fast detection of forgeries on 4K images.

Keywords: digital image forensics, fast copy move forgery detection, SURF, keypoints, DBSCAN

OPEN ACCESS

Edited by:
Cecilia Pasquini,
University of Trento, Italy

Reviewed by:
Roberto Caldelli,
Consorzio Nazionale Interuniversitario Per Le Telecomunicazioni, Italy
Guzin Ulutas,
Karadeniz Technical University, Turkey

***Correspondence:**
William Puech
william.puech@lirmm.fr

Specialty section:
This article was submitted to Image Processing, a section of the journal Frontiers in Signal Processing

Received: *28 March 2022*
Accepted: *02 May 2022*
Published: *24 June 2022*

Citation:
Bertojo L, Néraud C and Puech W (2022) A Very Fast Copy-Move Forgery Detection Method for 4K Ultra HD Images. Front. Sig. Proc. 2:906304. doi: 10.3389/frsip.2022.906304

1 INTRODUCTION

In recent decades, the democratisation of the Internet and the apparition of social networks have given people the opportunity to easily access and produce multimedia content. Images and videos are the most exchanged data and are usually perceived as undeniable proof. In addition, image and video editing softwares have become more powerful than ever and are now accessible to everyone. In the past decades, it has never been easier to remove or insert elements in images or videos (Chen et al., 2016; D'Amiano et al., 2019; Aloraini et al., 2021), to change the hue (Hou and Lee, 2017), to make images more aesthetic or sometimes to modify the meaning of multimedia contents and therefore spread false information. In this context, it is difficult to establish their integrity and authenticity when published online and in particular on social networks. If most of the manipulations carried out remain harmless, some can be carried out for malicious purposes. This has become a problem in various fields such as politics, journalism, military security and even medical imaging.

During the last few years, several active and passive approaches have been proposed to tackle the image authentication problem. The active approaches consist in embedding a digital watermark or a signature in the image in advance (Cox et al., 2002). Although these methods can be effective, they alter the image quality and they do not match real-life applications. The second class of approaches is

passive and allows us to detect forgeries in an image without any prior knowledge of the original image. There exists three main types of forgeries. First copying-and-moving an area of an image to duplicate it or hide it in another area of the same image, second impainting an area of an image in order to remove an element within it, and finally copying-and-pasting of an area of an image into another. There are multiple approaches to detect forgeries. For example it is possible to analyze the incoherence of light inside the scene (De Carvalho et al., 2013; Pomari et al., 2018), to look for double-JPEG compression areas(Li et al., 2008; Amerini et al., 2014) or to analyze some features inside the image in order to distinguish tampered areas from genuine ones (Cozzolino et al., 2015b). Another method has been developed for tamper detection in JPEG compressed image scenarios (Lin et al., 2011). JPEG is the most commonly used format for storing or sharing an image. It works by approximating the quantization table to detect altered regions. Another way to detect spliced areas is to search for noise inconsistencies on the border of the spliced area like in (Rao and Ni, 2016). We can distinguish two classes of copy-move forgery detection algorithms, the first class is based on block division while the second class is based on keypoint extraction. Some recent methods propose hybrid approaches which take advantages of both classes. Although keypoint-based approaches are faster than the block-based ones, the matching phase remains time-consuming.

In this paper, we propose a very fast copy-move forgery detection (VFCMFD) method with a reduced computation time compared to current state of the art methods while keeping a similar efficiency. Our proposed VFCMFD method allows us to analyze 4K Ultra HD images. Our method starts by extracting the keypoints using the SURF detector before applying our new VFCMFD method. Indeed, the proposed fast feature matching algorithm is used to match keypoints found according to their gradient's orientation. For each keypoint, the search for matches is performed only on those having an orientation close to the current keypoint. The results are then filtered with the DBSCAN algorithm in order to obtain a tampered area which can be extended to find convex hulls giving us a final binary mask locating the tampered areas. Unlike state-of-the-art approaches that are based on keypoint matching, the method we propose does not use any threshold for the detection of the keypoints. This allows low intensity keypoint matching and a very efficient detection of copy-move forgery, even in very uniform or weakly textured areas. Consequently, the number of keypoints extracted from the image increases significantly and due to the $O(n^2)$ complexity, the computation time also increases strongly. For that reason, we propose in our method to order all the keypoints and to perform a match search restricted to a window around the current keypoint. With this technique, we reduce the complexity from $\mathcal{O}(n^2)$ to $O(kn)$ with k being the size of the search window. Our approach allows us to drastically reduce the computation time while remaining very efficient with the matching of keypoints and thus the copy-move forgery detection in both uniform and weakly textured areas.

The main contribution and highlights of our proposed VFCMFD method can be summarized as follows:

- The proposed method allows us to detect copied-moved forged areas for 4K Ultra HD images.
- The method significantly reduces the computation time of keypoint-based techniques by using a new and fast feature matching algorithm.
- Even if our method strongly reduces the computation time, the results obtained are comparable to current state of the art methods in the case of plain copy-move.

The rest of the paper is organized as follows. First, in **Section 2**, we present previous work on image forgery detection, in particular for copy-move forgery detection. Then, in **Section 3**, our proposed Very Fast Copy-Move Forgery Detection (VFCMFD) method is presented with details. In **Section 4**, we present several experimental results applied to very large images and we compare our method to the current state of the art approaches. Finally, in **Section 5**, we conclude and present future work.

2 PREVIOUS WORK

Copy-move forgery is one of the most classic attacks to tamper with an image. This attack consists of copying one or more areas in an image and move them elsewhere in the image. Detecting them can be really challenging, especially when the forged areas have been processed by several attacks like noise addition, JPEG compression, blurring, scaling or rotation or if it involves a smooth or small area. Since 2012, there exists common databases to evaluate the copy-move forgery detectors, such as the database FAU proposed by Christlein et al. (2012) which contains 48 forged images with realistic copy-move manipulation and where the average size is 3000×2300 pixels, and the database GRIP proposed by Cozzolino et al. (2015a) created in 2015 which contains 80 images of size 768×1024 pixels, where some very smooth areas are tampered. Copy-move forgery detectors can be classified into two main categories, these are block-based and keypoint-based.

In **Section 2.1** we detail previous block-based methods while in **Section 2.2** we present previous keypoint-based methods which have been published before 2015. Finally, in **Section 2.3** we present the most recent copy-move forgery detectors that we have used to compare with our proposed VFCMFD method using the GRIP and the FAU databases.

2.1 Block-Based Methods

Block-based methods follow a common pipeline. Firstly, the detector divides the input image into overlapping circular or square blocks. Then for each block, a feature vector is computed in order to characterize a robust and low dimensional representation of the local image characteristics. Then, similar blocks are matched following the obtained descriptors. Most of the methods use an euclidean distance between descriptors to estimate the similarity between two blocks. Usually, there remains some matches, such as those not involved in a tampered area. Therefore, there is a filtering matching process to remove them. An optional post processing step of the detected areas may also be

performed. The main purpose of block-based techniques is to find the best descriptor for each block, which has to be low dimensional and robust to various attacks, such as lossy compression, blur, noise addition, contrast change or affine transformation.

In 2003, Fridrich et al. (2003) proposed a frequency-based feature, the quantized DCT coefficients. Later, Hu et al. (2011) developed a method that introduces a truncating procedure to reduce the dimension of the feature vectors in order to help improve the detection performances based on DCT. The use of DCT allows us to make the assumption that the more significant features are captured in fewer coefficients. This method is time consuming and is sensitive to additive noise or lossy compression. To tackle this problem, Popescu and Farid (2004) suggested reducing the feature dimension by using principal component analysis (PCA) as a compact representation of the blocks. This approach is robust to minor variations due to additive noise or lossy compression and the computation cost is significantly reduced, but the accuracy rate drops for small sized blocks or when the signal to noise ratio is lower than 24 *dB*. Luo et al. (2006) computed seven characteristic features based on the statistical analysis of the pixels of each block based on the average intensity of the pixels on the block in each channel (RGB) and the block intensity ratio in four directions. This method is more robust against JPEG lossy compression, blur and additive noise. To deal with the blur attack, Mahdian and Saic (2007) implemented a method based on blur moment invariants which is robust to blur degradation, additive noise, contrast change and lossy compression. It has a high detection rate but has a poor computation time. Another approach is given by Lynch et al. (2013) with an efficient expanding block algorithm. Direct block comparison is employed instead of indirect comparisons based on block features. Kang and Wei (2008) used singular value decomposition (SVD) to produce algebraic and geometric invariant vectors. While in previous algorithms blocks are directly extracted from the original image, Li et al. (2007) applied the DWT on the input image and SVD feature extraction is used on the low-frequency component wavelet subband to decrease the computation cost. The DWT reduces the image size by four and the SVD reduces the dimension of the descriptor. The computation cost is then drastically reduced. In most cases, the methods presented previously can detect a copy-move forgery, but they fail to detect forgery when the copied region is rotated or scaled. In order to tackle this issue, Myna et al. (2007) proposed applying a wavelet transform to the input image and then to map each block to log-polar coordinate. However, the phase correlation used as similarity criteria can only detect moderate radius rotation. Bayram et al. (2009) used Fourier-Mellin transform (FMT) as descriptors where each block data is first Fourier transformed, and the magnitude values are re-sampled into the log-polar coordinate. The log-polar values are then projected to 1D feature vectors. This method works only for small scaling and small rotations. Wang et al. (2009) tried to overcome the effect of rotation by using circular blocks instead of square ones and adopting the mean of pixel intensity on circles of different radii as features. Liu et al. (2011) improved this method by combining it with Hu moment. Later, Li et al. (2013) proposed a method which also uses circular overlapping blocks and extracts features thanks to Local Binary Patterns (LBP) which are able to detect forged areas precisely against geometric distortions like scaling, rotation, compression, additive noise and blurring, but failed to detect the forged areas with arbitrary rotated angles. Bravo-Solorio and Nandi (2009) represent blocks with the log-polar coordinates and compute a one dimensional feature vector which is the sum of the angle values to have a rotational invariance. This method works directly on pixel intensity and is therefore sensitive to a pixel change attack. Ryu et al. (2010) use a detector based on the Zernike moment whose magnitude is algebraically invariant against rotation. This method achieved an average detection precision rate of 83% in the case of region rotation.

2.2 Pioneering Keypoint-Based Methods

As presented in **Section 2.1**, block-based methods can be robust against various attacks such as noise addition, filtering and lossy compression, but they lack of robustness against rotation and scaling. For that purpose, the second type of copy-move forgery detection methods relies on keypoints which aim to directly match keypoints in the image instead of blocks. The detection of keypoints needs to be invariant to a large number of operations. Huang et al. (2008) pioneered this type of method in using Scale Invariant Feature Transform [SIFT (Lowe, 2004)] and the Best-Bin-First algorithm derived from the k-d tree to match the keypoints. This approach is really robust against geometric transformation attacks. Pan and Lyu (2010) improved this method by clustering the matches thanks to a tree and estimated the geometric transformation performed thanks to the RANSAC algorithm (Fischler and Bolles, 1981). Amerini et al. (2010) proposed a new way to match the keypoints by using the ratio between the distance of the closest neighbor to that of the second-closest one and used a cluster tree to filter the matches. This method allows the detection of multiple duplicated areas, where the matched correspondances may follow different geometric transformations. But in this case, using a global RANSAC algorithm does not work anymore. To deal with this problem, Amerini proposed to use a hierarchical clustering algorithm to group the matches into matching clusters and to apply the RANSAC algorithm into matched clusters. Shivakumar and Baboo (2011) proposed a method based on the Speed Up Robust Feature [SURF (Bay et al., 2006)] which is a descriptor inspired by SIFT but faster. Indeed, its descriptor is twice less large and more robust against some image transformation than SIFT descriptor. Tahaoglu et al. (2021) presented a method using keypoints and an estimation of the geometric transformations using the RANSAC algorithm to determine whether an image is authentic. If the image is tampered, the estimated homography is used to locate the altered regions. Kakar and Sudha (2012) used a method based on the MPEG-7 image signature tools they have modified to manage copy-move forgery in an image. Another approach proposed by Ardizzone et al. (2015) is based on keypoints in order to generate a Delaunay triangulation and to compare triangles instead of single points.

