Improved DSIFT Descriptor based Copy-Rotate-Move Forgery Detection

Ali Retha Hasoon Khayeat[1,2], Xianfang Sun[1], Paul L. Rosin[1]

[1] School of Computer Science & Informatices, Cardiff University, UK
KhayeatAR@Cardiff.ac.uk, SunX2@Cardiff.ac.uk, RosinPL@Cardiff.ac.uk
[2] Computer Science Department, College of Science, Kerbala University, Iraq
aliretha@gmail.com

Abstract. In recent years, there has been a dramatic increase in the number of images captured by users. This is due to the wide availability of digital cameras and mobile phones which are able to capture and transmit images. Simultaneously, image-editing applications have become more usable, and a casual user can easily improve the quality of an image or change its content. The most common type of image modification is cloning, or copy-move forgery (CMF), which is easy to implement and difficult to detect. In most cases, it is hard to detect CMF with the naked eye and many possible manipulations (attacks) can be used to make the doctored image more realistic. In CMF, the forger copies part(s) of the image and pastes them back into the same image. One possible transformation is rotation, where an object is copied, rotated and pasted. Rotation-invariant features need to be used to detect Copy-Rotate-Move (CRM) forgery. In this paper we presented three contributions. First, a new technique to detect CMF is developed, using Dense Scale-Invariant Feature Transform (DSIFT). Second, a new improved DSIFT descriptor is implemented which is more robust to rotation than Zernike moments. Third, a new method to remove false matching is proposed. Extensive experiments have been conducted to train, evaluate and test the algorithms, the new feature vector and the suggested method to remove false matching. We show that the proposed method can detect forgery in images with blurring, brightness change, colour reduction, JPEG compression, variations in contrast and added noise.

Keywords: Copy-Move Forgery, Copy-Rotate-Move, DSIFT Descriptor, Zernike Moments.

1 Introduction

Copy-move is the most common image manipulation (copy and paste), where regions of the image are cloned to hide/cover objects in the scene. If this is done with care, visual detection of cloning will be difficult. Moreover, because the cloned regions can be in any location or can have any shape, searching all possible image portions in different sizes and locations is computationally infeasible.

In Copy-Move Forgery (CMF), part(s) of the image are copied and pasted into the same image but in different places, possibly after a rotation. Moreover, because the copied-pasted region is from the same image, its characteristics (e.g. colour and noise) are compatible with that image. This type of forgery is more challenging to detect than other types, such as splicing and retouching. This is because the usual methods of detecting incompatibilities, using statistical measurements to compare different parts of the image, will be useless for CMF detection [1]. The first method to detect CMF was suggested by Fridrich et al. [**?**]. They divided the image into overlapping blocks and quantised the discrete cosine transform (DCT) coefficients of each block; they then sorted them lexicographically and checked the similarity between adjacent blocks.

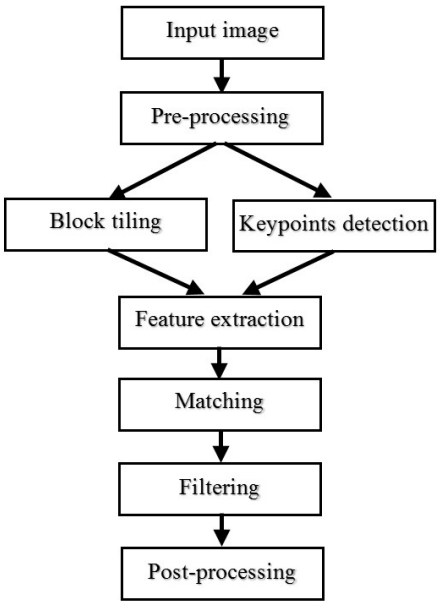

Fig. 1. The general block diagram of CMF detection.

A large number of CMF detection methods have been proposed; most of them follow a common pipeline, as shown in Fig 1. The general CMF detection system consists of several main steps. The first step is to pre-process the image, for example, by converting the RGB colour image to a greyscale image.

The second step is to extract features from the image. There are two different methods of carrying out this extraction, by dividing the image into blocks (densely) or by detecting interest points in the image (sparsely).

In the first method, the image can be divided into overlapping or non-overlapping blocks, and these blocks can have a square or circular shape. The features are extracted from the blocks. In the second, the numbers and the locations of the interest points vary and depend on the method itself (e.g. SIFT,

MSER, SURF, HOG, etc.). The features are then extracted in the neighbourhood of the interest points.

The third step is to find the matches (similarity) between extracted features. Many possible methods can be used to locate this similarity. The most common methods either 1/ sort the features vectors lexicographically and compute the Euclidean distance between adjacent stored blocks or 2/ build a kd-tree to all feature vectors and find the 2nd Approximate Nearest Neighbour (2ANN) for each feature. We tested both methods in our work.

In the final step, the false matches should be removed to refine the primary result.

Many methods of detecting CMF have been suggested. Christlein et al. [2] tested the 15 most prominent feature sets by creating a real-world copy-move dataset and a software framework for systematic image manipulation. They analysed the performance of the detection on a per-pixel basis and per-image basis. According to their experiments, SIFT and SURF keypoint-based features work very well, as well as block-based DCT, DWT, kernel PCA, PCA and Zernike moments.

Keypoint-based methods have the advantage of low computational complexity. There is a big difference in computational cost and amount of detected details in block-based methods and keypoint-based methods.

According to Christlein et al. [2] the Zernike moments achieved the most precise detection results (state of the art). Therefore, we compared our improved DSIFT with Zernike moments to determine the performance of our improved DSIFT descriptor versus the state of art feature.

One of our contributions is the combination of ideas from the keypoint and block-based methods. We chose the Scale Invariant Feature Transform (SIFT) method and applied it densely to enable block-based matching. SIFT is the most widely used descriptor; it is distinctive and relatively fast. However, in some cases, the high dimensionality of the descriptor is considered a drawback during the matching step [3].

2 Related works

Several papers have used SIFT to detect CMF, but as far as we are aware, none have used DSIFT [4] for this purpose. Huang et al. [3] extract SIFT keypoints from the image, and store them in a kd-tree to enable efficient retrieval of the 2nd Nearest Neighbours (2NN). In their work, they used images from the internet. This method can partially detect CMF (one clone only), but there was no consideration for post processing methods and they did not report the accuracy of their method.

Pan and Lyu [5], detected sparse SIFT keypoints, and used the best-bin-first algorithm followed by RANdom Sample Consensus (RANSAC)[6] to estimate the possible geometric transformations. They built a correlation coefficient map between pixels in the same region and applied Gaussian filtering (7×7) to reduce noise and threshold their results. They used their own forged images and their

method failed to detect forgery in some images with translation. Moreover, they falsely detected forgery in some original (untampered) images.

Amerini et al. [7] extracted SIFT features and used 2ANN to find multiple matches between feature vectors. They applied hierarchical clustering to their matched points and used RANSAC to estimate the geometric transform. The authors employed the Columbia