2.3 Recent Copy-Move Forgery Detection Methods

In this section we present the details of several recent copy-move forgery detectors that we have used to compare with our proposed VFCMFD method. The main drawback of the methods presented in **Section 2.2** is that they have difficulty in detecting forged areas when they are located in smooth areas. Furthermore, copied-moved areas are sparsely covered by matched keypoints. The areas with few keypoints could be then missed and taken for noise during the filtering process phase. To tackle this issue, Zandi et al. (2016) proposed a method with a novel interest point detector specialized for the copy-move forgery problem where even smooth areas are covered by keypoints. The interest point density is adjusted over the image and concentrated on the suspected areas by an iterative improvement strategy. Recently, Li and Zhou (2019) proposed a method which lowers the contrast threshold in the SIFT algorithm and resizes the image in order to generate a sufficient number of keypoints, even in smooth areas. They use hierarchical keypoint matching to reduce the computation costs due to the increase of keypoints, but also to obtain appropriate matching. Recent approaches such as Ardizzone et al. (2015) used a hybrid approach between block-based and keypoint-based methods that take advantage of both. After an over-segmentation of the host image, most of these approaches can be divided into two steps. The first step consists of approximately finding the blocks involved in the forgery and the second step to refine the detection in order to increase the precision rate. Li et al. (2015) proposed to segment the image into semantically independent patches and to estimate an affine transform matrix which is then refined thanks to an Expectation-Maximization-based algorithm. Pun et al. (2015) suggested an over-segmentation technique which segments adaptively to the host image thanks to the SLIC algorithm. The method is based on superpixels to refine the result in the second stage. Mei et al. (2019) also proposed a method using SLIC to segment the image before estimating the affine transformation performed between the suspect superpixels followed by a two-stage local search algorithm in order to generate the final binary mask. Most of these methods eliminate superpixels with only one keypoint. In order not to miss any forgery and to take into account these blocks, Gupta and Singh (2021) proposed a method based on FLANN matching. Wang et al. (2019) apply a superpixel segmentation and classify each block into two categories: smooth and textured areas. They use a SURF detector and the PCET coefficients as descriptors. Thus, they find rectangle areas with high density matched keypoints and use circular overlapping blocks to these areas to refine the results obtained. Muzaffer et al. (2020), and Meena and Tyagi (2020) also segmented the image into smooth and textured areas. In (Muzaffer et al., 2020) the authors used the quadtree decomposition: if a block does not exceed a defined size, then it is judged as complex and therefore textured, otherwise it is considered as smooth. A textured image is generated from the smooth part using the rotation invariant version of the LBP operator (LBPROT) in order to extract SIFT keypoints. In (Meena and Tyagi, 2020), authors took advantage that block-based methods are more efficient than keypoints based ones on smooth regions. The image is segmented using a standard deviation technique, SIFT points are extracted and matched in the textured regions while overlapping blocks are used in the smooth regions. In a recent study, Lyu et al. (2021) segment the input image into triangles using the Delaunay triangulation over the keypoints that are matched thanks to the generalized two nearest-neighbor (g2NN) algorithm. In the second stage, the obtained triangles are extended by adding their adjacent triangles iteratively and their vertices form a new subset of keypoints that are matched thanks to the g2NN algorithm. New methods focus on reducing time complexity while maintaining high performance. Diwan et al. (2019) proposed a block-based method using Local Linear Projection (LLP) which has a similarity preserving property. Similar blocks are projected close to each other in the LLP space which allows not to lexicographically sort all the blocks in the image. Chen et al. (2020) proposed to improve the time complexity of the keypoints based methods by clustering the extracted SIFT points according to their scale and colour.

In our proposed approach, we suggest a new method that significantly reduces the computation time of keypoint-based techniques by using the SURF detector algorithm to be able to detect tampered areas in very large images such as 4K Ultra HD images.

3 PROPOSED VFCMFD METHOD

In this section, we present with details our proposed Very Fast Copy-Move Forgery Detection (VFCMFD) method. First, in **Section 3.1** we detail the overview of the VFCMFD method. Then, the extraction of the keypoints is developed in **Section 3.2**. The fast matching algorithm is presented in **Section 3.3** and in **Section 3.4**, we explain the clustering process we have developed. Finally, **Section 3.5** presents the binary mask generation.

3.1 Overview of the Method

As illustrated in **Figure 1**, the proposed approach uses the SURF detector (Bay et al., 2006) in order to extract keypoints and features from the original image. An efficient and very fast matching algorithm is then applied on the extracted keypoints. For each extracted keypoint, we must calculate if at least one other extracted keypoint is similar enough to it. Due to the very high number of keypoints, the quadratic matching algorithm, as described in (Wang et al., 2019), requires too much computation time. For this reason, we have developed a very fast matching algorithm, whose computation time is vastly reduced, compared to the quadratic algorithm. In addition, the efficiency of our algorithm is very similar to the original one. In particular, our proposed very fast matching algorithm gives very good results when a tampered area has been copied and moved by translation, without rotation or scaling.

In the next step, as presented in **Figure 1**, for each pair of matched keypoints we consider the line segment between the two matched keypoints. The idea is that if an area has been copied-moved in the image, then there should be a large number of matches between the copied area and the moved one. In this case, we should have quite a

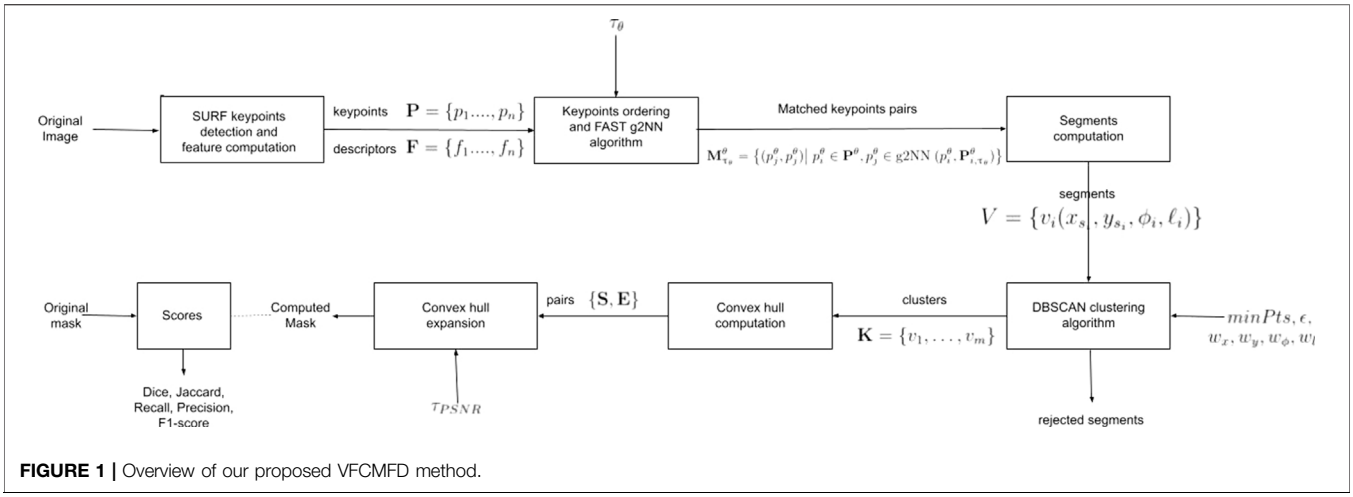

FIGURE 1 | Overview of our proposed VFCMFD method.

dense pack of nearly parallel line segments that are approximately the same length. We then propose to use the DBSCAN clustering algorithm in a four-dimensional space in order to detect clusters the dense areas corresponding to the copied-moved areas (Ester et al., 1996). The final step of the VFCMFD method, as illustrated in **Figure 1**, consists in computing the convex hull from the extracted keypoints belonging to the detected clusters in the previous step. These convex hulls give us a rough idea of the forged areas. In order to get more accurate results, we propose to expand the hulls in order to get the final binary mask. Note that when the original binary mask is given, we can compute accuracy scores such as F1-score (Christlein et al., 2012).

3.2 SURF Keypoint Extraction

Our proposed algorithm extracts keypoints using SURF detector (Bay et al., 2006). The SURF detector gives very similar results to the SIFT detector (Lowe, 2004), but is much more efficient in terms of computation time. Indeed, SURF algorithm gives accurate enough keypoints for our proposed application.

The first step is, considering an input image, to extract n SURF keypoints $\mathbf{P} = \{p_1, \ldots, p_n\}$ and compute their respective descriptors $\mathbf{F} = \{f_1, \ldots, f_n\}$. Each extracted keypoint p_i, with $i \in \{1, \ldots, n\}$, is characterized as well by an orientation noted θ_i. Note that the SURF keypoints are well adapted to our method since, as SIFT keypoints, they are robust to transformations such as rotations or scaling. Moreover, they are also robust to blurring, JPEG compression and noise addition which makes them really practical for forgery detection. The SURF detector is based on an approximation of a determinant of the Hessian blob detection method. Indeed, an Hessian matrix is computed for each pixel, which is kept only if the determinant of the Hessian matrix is larger than a threshold τ_{SURF}. As a function of the τ_{SURF} value, the method can miss some keypoint matches, in particular when the threshold value is too high. In order to not miss any keypoints, we take into account in our method all extracted keypoints with $\tau_{SURF} = 0$. Note that in this case, our algorithm extracts a very large number of keypoints. That is why in the next step of our algorithm, we have to find both a very fast and an efficient way to compute keypoint matching.

3.3 Fast Feature Matching Algorithm

Each extracted keypoint p_i, with $i \in \{1, \ldots, n\}$, is characterized by its feature vector f_i, called descriptor. A way to make matches between keypoints is to compare their respective descriptors. Intuitively, the best match for a keypoint $\{p_i, f_i\}$ is the keypoint whose descriptor f_j is the closest to f_i for the L^2 norm. However, as stated in (Amerini et al., 2011), since the feature space is high-dimensional (64 or 128 dimensions depending on whether we use the regular or extended SURF detector) this L^2 norm-based approach does not give good enough results.

As suggested by Amerini et al. (2011), it is more efficient to use a generalized two Nearest-Neighbor (g2NN) algorithm in order to find matches. Let $i \in \{1, \ldots, n\}$, we consider the vector $\mathbf{D}_i = \{d_1, \ldots, d_{n-1}\}$ the sorted Euclidean distances between a keypoint p_i and all the other keypoints. The regular 2NN test consists of checking whether:

$$\frac{d_1}{d_2} < \tau_{2NN}, \tag{1}$$

where $\tau_{2NN} \in (0, 1)$ is a threshold.

Amerini et al. (2011) states that while this test is efficient, its biggest flaw is that it does not let a keypoint have more than one match. This happens when an area of the image has been copied-moved more than once. In order to solve this problem, Amerini et al. (2011) improved the 2NN test by making a g2NN in the following way. Instead of only testing the ratio between the first and the second Euclidean distances, this ratio is calculated:

$$\frac{d_i}{d_{i+1}} < \tau_{g2NN}. \tag{2}$$

If we find a ratio that is greater than τ_{g2NN}, then we stop the process. We then get a set of Euclidean distances $\mathbf{D}'_i = \{d_1, \ldots, d_k\}$. The keypoints corresponding to the distances in $\mathbf{D}'_i$ are the matching ones for the current keypoint p_i. This procedure is repeated for all the keypoints p_i in order to obtain a set of matched keypoints for each keypoint.

Even if this algorithm gives really good results, it is really slow on large images such as 4K Ultra HD images because its

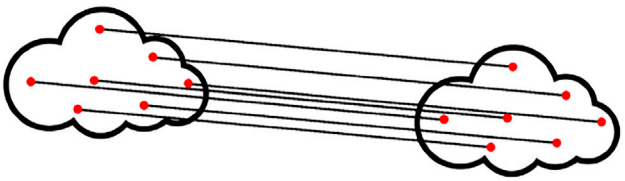

FIGURE 2 | Example of a tampered image where the red dots are the matched keypoints and the clouds are the original and the copied-moved areas.

complexity is $\mathcal{O}(n^2)$, where n is the number of extracted keypoints. This is why in this article we propose a solution to find similar matches with a faster algorithm. First, we sort our vector of keypoints by the value of their angles:

$$\mathbf{P}^\theta = \left\{p_1^\theta, \ldots, p_n^\theta\right\}. \tag{3}$$

For each keypoint p_i^θ, with $i \in \{1, \ldots, n\}$, of $\mathbf{P}^\theta$, we define the vector:

$$\mathbf{P}_{i,\tau_\theta}^\theta = \left\{p_j^\theta \in \mathbf{P}^\theta \mid |\theta_i - \theta_j| \leq \tau_\theta\right\}, \tag{4}$$

where τ_θ is a threshold on the difference of the orientation of the two keypoints.

With our proposed approach we do not have to loop over all the keypoints to compute $\mathbf{P}_{i,\tau_\theta}^\theta$. Indeed, as we already have the keypoints sorted by their angles, the vector $\mathbf{P}_{i,\tau_\theta}^\theta$ only consists in a small window around each keypoint p_i^θ in $\mathbf{P}^\theta$. We then define the function as:

$$\begin{array}{rccc} \text{g2NN:} & \mathbf{P}^\theta \times \mathcal{P}\left(\mathbf{P}^\theta\right) & \rightarrow & \mathcal{P}\left(\mathbf{P}^\theta\right) \\ & \left(p_i^\theta, X\right) & \longmapsto & Y, \end{array} \tag{5}$$

which applies the g2NN algorithm described in (Amerini et al., 2011) but only on a specific subset of keypoints.

The output of the algorithm consists of a set of keypoints which are matched with p_i^θ if at least one is found. We define then the vector of matches as:

$$\mathbf{M}_{\tau_\theta} = \left\{\left(p_i^\theta, p_j^\theta\right) \middle|\ p_i^\theta \in \mathbf{P}^\theta, p_j^\theta \in \text{g2NN}\left(p_i^\theta, \mathbf{P}_{i,\tau_\theta}^\theta\right)\right\}. \tag{6}$$

Algorithm 1 presents with details all the steps of the proposed approach. This algorithm is faster than the original g2NN algorithm because there are far fewer matches to check. First, we only select the keypoints whose angle is very close to the current keypoint p_i^θ. This step reduces a lot of the complexity of the algorithm. Finally, we apply the g2NN algorithm on a vector whose size is a lot smaller than in the original algorithm. In terms of complexity, our proposed fast feature matching is in $\mathcal{O}(kn)$, where n is the number of keypoints and k the size of the window taken around each keypoint. The smaller the value of τ_θ, the smaller the value of k.

Algorithm 1. Fast keypoint matching algorithm.

input : The ordered keypoints $\mathbf{P}^\theta$, a threshold τ_θ
output : A vector M such as $M[i]$ contains all keypoints matching p_i^θ
$M \leftarrow []$;
for $i \in \{1, \ldots, n\}$ **do**
  $\mathbf{P}_{i,\tau_\theta}^\theta \leftarrow \left\{p_j^\theta \in \mathbf{P}^\theta \middle|\ |\theta_i - \theta_j| \leq \tau_\theta\right\}$;
  $M[i] \leftarrow \text{g2NN}\left(p_i^\theta, \mathbf{P}_{i,\tau_\theta}^\theta\right)$;
return M

3.4 Clustering Algorithm

In this section we present with details the clustering algorithm we have developed. Let p_i^θ and p_j^θ be two matched keypoints $(p_i^\theta, p_j^\theta) \in \mathbf{M}_{\tau_\theta}$ such as $p_i^\theta \leq p_j^\theta$ for the lexicographical order. We can then construct a line segment v from p_i^θ, the start point noted $p_s(x_s, y_s)$, to p_j^θ, the end point noted $p_e(x_e, y_e)$. From an image where an area has been copied and moved to another area in the same image (copy-move forgery), as presented in **Section 3.2** we chose to use SURF keypoints because they are robust to such operations. In particular if a copied-moved area is only translated, we then obtain a set of nearly parallel line segments between the two areas, as illustrated in **Figure 2**.

The main objective of our proposed clustering algorithm is to create clusters from sets of line segments and to reject outlier line segments. Indeed, all the line segments of the same cluster must be approximately parallel, of the same length and close to each other. Since we want to cluster parallel line segments we need to discriminate the line segments by their orientation which is noted ϕ. We also want to have clusters composed of line segments that all start in approximately the same area by discriminating them by their start points (x_s, y_s). Note that if we consider an area which is copied and moved twice at two different places, the length ℓ of each line segment and the angle ϕ to the abscissa axis should also be used to discriminate the clusters. Each line segment v can be then characterized by four elements which are x_s, y_s, ϕ and ℓ.

It is important to note that after the matching step presented in **Section 3.3**, there may be a certain number of line segments that do not correspond to a copy-move forgery. Usually, these line segments are isolated. The usual representation of line segments of a tampered area consists of large sets of nearly parallel line segments between copied-moved areas, and some random line segments that do not represent anything. The filtering process is then performed by the DBSCAN algorithm (Ester et al., 1996) which is a density-based algorithm that relies on the estimated density of clusters to perform the partitioning without knowing the number of clusters. The DBSCAN algorithm includes dense areas of line segments in the same cluster, and classifies random line segments as noise. Considering a line segment v characterized by (x_s, y_s, ϕ, ℓ), the DBSCAN algorithm searches for line segments that are within a radius ε around v. If in this radius, there are more than *minPts* line segments, then v is considered to belong to this cluster.

The problem with this approach comes from the fact that we want to make a cluster out of nearly parallel and same-length line segments, but the line segments set can be very wide. For example, a large copy-move forgery may have a cluster that covers almost half the image size. The clustering algorithm must then be able to discriminate strongly between ϕ and ℓ and be more flexible on x_s and y_s. So, we propose to add weights on each component to obtain a weighted pseudo-distance between two line segments v_i, $(x_{s_i}, y_{s_i}, \phi_i, \ell_i)$ and v_j, $(x_{s_j}, y_{s_j}, \phi_j, \ell_j)$:

$$d(v_i, v_j) = w_x \frac{(x_{s_i} - x_{s_j})^2}{(H+L)/2} + w_y \frac{(y_{s_i} - y_{s_j})^2}{(H+L)/2} + w_\phi \frac{(\phi_i - \phi_j)^2}{\phi_i} + w_\ell \frac{(\ell_i - \ell_j)^2}{\ell_i}, \tag{7}$$

where H and L are the size of the image in order to obtain relative distances.

3.5 Binary Mask Generation

Once the clusters are detected, we have to generate a binary mask locating the tampered areas, noted $\mathcal{M}$. To generate this binary mask, for each cluster we consider all of the start and end points of the line segments. Let us consider a cluster $\mathbf{K} = \{v_1, \ldots, v_m\}$, where a line segment v_i, with $i \in \{1, \ldots, m\}$, can be characterized by its start point $p_{s_i} = (x_{s_i}, y_{s_i})$ and its end point $p_{e_i} = (x_{e_i}, y_{e_i})$. We define the two following sets as:

$$\mathbf{S} = \{p_{s_i} \mid i \in \{1, \ldots, m\}\}, \tag{8}$$

$$\mathbf{E} = \{p_{e_i} \mid i \in \{1, \ldots, m\}\}. \tag{9}$$

The set **S** contains all the start points of the line segments in **K** while the set **E** contains all of its end points. We compute the convex hull of the points in **S** and the convex hull of the points in **E**, noted respectively $\mathcal{H}_S$ and $\mathcal{H}_E$. These two convex hulls make it possible to locate the heart of the tampered areas since they are inside the real tampered areas. We can observe that at the beginning the two convex hulls $\mathcal{H}_S$ and $\mathcal{H}_E$ have the same shape. Since the two computed convex hulls are inside the real tampered areas and do not fully match them, we propose to expand them as presented in Algorithm 2. From the two convex hulls $\mathcal{H}_S$ and $\mathcal{H}_E$ and a threshold τ_{PSNR}, **Algorithm 2** describes how to obtain a binary mask $\mathcal{M}$.

Algorithm 2. Colour clustering algorithm.

input : A pair of convex hulls $\mathcal{H}_S$ and $\mathcal{H}_E$, a threshold τ_{PSNR}
output : A binary mask $\mathcal{M}$ containing the two sets of pixels corresponding to the copied and the moved areas

$finished \longleftarrow false$;
while ***not*** $finished$ **do**
  $B_S \longleftarrow \text{border}(\mathcal{H}_S)$;
  $B_E \longleftarrow \text{border}(\mathcal{H}_E)$;
  for $(p_{s_i}, p_{e_i}) \in B_S \times B_E$ **do**
    $finished \longleftarrow true$;
    $\mathcal{N}_{p_{s_i}} \longleftarrow \text{neighbourhood}(p_{s_i})$;
    $\mathcal{N}_{p_{e_i}} \longleftarrow \text{neighbourhood}(p_{e_i})$;
    $MSE \longleftarrow 0$;
    for $(p_{s_{i,j}}, p_{e_{i,j}}) \in \mathcal{N}_{p_{s_i}} \times \mathcal{N}_{p_{e_i}}$ **do**
      $MSE \longleftarrow MSE + (Y(p_{s_{i,j}}) - Y(p_{s_{i,j}}))^2$;
    $MSE \longleftarrow \frac{MSE}{|\mathcal{N}_{p_{s_i}}|}$;
    $PSNR \longleftarrow 10 \log \frac{255^2}{MSE}$;
    if $PSNR \geq \tau_{PSNR}$ **then**
      $\mathcal{H}_S \longleftarrow \mathcal{H}_S \cup \mathcal{N}_{p_{s_i}}$;
      $\mathcal{H}_E \longleftarrow \mathcal{H}_E \cup \mathcal{N}_{p_{e_i}}$;
      $finished \longleftarrow false$;
$\mathcal{M} \longleftarrow \mathcal{H}_S \cup \mathcal{H}_E$;
return $\mathcal{M}$

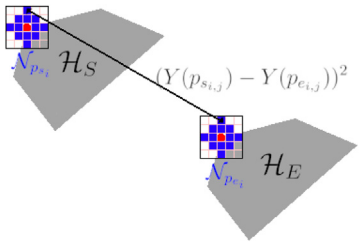

FIGURE 3 | Expansion of two convex hulls $\mathcal{H}_S$ and $\mathcal{H}_E$ by computing the *MSE* between the $\mathcal{N}_{p_{s_i}}$ and $\mathcal{N}_{p_{e_i}}$ neighborhoods of two matched keypoints marked in red.

With a function border returning the pixels forming the boundary hull of a convex hull, we can then extract two sets of points B_S and B_E representing respectively the boundaries of the two convex hulls $\mathcal{H}_S$ and $\mathcal{H}_E$. For each couple of points $(p_{s_i}, p_{e_i}) \in B_S \times B_E$, we then compute its neighborhood $\mathcal{N}_{p_{s_i}}$ and $\mathcal{N}_{p_{e_i}}$, as illustrated in **Figure 3**. The $\mathcal{N}_{p_{s_i}}$ and $\mathcal{N}_{p_{e_i}}$ neighborhoods are computed from two circular areas of radius $k \in \mathbb{N}$ pixels around the start point p_{s_i} and the end point p_{e_i} respectively. We then compute the mean squared error (*MSE*) between $\mathcal{N}_{p_{s_i}}$ and $\mathcal{N}_{p_{e_i}}$:

$$MSE = \frac{1}{|\mathcal{N}_{p_{s_i}}|} \sum_{-\lfloor \frac{k}{2} \rfloor \leq j \leq \lfloor \frac{k}{2} \rfloor} \left(Y(p_{s_{i,j}}) - Y(p_{e_{i,j}})\right)^2, \tag{10}$$

where $Y(p)$ is the greylevel of a point p, and k the radius of the circular areas around $p_{s_i}$and p_{e_i}, as illustrated in **Figure 3**.

For the first iterations, since the two convex hulls are entirely inside the tampered areas, the value of *MSE* (**Eq. 10**) is very near or equal to 0. Indeed, because of the copy-move forgery, the two areas $\mathcal{N}_{p_{s_i}}$ and $\mathcal{N}_{p_{e_i}}$ contain the same pixel values. At the end of each iteration, from the *MSE* we can calculate the *PSNR* between $\mathcal{N}_{p_{s_i}}$ and $\mathcal{N}_{p_{e_i}}$ as:

$$PSNR(\mathcal{N}_{p_{s_i}}, \mathcal{N}_{p_{e_i}}) = 10 \log \frac{255^2}{MSE}. \tag{11}$$

As presented in **Algorithm 2**, if the *PSNR* value is greater than a threshold τ_{PSNR}, then we merge the areas $\mathcal{N}_{p_{s_i}}$ and $\mathcal{N}_{p_{e_i}}$ respectively to the convex hulls $\mathcal{H}_S$ and $\mathcal{H}_E$.

After several expansions of $\mathcal{H}_S$ and $\mathcal{H}_E$ they reach the border of the real tampered areas. In this case, the expansion of $\mathcal{H}_S$ and $\mathcal{H}_E$ no longer contains exactly the same pixels. In this case the *PSNR* value decreases and becomes lower than the threshold τ_{PSNR}. The convex hull expansion algorithm stops and the binary mask $\mathcal{M}$ for this tampered area is returned. If more than one area has been tampered, we repeat this algorithm for each cluster in order to get the final binary mask $\mathcal{M}$.

4 RESULTS AND DISCUSSION

In this section, we present several experimental results obtained with our proposed VFCMFD method. First, in **Section 4.1**, we

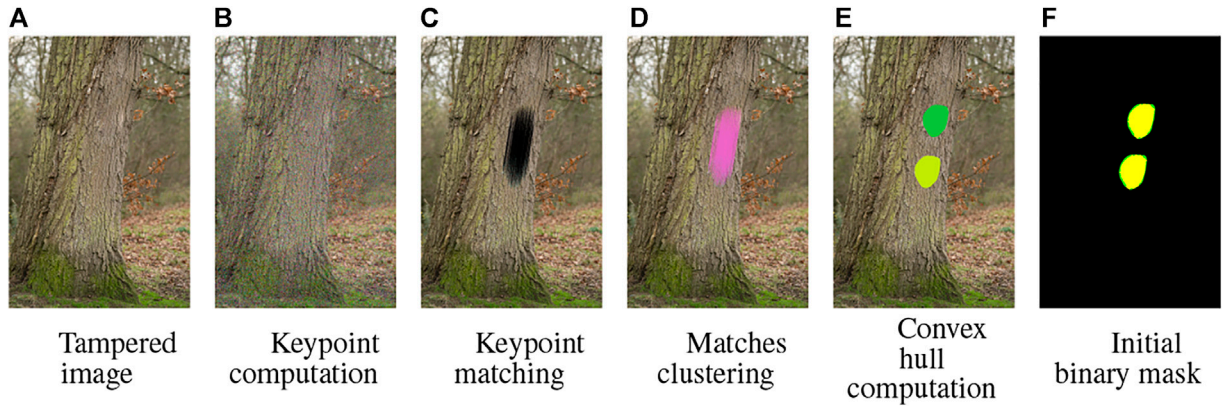

FIGURE 4 | The results obtained on a image from the DGA database: **(A)** A tampered image (2640 × 2360 pixels), **(B)** 117,631 keypoints are detected, **(C)** 809 keypoints are matched, **(D)** From 803 matched keypoints 1 cluster is generated, **(E)** Computation of the two convex hulls, **(F)** Comparison with the ground truth map, Dice index = 0.94 (pixels in yellow are true positives, pixels in red false positives and pixels in green false negatives).

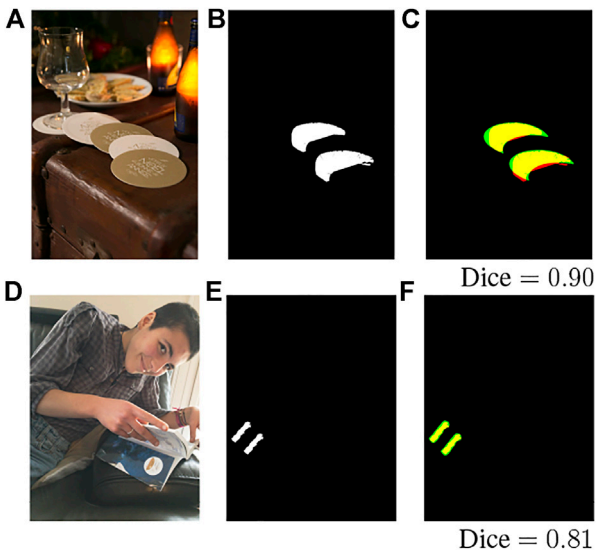

FIGURE 5 | Two other tampered images from the DGA database of size: **(A)** 2640 × 2360 pixels), **(D)** 3024 × 4032 pixels), **(B)** and **(E)** Detection of the tampered areas, **(C)** and **(F)** Comparison with the ground truth maps.

present the databases used for our experimentation as well as metrics used to evaluate the performances of the method we propose. In **Section 4.2** we present a full example of our method and in **Section 4.3** we compare it with existing state of the art ones. In **section 4.4**, we analyze the performance of our proposed VFCMFD method, first in terms of computation time and results obtained as a function of the threshold τ_θ of the angle, and then in terms of robustness against noise addition and JPEG compression attacks.

4.1 Databases and Criteria

Two databases designed for the copied-moved forgery detection have been used to perform our experimentation. The first one is the FAU database, which was created in 2012 by Christlein et al. (2012). It includes 48 source images taken by different cameras. The aim of this database is to provide a common measurement for each copy-move forgery detector by providing large images with an average size of 3000 × 2300 pixels and realistic copy-move forgeries with tampered areas around 10% of the image size. This database provides different types of copy-move attacks which are 48 **plain copy-move**, where each image is tampered with a plain copy-move, **rotation**, where the tampered areas are rotated between 2° and 10°(with a step of 2°) before being translated, **scaling**, where the tampered areas are scaled with different scale factors between 0.91 and 1.09 with a step of 0.02, **JPEG compression**, where the tampered area undergoes JPEG compression with different quality factors (QF) between 100 and 20% with a step size of 10 before being pasted, **noise addition**, where the images are normalized and a zero-mean Gaussian noise is added with a standard deviation between 0.02 and 0.10 with step size 0.02, **combined transformations**, where the images undergo several attacks from the previously mentioned ones, and finally **multiple copies**, where a square block of size 68 × 68 pixels are copied and pasted five times in each image. The second database used is the GRIP one which was created in 2015 by Cozzolino et al. (2015a). Its creation was motivated by the need to set up a bank of images that would allow realistic and accurate forgeries to be performed in a reasonable amount of time to validate the different methods and to adjust the different parameters. All 80 images provided have the same size of 1024 × 768 pixels. It consists of plain copy-move attacks with manipulation that are difficult to detect, such as forgeries made on very small areas (less than 1% of the image) or in very smooth areas. For our experiments we also worked on 10 images of a very large size provided by the DGA[1]. Five of them have a size of around 2600–3000,× 2400–4000 pixels and the five others around 6000 × 4000 pixels. Images illustrated in **Figure 4A**, **Figure 5A** and **Figure 5D** are from this database.

[1]The DGA (Direction Générale de l'Armement) is a partner in our project: https://www.defense.gouv.fr/dga.

TABLE 1 | Parameters used for our proposed VFCMFD method.

Parameters	Value	Part
τ_{SURF}	0	Keypoint extraction (**Section 3.2**)
τ_{g2NN}	0.5	Fast g2NN (**Section 3.3**)
τ_θ	4°	
w_ℓ	0.25	Clustering DBSCAN (**Section 3.4**)
w_ϕ	0.25	
w_x	0.25	
w_y	0.25	
ϵ	0.5	
minPts	10	
τ_ℓ	50 pixels	
τ_{PSNR}	100 *dB*	Binary mask generation (**Section 3.5**)
Radius *k*	13 pixels	

We perform tests at image and pixel level. At image level the goal is to measure how well a method can make the distinction between tampered images and original ones. To test this, we took the original images from each database as well as the forged ones. The tests were carried out on 160 images from the GRIP database and on 96 images for the FAU database. Measurements were completed at pixel level and can be used to measure the quality of detection of tampered areas in a non-authentic image. Indeed, a method can detect an image as non-identical, but indicate pixels that are not involved in the forgery. To test this, we took the tampered images from each base (48 for FAU one and 80 for GRIP one).

The metrics used to evaluate the performance are precision, recall, F1-score, Dice and Jaccard indices. We define the number of forged pixels (resp. images) that have been truly detected (*TP*), the number of pixels (resp. images) that have been wrongly detected as forged (*FP*) and finally the pixels (resp. images) that are wrongly detected as authentic (*FN*). Precision refers to the rate of pixels truly detected as tampered out of all pixels detected as tampered: $precision = \frac{TP}{TP+FP}$. A precision close to 1 means that areas classified as tampered have a probability close to 1 to be tampered areas. Recall refers to the rate of pixels truly detected as tampered out of all tampered pixels: $recall = \frac{TP}{TP+FN}$. A recall value close to 1 means that all tampered areas have been detected as such. The F1-score is a measure that combines both precision and recall: $F1 - score = \frac{2 \times precision \times recall}{precision+recall}$. The Jaccard index indicates the rate of true positives on pixels that are detected as tampered or that are really tampered. It is a tool to evaluate the similarity between two binary masks: $Jaccard = \frac{TP}{FN+FP}$. The Dice index is double the pixels truly detected as tampered, over the sum of the pixels detected as tampered or actually tampered. Like the Jaccard index, this gives a good measurement for evaluating the performance of the method: $Dice = \frac{2 \times TP}{2 \times TP+FN+FP}$.

We have shown in **Section 3** that parameters are needed for the different steps of our method. In **Table 1**, we show the values we used for each of these parameters to perform our experiments. While for the keypoint extraction step (**Section 3.2**) we set $\tau_{SURF} = 0$, and for the fast g2NN (**Section 3.3**) we set $\tau_{g2NN} = 0.5$ and $\tau_\theta = 4°$, for the clustering step (**Section 3.4**), regarding equ. 7, we set the four weights to 0.25, *minPts* = 10 and $\tau_\ell = 50$ pixels. For the binary mask generation (**Section 3.5**), for the expansion we set $\tau_{PSNR} = 100\,dB$ and the radius $k = 13$ pixels.

4.2 A Full Example

As the objective is to process 4K images, in this section first we apply our algorithm to a large image (2640 × 2360 pixels), illustrated in **Figure 4**, from the DGA database. In **Figure 4A** the image representing a tree trunk is altered in the area of the tree bark. This image is a new challenge, because of the patterns formed by the tree bark and makes it much more difficult for CMFD algorithms to find forgeries. **Figure 4** shows the results obtained after each step of our method until the convex hull is obtained. In **Figure 4B**, 117, 631 keypoints are detected and among all of these keypoints, 809 are matched (**Figure 4C**). From these 809 matches, 803 are selected to generate 1 cluster as illustrated in **Figure 4D**, from which the two convex hulls are computed (**Figure 4E**).

Finally, **Figure 4F** shows a comparison between the mask we obtain with our method and the ground truth mask. The pixels in yellow correspond to the true positives, while the pixels in red are the false positives and the pixels in green, the false negatives. We can observe that our method correctly detects and locates the tampered areas and does not produce many false positives.

Figure 5 illustrates two other examples applied to images from the DGA database. The first image illustrated **Figure 5A**, of size 2640 × 3960 pixels, shows a dark scene with several similar coasters. One of these coasters has been copied and moved while the others remain unchanged. The main difficulty is not to confuse the original coasters by making false positives. Our proposed algorithm successfully detects and locates the two areas involved in the forgery (Dice index = 0.90), as shown in **Figure 5B**. In **Figure 5C** to compare with the ground truth mask, the pixels in yellow are the true positives. The second image illustrated in **Figure 5D**, of size 3024 × 4032 pixels, shows a very small forgery on the belt loops of the jeans. The tampered area is very small compared to the size of the image (less than 0.5%). The difficulty lies in the size of the forgery, but also in the pattern of shirt which can generate false positives. Our VFCMFD method also succeeds in correctly locating the forgery with a Dice index of 0.81.

4.3 Comparison With Previous Work

In this section, we compare the results obtained by our VFCMFD method with previous ones. First, we present comparisons with challenging images, then we compare our method with previous ones on the whole FAU database and finally on the whole GRIP database.

4.3.1 Comparison on Challenging Images

In this part we analyze the results obtained on copied-moved images from the GRIP and FAU databases which have been tested by Wang et al. (2019). These images are specially challenging because the TP_C01_023 forgery occurs in smooth regions, the TP_C01_005 and TP_C01_30 occur in high-brightness smooth regions, the TP_C01_024 and the TP_C01_049 images (all four from the GRIP database) occurs in images with similar patterns

TABLE 2 | Comparison of the F1-scores at pixel level obtained by our method with(Cozzolino et al., 2015a; Pun et al., 2015; Zandi et al., 2016) and(Wang et al., 2019) methods on challenging images.

Methods	Cozzolino et al. (2015a)	Zandi et al. (2016)	Pun et al. (2015)	Wang et al. (2019)	Proposed
TP_C01_023	0.97	0.97	0.23	0.96	**0.99**
TP_C01_005	0.00	0.88	0.00	0.94	**0.98**
TP_C01_030	0.82	0.82	0.21	0.92	**0.97**
TP_C01_024	0.72	0.83	0.85	0.97	**0.99**
TP_C01_049	0.16	0.87	0.54	0.96	**0.99**
Bricks	0.97	0.95	0.89	0.96	**0.98**

The values in bold are the F1 scores obtained using our method on the different databases (6 difficult images, FAU and GRIP)

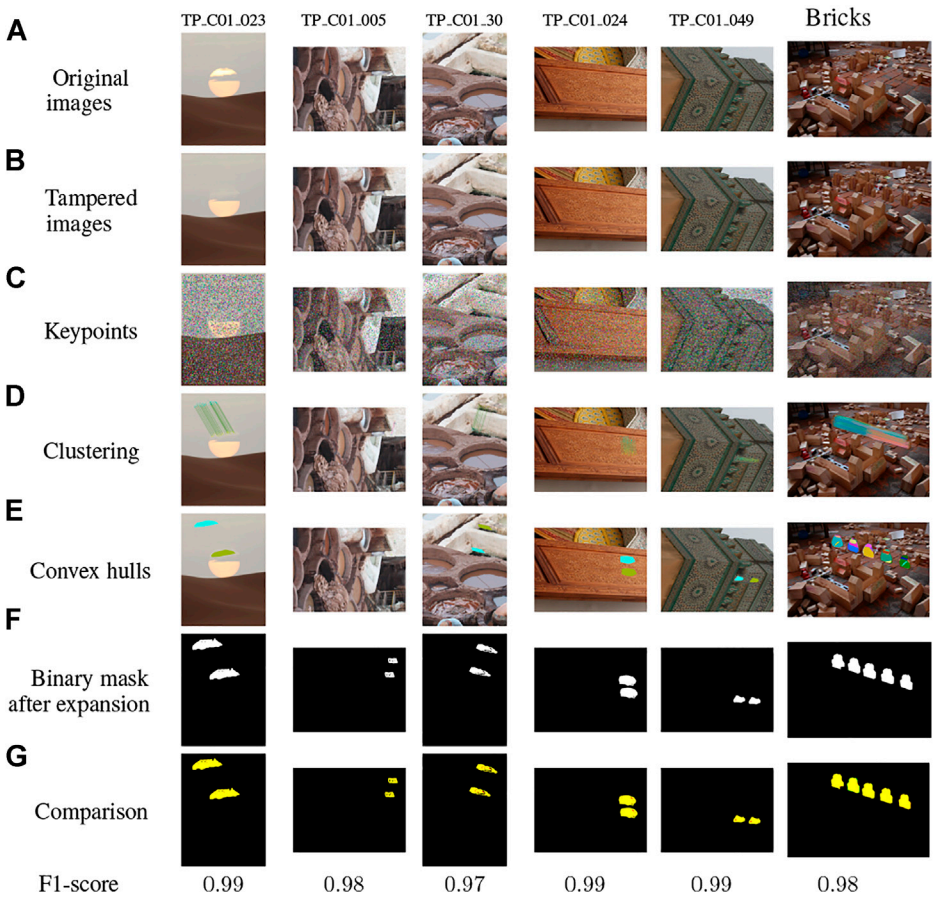

FIGURE 6 | Results obtained from each step of our proposed VFCMFD method using challenging images. **(A)** Original images; **(B)** Tampered images; **(C)** Keypoints; **(D)** Clustering; **(E)** Convex hulls; **(F)** Binary mask after expansion; **(G)** Comparison.

and the image Bricks (from the FAU database) which occurs in multiple regions. Our method is compared with the previous methods of (Cozzolino et al., 2015a; Pun et al., 2015; Zandi et al., 2016) and (Wang et al., 2019).

First, **Table 2** presents the results obtained with our method for these challenging images compared to the methods (Cozzolino et al., 2015a; Zandi et al., 2016; Pun et al., 2015; Wang et al., 2019). **Table 2** shows that our VFCMFD method is extremely accurate even when the forgery is carried out on a very smooth area like the sky. The good results obtained in terms of F1-score by our method (between 0.97 and 0.99) can be attributed to the large number of keypoints generated by setting the τ_{SURF} threshold to 0 and other appropriate parameters. For the TP_C01_023 image, the results obtained with our method (F1-score = 0.99) are very similar and even slightly higher than those obtained by (Cozzolino et al., 2015a; Zandi et al., 2016) and (Wang et al., 2019). In TP_C01_005 and TP_C01_030, the difficulty is that very few keypoints are detected because the

TABLE 3 | Comparison, in terms of F1-score at pixel level, obtained by our method with the methods (Huang et al., 2008; Amerini et al., 2011; Shivakumar and Baboo, 2011; Li et al., 2015; Pun et al., 2015; Zandi et al., 2016; Li and Zhou, 2019; Mei et al., 2019; Chen et al., 2020; Lyu et al., 2021) and (Diwan et al., 2019) on the entire FAU database.

Methods	Pixel
	F1-score
Huang et al. (2008)	0.6354
Shivakumar and Baboo (2011)	0.6954
Li et al. (2015)	0.7447
Lyu et al. (2021)	0.8142
Zandi et al. (2016)	0.8607
Amerini et al. (2011)	0.8780
Li and Zhou (2019)	0.8838
Proposed method	**0.8963**
Pun et al. (2015)	0.8997
Mei et al. (2019)	0.9054
Chen et al. (2020)	0.9924
Diwan et al. (2019)	0.9976

The values in bold are the F1 scores obtained using our method on the different databases (6 difficult images, FAU and GRIP)

tampered areas are small and occur in a high-brightness, smooth region. Our method obtains similar results than those of (Wang et al., 2019) and better results than (Cozzolino et al., 2015a; Pun et al., 2015) and (Zandi et al., 2016). Images TP_C01_024 and TP_C01_049 contain forged areas that are very similar to genuine areas. The CMFD methods can generate false positives and therefore has a low precision. Our method provides good results due to fast feature g2NN matching process and the DBSCAN filtering step. Our method even succeeds in obtaining better results than those obtained by (Wang et al., 2019). In the image Bricks (from the FAU database), the same area is copied and moved four times. For these images, our method also provides very good results due to the fast feature g2NN matching process and the DBSCAN filtering step.

In **Figure 6**, we detail the results obtained for each step of our method for these challenging images. While in **Figure 6A**, the original images are presented (from left to right: TP_C01_023, TP_C01_005, TP_C01_030, TP_C01_024, TP_C01_049 and Bricks), in **Figure 6B** we can see the corresponding tampered images. The extracted SURF keypoints and feature computation are illustrated in **Figure 6C**, and the results of our proposed fast g2NN matching and clustering algorithms are shown in **Figure 6D**. The convex hulls are then obtained, as illustrated in **Figure 6E** from which we can generate initial binary masks before expansion. Finally the expanded convex hulls are illustrated in **Figure 6F**, and we can compare them with the original ground truth maps (**Figure 6G**). Even if these images are very challenging, we can observe that the results obtained are visually very good and confirmed by a F1-score between 0.97 and 0.99.

4.3.2 Comparison With the Entire FAU Database

In this part we compare our method with previous ones on the entire FAU database. To do this, we compare our method with the previous methods of (Huang et al., 2008; Amerini et al., 2011; Shivakumar and Baboo, 2011; Li et al., 2015; Pun et al., 2015; Zandi et al., 2016; Li and Zhou, 2019; Mei et al., 2019; Chen et al., 2020; Diwan et al., 2019) and(Lyu et al., 2021).

Table 3 presents the results obtained in terms of F1-score at pixel level. At pixel level, our method is very efficient with an average F1-score of **0.8963**. The scores obtained are better than those of the methods (Huang et al., 2008; Shivakumar and Baboo, 2011; Li et al., 2015; Zandi et al., 2016; Lyu et al., 2021) with F1-scores between 0.6354 and 0.8607, and is comparable to the methods of (Amerini et al., 2011; Pun et al., 2015; Li and Zhou, 2019) and (Mei et al., 2019) which have respectively an F1-score of 0.8780, 0.8997, 0.8838, and 0.9054. Only the (Diwan et al., 2019) and (Chen et al., 2020) methods perform better and have respectively an F1-score of 0.9924 and 0.9976. The first approach is a block-based method that uses the LLP (Local Linear Projection) while the second is a keypoint-based method which starts by extracting SIFT points before classifying them by scale and colour. Even though these methods have higher F1-scores, ours is much faster.

At image level, our method detects all tampered images. Indeed, it does not produce any false negatives (FN = 0/48) but some original images are detected as forged due to false keypoint matching (FP = 22/48). The average F1-score at the image level is 0.8136. Although our method is less efficient than recent methods such as the one proposed by Lyu et al. (2021) which has an F1-score of 0.8624, it outperforms methods proposed by Zandi et al. (2016) and (Li et al., 2015), which have F1-scores of 0. 7934 and 0.7447.

4.3.3 Comparison on the Entire GRIP Database

In this part we compare our method with the method of (Bravo-Solorio and Nandi, 2009; Amerini et al., 2011; Cozzolino et al., 2015a; Li et al., 2015; Silva et al., 2015; Zandi et al., 2016; Diwan et al., 2019; Li and Zhou, 2019; Meena and Tyagi, 2020; Tahaoglu et al., 2021; Chen et al., 2020) and (Gupta and Singh, 2021) on the entire GRIP database.

Table 4 presents the results obtained in terms of F1-score at pixel levels. Our method obtains an average F1-score of **0.9606**

TABLE 4 | Comparison, in terms of F1-score at pixel level, obtained by our method with the methods (Bravo-Solorio and Nandi, 2009; Amerini et al., 2011; Cozzolino et al., 2015a; Li et al., 2015; Silva et al., 2015; Zandi et al., 2016; Diwan et al., 2019; Li and Zhou, 2019; Meena and Tyagi, 2020; Chen et al., 2020; Tahaoglu et al., 2021) and (Gupta and Singh, 2021) on the entire GRIP database.

Methods	Pixel
	F1-score
Li et al. (2015)	0.2774
Silva et al. (2015)	0.6662
Zandi et al. (2016)	0.6444
Bravo-Solorio and Nandi (2009)	0.8482
Cozzolino et al. (2015a)	0.9299
Tahaoglu et al. (2021)	0.9300
Li and Zhou, (2019)	0.9466
Diwan et al. (2019)	0.9469
Amerini et al. (2011)	0.9538
Meena and Tyagi (2020)	0.9581
Proposed method	**0.9606**
Gupta and Singh, (2021)	0.9748
Chen et al. (2020)	0.9972

The values in bold are the F1 scores obtained using our method on the different databases (6 difficult images, FAU and GRIP)

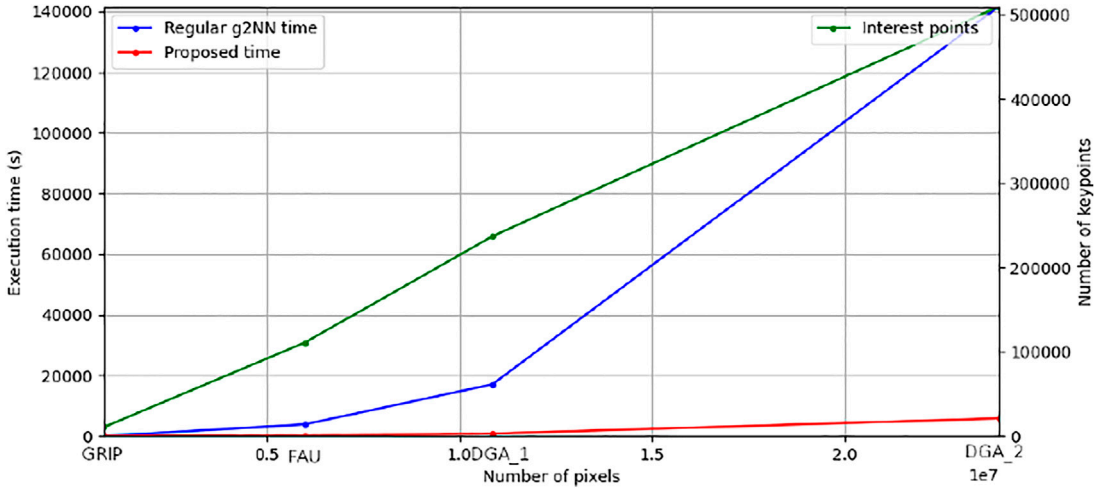

FIGURE 7 | Computation time as a function of the image size.

for the GRIP database, which is very efficient. We obtain better results than the methods of (Bravo-Solorio and Nandi, 2009; Amerini et al., 2011; Cozzolino et al., 2015a; Li et al., 2015; Silva et al., 2015; Zandi et al., 2016; Li and Zhou, 2019; Diwan et al., 2019) and (Tahaoglu et al., 2021) with F1-scores between 0.2774 and 0.9469. The methods of (Amerini et al., 2011) and (Meena and Tyagi, 2020) which have respectively an F1-score of 0.9538 and 0.9581 are comparable to our method. Only the solutions of (Gupta and Singh, 2021), and (Chen et al., 2020) obtain higher results with F1-score of 0.9748 and 0.9972. This said, our results stay close to theirs and as mentioned in **Section 4.3.2**, our method is much faster.

At the image level, our method detects almost all tampered images as such. Of the 80 images in the GRIP database, the method generates only one false negative on image TP_C01_14. The tampered area of this image is located at sky level. The method detects only 14 matches, of which only one is related to forgery. Forgeries in blurred and uniform areas remain a real challenge for keypoints-based methods. Compared to the FAU database, the method generates some false positives ($FP = 26/80$). These are due to matches made in similar areas in the original studied image, such as on patterns. The average F1-score at the image level is 0.8541.

We can conclude that the very good results obtained by our method can be explained by the large number of keypoints detected as well as by our efficient fast feature matching algorithm. Note that the distances used in our VFCMFD method (**Table 1**) are suitable for plain copy-move forgery detection. Even if we decrease the complexity ($\mathcal{O}(kn)$ instead of $\mathcal{O}(n^2)$), our VFCMFD method produces results as good as other state-of-the-art methods in plain copy-move forgery detection.

4.4 Performance Analysis

All the results presented in this paper have been computed using OpenCV 4.0 and C++17 on a 64-bit Ubuntu 16.04.1 with the Intel® Core™ i7-7820X CPU, 16 cores, at frequency of 3.60 GHz and 110 GB RAM. The fast feature matching algorithm described in **Section 3.3** is parallelized by separately applying the same procedure on each keypoint. We ran all the tests using 50 threads, we split the set of keypoints in 50 parts and each thread runs the algorithm on its assigned set of keypoints. In this section, we first present the performances of our VFCMFD method in terms of computation time. Second we analyze the obtained results as a function of the threshold τ_θ of the angle. Finally we analyze the robustness of our VFCMFD method against noise addition and JPEG compression attacks.

TABLE 5 | Computational time for the different steps on an image from the FAU database.

Step/Method	Proposed Method (s)	Classical g2NN (s)
KeyPoints	2.21	1.63
Matches	177.50	3871.23
Lines	0.00	0.00
Clusters	5.04	4.92
Hull	0.00	0.00
Mask	0.00	0.00
extendMask	1.79	1.27
Total	187.71	3880.19

4.4.1 Computation Time Comparison

Our method significantly reduces the computation time of keypoint-based techniques by using a new fast feature matching algorithm. In this part, we analyse the performance of our method compared to the standard g2NN proposed by Amerini et al. (2011).

We provide, in **Figure 7**, as a function of the image size in pixels, a plot in green showing the number of keypoints, in blue the computation time of the standard g2NN matching algorithm (Amerini et al., 2011) (with a complexity in $\mathcal{O}(n^2)$), and in red the computation time of our method. In **Figure 7**, the tests were carried out on 80 images from the GRIP database, 48 images from the FAU database, DGA_1 which is composed of five images of a

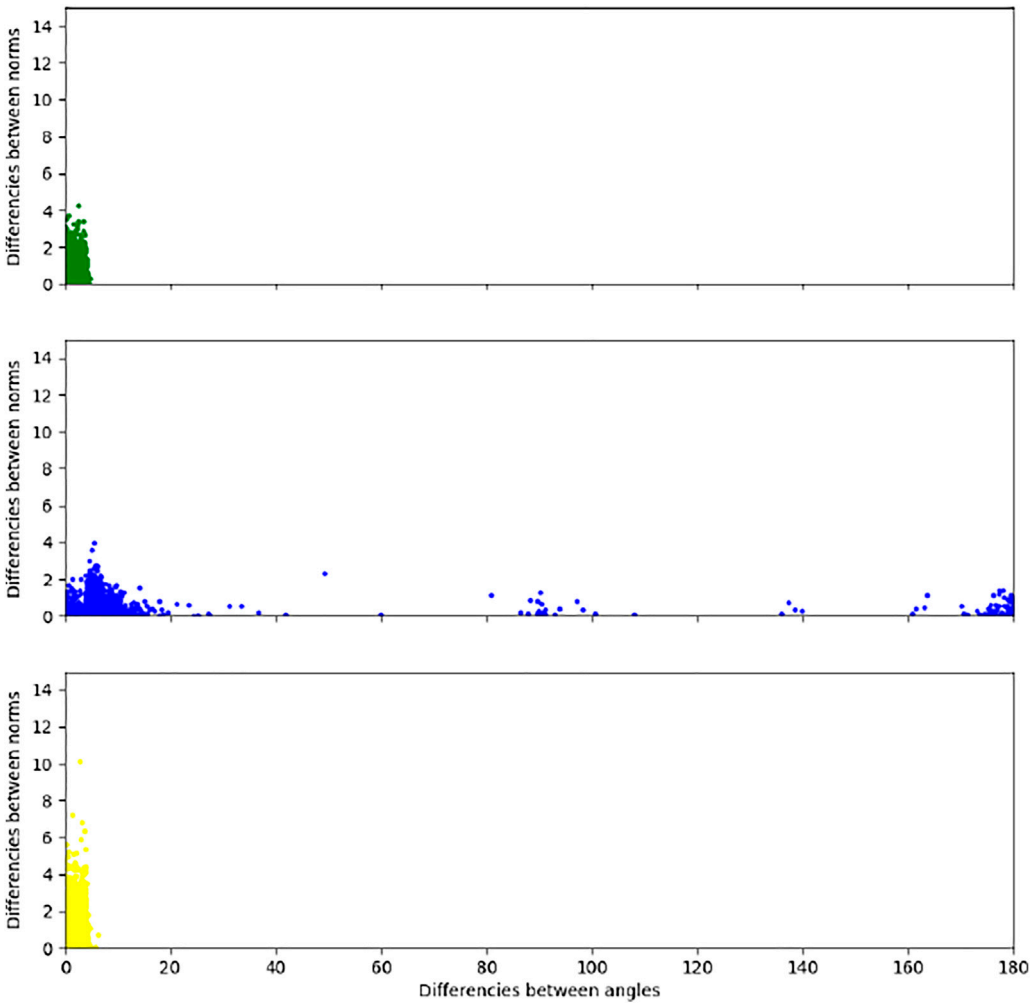

FIGURE 8 | Distribution of the matched keypoints according to their differences in angles and norms, with $\tau_{SURF} = 0$: In green, the keypoints that are matched in the same way with a standard g2NN and with our approach; In blue, the keypoints that are matched with a standard g2NN, but not with our approach; In yellow, the new keypoints that are matched with our approach, with $\tau_\theta = 4^\circ$, but not with a standard g2NN.

TABLE 6 | Same, false and new matches found by both algorithms on GRIP database (Cozzolino et al., 2015a) and on FAU database (Christlein et al., 2012).

GRIP Database	In cluster	Outs. Cluster	Total
Same Matches	7506	25,447	32,953
—	94.02%	67.69%	72.30%
False matches	0	0	0
—	0.00%	0.00%	0.00%
New matches	477	12,148	12,625
—	5.98%	32.31%	27.70%
FAU Database	**In cluster**	**Outs. Cluster**	**Total**
Same Matches	184,408	111,490	295,898
—	92.74%	68.55%	81.86%
False matches	3571	167	3738
—	1.80%	0.10%	1.03%
New matches	10,859	50,980	61,839
—	5.46%	31.35%	17.11%

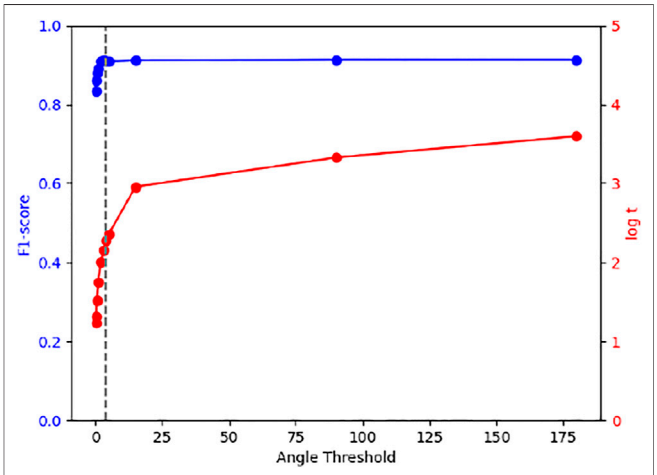

FIGURE 9 | Obtained F1-score and computation time as a function of the angle threshold used for our proposed approach.

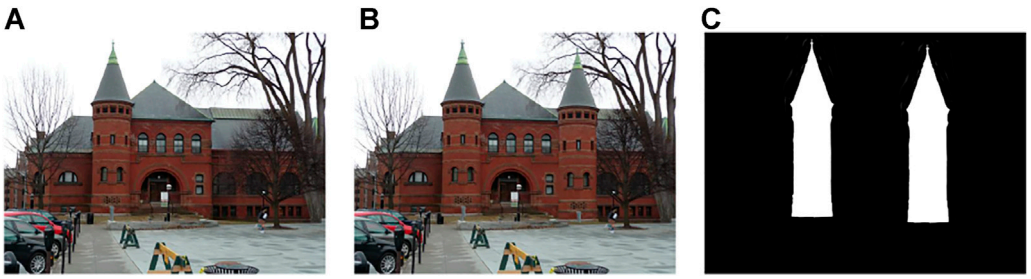

FIGURE 10 | Red_tower image from FAU database (Christlein et al., 2012): **(A)** Original image, **(B)** Tampered image and **(C)** Detection of the tampered areas.

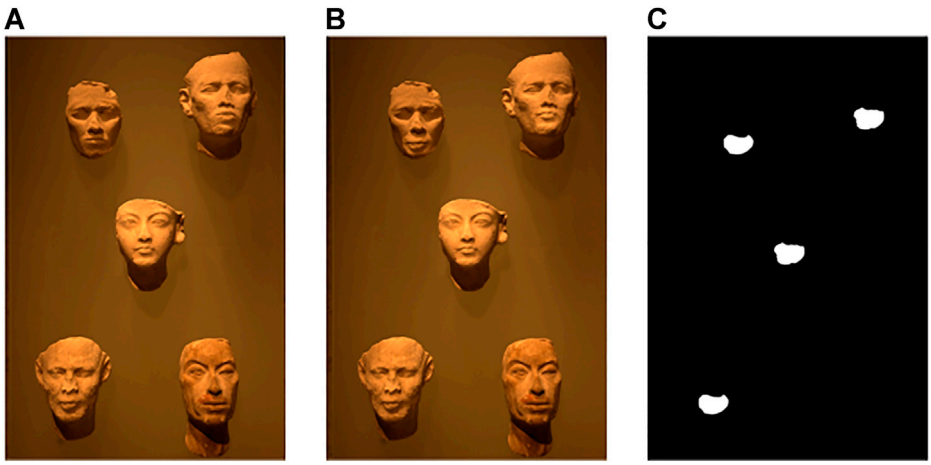

FIGURE 11 | Mask image from FAU database (Christlein et al., 2012): **(A)** Original image, **(B)** Tampered image and **(C)** Detection of the tampered areas.

size of around 2600–3000 × 2400–4000 pixels and DGA_2 which is composed of five images of a size around 6000 × 4000 pixels. As we can see in **Figure 7**, the computation time increases drastically when using the standard g2NN matching algorithm (Amerini et al., 2011) with large images. In particular, for very large images from the DGA_2 database with more than 24 million of pixels and more than 500,000 keypoints extracted, the average computation time exceeds 140,000 s corresponding to 38.9 h (more than 1.5 days). With our method the necessary computation time for such an image is on average 5830 s which is around 1.6 h. The gain in time is mainly due to the matching phase.

In order to evaluate the performance of our approach, **Table 5** presents the average times for each phase on an image from the FAU database. We can observe that the computation time necessary for the matching phase is therefore considerably reduced, thanks to our efficient method. This is due to the fact that for this step we use a threshold τ_θ for the angle to limit the number of matches.

4.4.2 Analysis of the Threshold τ_θ of the Angle

Even if our VFCMFD method strongly reduces the computation time, the results obtained are comparable to those of other state of the art methods as presented in **Section 4.3**. Indeed, the main novelty of our fast matching algorithm is the reduction of the number of comparisons made between different keypoints. Our VFCMFD method uses a threshold to select the keypoints that are most likely to match. The higher the threshold, the more comparisons are needed.

In **Figure 8**, we compare the matches obtained by a standard g2NN algorithm with our proposed fast matching algorithm on the entire database GRIP (Cozzolino et al., 2015a). Then, for all matched keypoints, we compute the difference between the norm of their descriptors and the difference between their angles. In **Figure 8**, we plot the difference between the norms of the keypoint descriptors as a function of the difference angles. In green, we represent the keypoints that are matched in the same way with a standard g2NN and with our approach. In blue, we represent the keypoints that are matched with a standard g2NN, but not with our approach. And in yellow, we represent the new keypoints that are matched with our approach but not with a standard g2NN. This means that our method eliminates the matching of keypoints with angles that differ greatly, and thus eliminates matches related to similarities in the image. With our approach, there are no keypoints that would have been selected

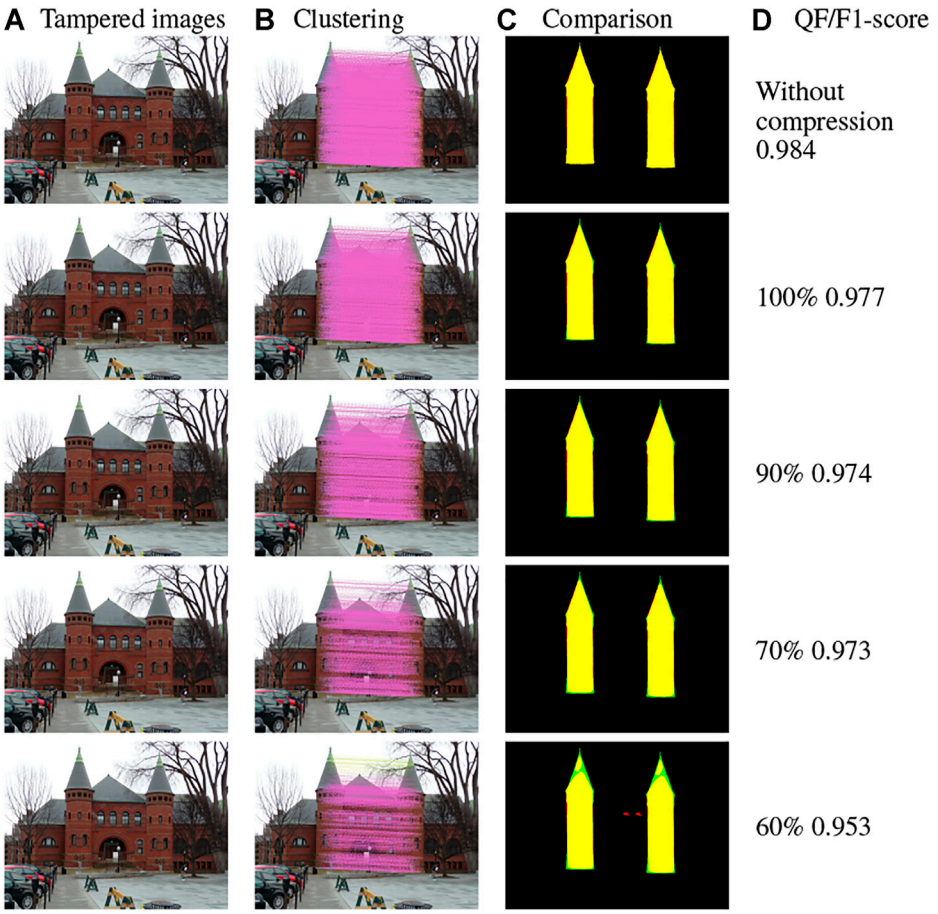

FIGURE 12 | Tampered images, clusters found and comparison between the obtained mask by our method and the ground truth for the Red_tower image from the FAU database using various JPEG QF. **(A)** Tampered images; **(B)** Clustering; **(C)** Comparison; **(D)** QF/F1-Score.

with a standard g2NN and matched with another point in our method.

We can observe in **Figure 8** that the majority of the matches are close to (0, 0). This means that the matches are usually made between keypoints whose orientations are very similar. Consequently, instead of comparing a keypoint to all the other keypoints, it is very interesting to compare it to the keypoints of angle lower than the threshold τ_θ. As presented in **Table 1**, for our experiments using our fast g2NN method we have used $\tau_{SURF} = 0$ and $\tau_\theta = 4^\circ$. As we can observe, our proposed VFCMFD method correctly matches the majority of keypoints, but a lot of other matches appear as well.

Table 6 presents the percentages of the same, the false and the new matches between our method and the standard g2NN, respectively for the GRIP database and the FAU database. The first row "Same Matches" represents the keypoints that have been matched by the standard g2NN method and are still matched by the new method, the second row "False Matches" represents the keypoints that are mapped by the standard g2NN method and that are mapped to another keypoint with our proposed fast g2NN while the last row "New Matches" represents the keypoints that have not been matched by the standard g2NN method and that are matched with our proposed fast g2NN method. We can notice that with our proposed VFCMFD method, most of the matches taking place in clusters are keypoints that are matched by a standard method. But more than 30% of the matches outside clusters are new matches. These new matches can be explained by the ratio used when applying the g2NN. For the GRIP database, in **Table 6**, we can note that 0 false matches were detected.

Figure 9 illustrates the F1-score obtained and the computation time as a function of the angle threshold used for our proposed approach. We can observe that after a threshold of 4°, while the computation time continues to increase, the F1-score hardly changes.

This analysis enables us to conclude that the approach we propose is very efficient in terms of F1-score and allows us to drastically reduce the computation time.

4.4.3 Performance Analysis of Robustness Against Noise Addition and JPEG Compression Attacks

In this section, we study the robustness of our VFCMFD method on images that underwent JPEG compression or Gaussian noise attacks. All the tests were performed on

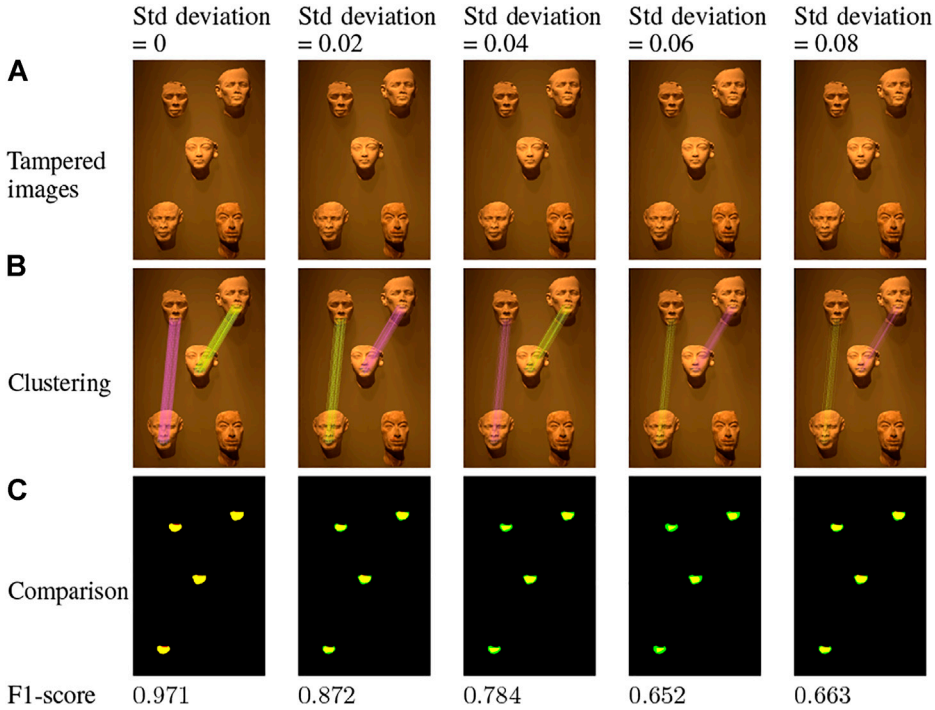

FIGURE 13 | Tampered images, clusters found and comparison between the obtained mask by our method and the ground truth for the Mask image from the FAU database using various Gaussian noises. **(A)** Tampered images; **(B)** Clustering; **(C)** Comparison.

both the entire FAU database (Christlein et al., 2012) and the entire GRIP database Cozzolino et al. (2015a). The performance of the proposed method is illustrated in detail on two images from the FAU database. The Red_tower image, **Figure 10**, is used to illustrate the robustness against JPEG attacks, while the Mask image, **Figure 11**, is used for the noise addition analysis.

For each figure, we present a set of three images provided by the FAU database (Christlein et al., 2012): the original images, **Figure 10A** and **Figure 11A**, the tampered images, **Figure 10B** and **Figure 11B**, and the binary masks locating the tampered areas in white, **Figure 10C** and **Figure 11C**. In the first set of images, the red tower has been duplicated (**Figure 10B**), while in the second set, the mouth of two faces have been substituted by the mouth of two other faces (**Figure 11B**).

Figure 12 presents the results obtained on the image illustrated **Figure 10B** when using JPEG compression with quality factor (QF) ranging from 100 to 60%. Images in **Figure 12A** illustrate the tampered images according to the applied QF. Therefore, the two areas involved in the tampering are not completely similar, making the detection more difficult. A JPEG attack can be considered realistic and undetectable by the human visual system (HVS) when the QF is greater than or equal to 70%. Images in **Figure 12B** show the clusters detected in each attacked image, each color corresponds to a different cluster. As the QF decreases, fewer clusters are detected because the g2NN algorithm finds fewer matches due to JPEG compression. Indeed, the detector finds 7, 235 matches for the image without JPEG compression against 1, 662 matches for JPEG compression with a QF of 60%. However, as illustrated in images of **Figure 12B** the number of matches remains high regardless of the QF from 100 to 60%. Above a QF of 70%, only one cluster is detected and from a QF of 60%, two clusters are observed due to the low density of matching. In images of **Figure 12C**, we compare the masks obtained by our method with the ground truth presented in **Figure 10C**. Finally, in **Figure 12D**, the F1-scores obtained are presented. We can note that the method we propose is very efficient. Indeed, whatever the QF the F1-score is always higher than 0.95 and is equal to 0.978 for a QF of 100%. These results show that the proposed VFCMFD method is robust to JPEG compression. Regarding the F1-scores obtained on the Red_tower image as a function of the JPEG QF, up to a QF of 60%, the evolution is almost constant and higher than 0.95. Below a QF of 40%, the F1-score value drops to 0.6 for a QF of 20%, the JPEG compression becomes too great and the copied and moved areas have become too distinct to be properly matched. However, in this case, as the image quality is strongly degraded, the attack is no longer realistic.

Figure 13 shows the results obtained on the image illustrated in **Figure 11B** when the copied and moved area undergoes a white Gaussian noise attack of zero mean with standard deviations between 0.02 and 0.08. Images in **Figure 13A** present the tampered images as a function of the Gaussian noise. Even for a standard deviation of 0.08, it is very difficult for the HVS to detect this attack. Images in **Figure 13B** show the clusters detected in each noisy image

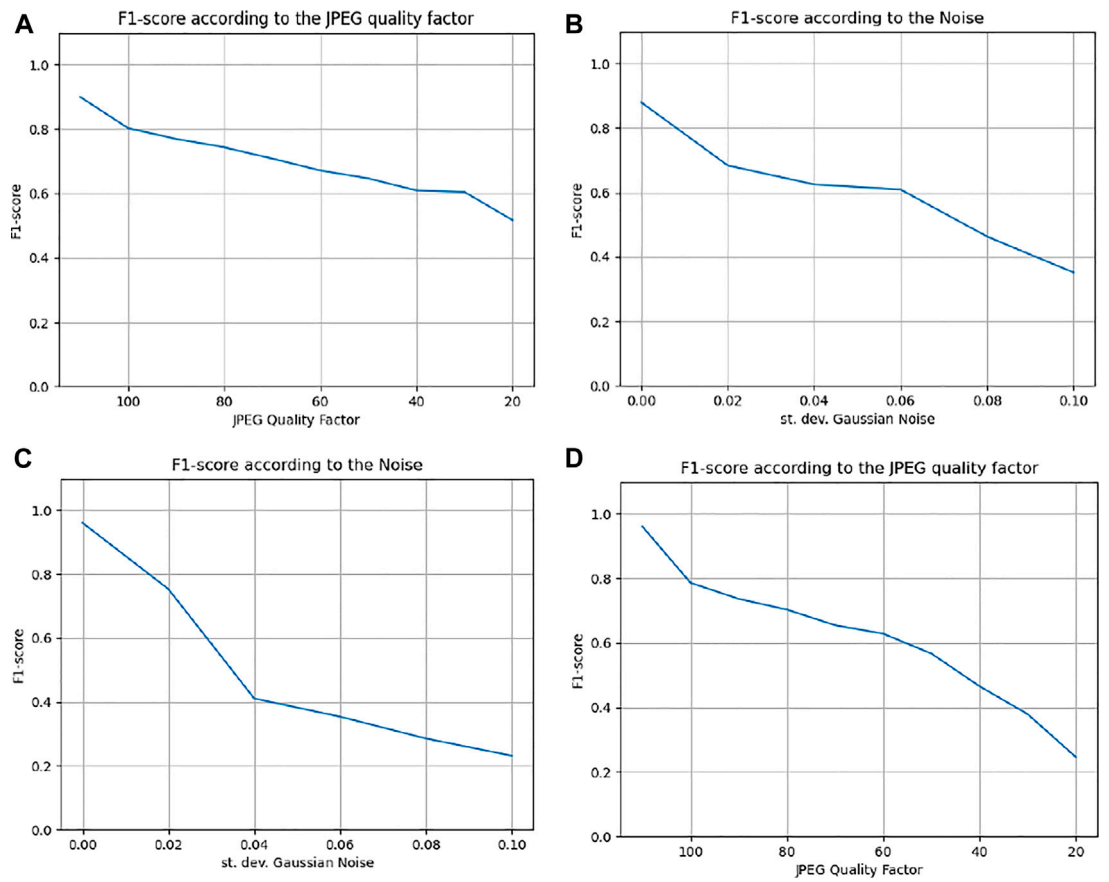

FIGURE 14 | Average F1-score obtained: **(A)** On the entire FAU database for JPEG compression, and **(B)** For Gaussian noise, **(C)** On the entire GRIP database for JPEG compression, and **(D)** Gaussian noise.

where each color corresponds to a cluster. As the standard deviations of Gaussian noise increases, fewer clusters are detected. As with JPEG compression, this is mainly because the detector finds fewer matches when the image is heavily degraded. The method finds 318 matches for the image that has not been subject to a noise addition attack against 74 for an image where additive Gaussian noise has been applied with a standard deviation of 0.08. Although the density of matches related to a forgery decreases, the method finds the two clusters related to both forgeries for all images of **Figure 13A**. In images of **Figure 13C**, we show the comparison between the masks obtained by our method with the ground truth mask presented in **Figure 11C**. Regarding the F1-score obtained, even if they decrease when the standard deviation of the Gaussian noise increases, we obtain very good performances whatever the standard deviation, as shown in **Figure 13**. Indeed, all F1-scores are higher than 0.65, and for a Gaussian noise with a standard deviation lower than 0.04, the F1-score remains higher than 0.78. Regarding the evolution of the F1-score obtained on the Mask image as a function of the standard deviation of the added Gaussian noise, up to a standard deviation of 0.06 the F1-score is higher than 0.6. We notice a clear decrease when the standard deviation reaches 0.1 due to the fact the noise becomes too important and the copied and moved areas do not look similar anymore.

Figure 14A and **Figure 14B** show the evolution of the average F1-score obtained on the entire FAU database. While **Figure 14A** illustrates this evolution as a function of the JPEG QF used during compression, **Figure 14B** illustrates this evolution as a function of the standard deviation of Gaussian noise. We can see that the average F1-scores obtained for both graphs decrease almost in a linearly way. A slightly faster decrease can be observed for the JPEG compression from a QF of 100% and between QF 30% and 20% (**Figure 14A**). The decrease is stronger in the case of the addition of Gaussian noise, **Figure 14B**.

Regarding the GRIP database, **Figure 14C** illustrates the evolution of the F1-score as a function of the JPEG QF used during compression, and **Figure 14D** illustrates this evolution as a function of the standard deviation of Gaussian noise. We can see that the average F1-scores obtained for the two graphs decrease a little more strongly than for the FAU database, although this remains relatively linear. We can observe on **Figure 14C** that for a JPEG compression up to a QF of 60%, the F1-score obtained is higher than 0.6. The decrease is stronger

in the case of the addition of Gaussian noise, as illustrated in **Figure 14D**.

In conclusion, we observe that the proposed VFCMFD method is robust to JPEG compression and Gaussian noise addition even if the results depend on the image content. Regarding scaled and rotated images, this has not been discussed in this paper because our method has not been specially developed for such attacks, but the proposed method succeeds in working with scaled images and rotated images when the angle is less than 4°. For scaled images, we obtain an F1-score always higher than 0.7, and for images rotated with an angle lower than 4°, the F1-score obtained is also higher than 0.7.

5 CONCLUSION

In this paper, we proposed a very efficient fast copy-move forgery detection (VFCMFD) method to detect copied-moved forged areas for 4K Ultra HD images.

This proposed method takes advantage of the robustness of SURF keypoints in order to find the tampered areas of an image with a high level of accuracy. The SURF detector provides for each keypoint extracted from the image its coordinates, its main orientation, computed from the gradient directions in a neighbourhood around the current keypoint, and a feature vector. Although standard keypoint-based methods are effective in detecting different types of copied moved forgeries, their matching phase usually has a quadratic complexity. Indeed, in the classical methods, each keypoint is compared to all the other keypoints extracted in the image. However, for forgeries that do not undergo a large rotation, the matched keypoints have a similar principal orientation. The method we propose does not use any threshold for the detection of the keypoints. This allows low intensity keypoint matching and a very efficient detection of copy-move forgery, even in very uniform or weakly textured areas. Consequently, without threshold the number of keypoints extracted from the image increases significantly and due to the $O(n^2)$ complexity, the computation time also increases strongly. For that reason, we propose to order all the keypoints and to perform a match search restricted to a window around the current keypoint. By comparing each keypoint only with keypoints of similar orientation, the computation time of the matching phase is significantly reduced from $\mathcal{O}(n^2)$ to $\mathcal{O}(kn)$, where n is the number of keypoints and k the size of the window taken around each keypoint. Finally a clustering algorithm (DBSCAN) is used in order to build clusters out of large patches of parallel equal length lines. We chose to use this algorithm because of its ability to reject lines instead of classifying everything. Eventually, clusters define convex hulls that we expand in order to best fit the limits of the tampered areas from which we compute the binary mask. Experimental results show that our proposed approach is very efficient in terms of F1-score and allows us to drastically reduce the computation time. Our method detects forgeries very quickly in 4K images, unlike more conventional methods.

In future work, we will investigate differentiating copied areas from moved ones in the same image. Noise analysis on the expanded convex hulls could help us make that difference. Furthermore, we want to improve this method to make it robust to various geometric attacks such as scaling or rotation.

DATA AVAILABILITY STATEMENT

The original contributions presented in the study are included in the article/Supplementary Material, further inquiries can be directed to the corresponding author.

AUTHOR CONTRIBUTIONS

CN and WP conceived the presented idea. LB and CN performed the computations. CN, LB, and WP analyzed the obtained experimental results. WP encouraged LB and CN to investigated and supervised the finding of this work. All authors discussed the results and contributed to the final manuscript.

ACKNOWLEDGMENTS

We would like to thank the financial support of the ANR-16-DEFA-0001 OEIL (statistiques rObustEs pour l'apprentIssage Léger) research project of the French ANR/DGA challenge DEFALS (DEtection de FALSifications dans des images).

REFERENCES

Aloraini, M., Sharifzadeh, M., and Schonfeld, D. (2021). Sequential and Patch Analyses for Object Removal Video Forgery Detection and Localization. *IEEE Trans. Circuits Syst. Video Technol.* 31, 917–930. doi:10.1109/tcsvt.2020.2993004

Amerini, I., Ballan, L., Caldelli, R., Del Bimbo, A., and Serra, G. (2010). "Geometric Tampering Estimation by Means of a Sift-Based Forensic Analysis," in 2010 IEEE International Conference on Acoustics, Speech and Signal Processing, Dallas, TX, March 14–19, 2010, 1702–1705. doi:10.1109/icassp.2010.5495485

Amerini, I., Ballan, L., Caldelli, R., Del Bimbo, A., and Serra, G. (2011). A SIFT-Based Forensic Method for Copy-Move Attack Detection and Transformation Recovery. *IEEE Trans.Inform.Forensic Secur.* 6, 1099–1110. doi:10.1109/TIFS.2011.2129512

Amerini, I., Becarelli, R., Caldelli, R., and Del Mastio, A. (2014). "Splicing Forgeries Localization through the Use of First Digit Features," in 2014 IEEE International Workshop on Information Forensics and Security (WIFS), Atlanta, GA, December 3–5, 2010, 143–148. doi:10.1109/wifs.2014.7084318

Ardizzone, E., Bruno, A., and Mazzola, G. (2015). Copy-Move Forgery Detection by Matching Triangles of Keypoints. *IEEE Trans.Inform.Forensic Secur.* 10, 2084–2094. doi:10.1109/tifs.2015.2445742

Bay, H., Tuytelaars, T., and Van Gool, L. (2006). "Surf: Speeded up Robust Features," in European Conference on Computer Vision, Graz, Austria, May 7–13, 2006, 404–417. doi:10.1007/11744023_32

Bayram, S., Taha Sencar, H., and Memon, N. (2009). "An Efficient and Robust Method for Detecting Copy-Move Forgery," in 2009 IEEE International Conference on Acoustics, Speech and Signal Processing, Taipei, Taiwan, April 19–24, 2009, 1053–1056. doi:10.1109/icassp.2009.4959768

Bravo-Solorio, S., and Nandi, A. K. (2009). "Passive Forensic Method for Detecting Duplicated Regions Affected by Reflection, Rotation and Scaling," in 2009 17th European Signal Processing Conference, Glasgow, UK, August 24–28, 2009, 824–828.

Chen, S., Tan, S., Li, B., and Huang, J. (2016). Automatic Detection of Object-Based Forgery in Advanced Video. *IEEE Trans. Circuits Syst. Video Technol.* 26, 2138–2151. doi:10.1109/tcsvt.2015.2473436

Chen, H., Yang, X., and Lyu, Y. (2020). Copy-move Forgery Detection Based on Keypoint Clustering and Similar Neighborhood Search Algorithm. *IEEE Access* 8, 36863–36875. doi:10.1109/access.2020.2974804

Chi-Man Pun, C.-M., Xiao-Chen Yuan, X.-C., and Xiu-Li Bi, X.-L. (2015). Image Forgery Detection Using Adaptive Oversegmentation and Feature Point Matching. *IEEE Trans.Inform.Forensic Secur.* 10, 1705–1716. doi:10.1109/tifs.2015.2423261

Christlein, V., Riess, C., Jordan, J., Riess, C., and Angelopoulou, E. (2012). An Evaluation of Popular Copy-Move Forgery Detection Approaches. *IEEE Trans.Inform.Forensic Secur.* 7, 1841–1854. doi:10.1109/tifs.2012.2218597

Cox, I. J., Miller, M. L., Bloom, J. A., and Honsinger, C. (2002). Digital Watermarking. *The Morgan Kaufmann Series in Multimedia Information and Systems*. Computer Science Technical Reports : Dartmouth College.

Cozzolino, D., Poggi, G., and Verdoliva, L. (2015a). Efficient Dense-Field Copy-Move Forgery Detection. *IEEE Trans.Inform.Forensic Secur.* 10, 2284–2297. doi:10.1109/tifs.2015.2455334

Cozzolino, D., Poggi, G., and Verdoliva, L. (2015b). "Splicebuster: A New Blind Image Splicing Detector," in 2015 IEEE International Workshop on Information Forensics and Security (WIFS), Rome, Italy, November 16–19, 2015, 1–6. doi:10.1109/wifs.2015.7368565

D'Amiano, L., Cozzolino, D., Poggi, G., and Verdoliva, L. (2019). A Patchmatch-Based Dense-Field Algorithm for Video Copy–Move Detection and Localization. *IEEE Trans. Circuits Syst. Video Technol.* 29, 669–682. doi:10.1109/TCSVT.2018.2804768

De Carvalho, T. J., Riess, C., Angelopoulou, E., Pedrini, H., and de Rezende Rocha, A. (2013). Exposing Digital Image Forgeries by Illumination Color Classification. *IEEE Trans.Inform.Forensic Secur.* 8, 1182–1194. doi:10.1109/tifs.2013.2265677

Diwan, A., Mall, V., Roy, A., and Mitra, S. (2019). "Detection and Localization of Copy-Move Tampering Using Features of Locality Preserving Projection," in 2019 Fifth International Conference on Image Information Processing (ICIIP), Shimla, India, November 15–17, 2019, 397–402. doi:10.1109/iciip47207.2019.8985823

Ester, M., Kriegel, H.-P., Sander, J., and Xu, J. (1996). "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," in KDD'96: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, Portland, Oregon, August 2–4, 1996, 226–231.

Fischler, M. A., and Bolles, R. C. (1981). Random Sample Consensus: a Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM* 24, 381–395. doi:10.1145/358669.358692

Fridrich, A. J., Soukal, B. D., and Lukáš, A. J. (2003). "Detection of Copy-Move Forgery in Digital Images," in Digital Forensics Research Conference (DFRWS), Cleveland, OH, August 2–4, 2003.

Gupta, M., and Singh, P. (2021). "An Image Forensic Technique Based on Sift Descriptors and Flann Based Matching," in 2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT), Kharagpur, India, July 6–8, 2021, 1–7. doi:10.1109/icccnt51525.2021.9579701

Hou, J.-U., and Lee, H.-K. (2017). Detection of Hue Modification Using Photo Response Nonuniformity. *IEEE Trans. Circuits Syst. Video Technol.* 27, 1826–1832. doi:10.1109/tcsvt.2016.2539828

Hu, J., Zhang, H., Gao, Q., and Huang, H. (2011). "An Improved Lexicographical Sort Algorithm of Copy-Move Forgery Detection," in 2011 Second International Conference on Networking and Distributed Computing ICNDC, Beijing, China, September 21–24, 2011, 23–27. doi:10.1109/icndc.2011.12

Huang, H., Guo, W., and Zhang, Y. (2008). "Detection of Copy-Move Forgery in Digital Images Using Sift Algorithm," in In 2008 IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application, Wuhan, China, December 19–20, 2008, 272–276. doi:10.1109/paciia.2008.2402

Kakar, P., and Sudha, N. (2012). Exposing Postprocessed Copy-Paste Forgeries through Transform-Invariant Features. *IEEE Trans.Inform.Forensic Secur.* 7, 1018–1028. doi:10.1109/tifs.2012.2188390

Kang, X., and Wei, S. (2008). "Identifying Tampered Regions Using Singular Value Decomposition in Digital Image Forensics," in 2008 International Conference on Computer Science and Software Engineering, Wuhan, China, December 12–14, 2008, 926–930. doi:10.1109/csse.2008.8763

Li, Y., and Zhou, J. (2019). Fast and Effective Image Copy-Move Forgery Detection via Hierarchical Feature Point Matching. *IEEE Trans.Inform.Forensic Secur.* 14, 1307–1322. doi:10.1109/tifs.2018.2876837

Li, G., Wu, Q., Tu, D., and Sun, S. (2007). "A Sorted Neighborhood Approach for Detecting Duplicated Regions in Image Forgeries Based on Dwt and Svd," in 2007 IEEE International Conference on Multimedia and Expo, Beijing, China, July 2–5, 2007, 1750–1753. doi:10.1109/icme.2007.4285009

Li, B., Shi, Y., and Huang, J. (2008). "Detecting Doubly Compressed Jpeg Images by Using Mode Based First Digit Features," in 2008 IEEE 10th Workshop on Multimedia Signal Processing, Cairns, QLD, October 8–10, 2008, 730–735. doi:10.1109/mmsp.2008.4665171

Li, L., Li, S., Zhu, H., Chu, S.-C., Roddick, J. F., and Pan, J.-S. (2013). An Efficient Scheme for Detecting Copy-Move Forged Images by Local Binary Patterns. *J. Inf. Hiding Multimedia Signal Process.* 4, 46–56.

Li, J., Xiaolong Li, X., Bin Yang, B., and Xingming Sun, X. (2015). Segmentation-based Image Copy-Move Forgery Detection Scheme. *IEEE Trans.Inform.Forensic Secur.* 10, 507–518. doi:10.1109/tifs.2014.2381872

Lin, G.-S., Chang, M.-K., and Chen, Y.-L. (2011). A Passive-Blind Forgery Detection Scheme Based on Content-Adaptive Quantization Table Estimation. *IEEE Trans. Circuits Syst. Video Technol.* 21, 421–434. doi:10.1109/tcsvt.2011.2125370

Liu, G., Wang, J., Lian, S., and Wang, Z. (2011). A Passive Image Authentication Scheme for Detecting Region-Duplication Forgery with Rotation. *J. Netw. Comput. Appl.* 34, 1557–1565. doi:10.1016/j.jnca.2010.09.001

Lowe, D. G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vis.* 60, 91–110. doi:10.1023/b:visi.0000029664.99615.94

Luo, W., Huang, J., and Qiu, G. (2006). "Robust Detection of Region-Duplication Forgery in Digital Image," in 18th International Conference on Pattern Recognition (ICPR'06)18th ICPR'06, Hong Kong, August 20–24, 2006, 746–749. doi:10.1109/icpr.2006.10034

Lynch, G., Shih, F. Y., and Liao, H.-Y. M. (2013). An Efficient Expanding Block Algorithm for Image Copy-Move Forgery Detection. *Inf. Sci.* 239, 253–265. doi:10.1016/j.ins.2013.03.028

Lyu, Q., Luo, J., Liu, K., Yin, X., Liu, J., and Lu, W. (2021). Copy Move Forgery Detection Based on Double Matching. *J. Vis. Commun. Image Represent.* 76, 103057. doi:10.1016/j.jvcir.2021.103057

Mahdian, B., and Saic, S. (2007). Detection of Copy-Move Forgery Using a Method Based on Blur Moment Invariants. *Forensic Sci. Int.* 171, 180–189. doi:10.1016/j.forsciint.2006.11.002

Meena, K. B., and Tyagi, V. (2020). A Hybrid Copy-Move Image Forgery Detection Technique Based on Fourier-Mellin and Scale Invariant Feature Transforms. *Multimed. Tools Appl.* 79, 8197–8212. doi:10.1007/s11042-019-08343-0

Mei, F., Lyu, J., and Lyu, Y. (2019). "Copy-move Forgery Detection Based on Interest Point and Local Search Algorithm," in 2019 IEEE 6th International Conference on Cloud Computing and Intelligence Systems (CCIS), Singapore, December 19–21, 2019, 207–211. doi:10.1109/ccis48116.2019.9073728

Muzaffer, G., Ulutas, G., and Ustubioglu, B. (2020). "Copy Move Forgery Detection with Quadtree Decomposition Segmentation," in 2020 43rd International Conference on Telecommunications and Signal Processing (TSP), Milan, Italy, July 7–9, 2020, 208–211. doi:10.1109/tsp49548.2020.9163516

Myna, A., Venkateshmurthy, M., and Patil, C. (2007). "Detection of Region Duplication Forgery in Digital Images Using Wavelets and Log-Polar Mapping," in International Conference on Computational Intelligence and Multimedia Applications (ICCIMA 2007), Sivakasi, India, December 13–15, 2007, 371–377. doi:10.1109/iccima.2007.2713

Pan, X., and Lyu, S. (2010). Region Duplication Detection Using Image Feature Matching. *IEEE Trans.Inform.Forensic Secur.* 5, 857–867. doi:10.1109/tifs.2010.2078506

Pomari, T., Ruppert, G., Rezende, E., Rocha, A., and Carvalho, T. (2018). "Image Splicing Detection through Illumination Inconsistencies and Deep Learning," in 2018 25th IEEE International Conference on Image Processing (ICIP), Athens, Greece, October 7–10, 2018, 3788–3792. doi:10.1109/icip.2018.8451227

Popescu, A. C., and Farid, H. (2004). *Exposing Digital Forgeries by Detecting Duplicated Image Regions*. Computer Science Technical Reports : Dartmouth College.

Rao, Y., and Ni, J. (2016). "A Deep Learning Approach to Detection of Splicing and Copy-Move Forgeries in Images," in 2016 IEEE International Workshop on Information Forensics and Security (WIFS), Abu Dhabi, United Arab Emirates, December 4–7, 2016, 1–6. doi:10.1109/wifs.2016.7823911

Ryu, S.-J., Lee, M.-J., and Lee, H.-K. (2010). "Detection of Copy-Rotate-Move Forgery Using Zernike Moments," in International Workshop on Information Hiding, Calgary, AB, June 28–30, 2010, 51–65. doi:10.1007/978-3-642-16435-4_5

Shivakumar, B., and Baboo, S. S. (2011). Detection of Region Duplication Forgery in Digital Images Using Surf. *Int. J. Comput. Sci. Issues* 8, 199.

Silva, E., Carvalho, T., Ferreira, A., and Rocha, A. (2015). Going Deeper into Copy-Move Forgery Detection: Exploring Image Telltales via Multi-Scale Analysis and Voting Processes. *J. Vis. Commun. Image Represent.* 29, 16–32. doi:10.1016/j.jvcir.2015.01.016

Tahaoglu, G., Uluas, G., and Ustubioglu, B. (2021). "A New Approach for Localization of Copy-Move Forgery in Digital Images," in 2021 44th International Conference on Telecommunications and Signal Processing (TSP), Brno, Czechia, July 26–28, 2021, 183–186. doi:10.1109/tsp52935.2021.9522680

Wang, J., Liu, G., Li, H., Dai, Y., and Wang, Z. (2009). "Detection of Image Region Duplication Forgery Using Model with Circle Block," in 2009 International Conference on Multimedia Information Networking and Security, Wuhan, China, November 18–20, 2009, 25–29. doi:10.1109/mines.2009.1421

Wang, C., Zhang, Z., Li, Q., and Zhou, X. (2019). An Image Copy-Move Forgery Detection Method Based on Surf and Pcet. *IEEE Access* 7, 170032–170047. doi:10.1109/access.2019.2955308

Zandi, M., Mahmoudi-Aznaveh, A., and Talebpour, A. (2016). Iterative Copy-Move Forgery Detection Based on a New Interest Point Detector. *IEEE Trans.Inform.Forensic Secur.* 11, 2499–2512. doi:10.1109/tifs.2016.2585118

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.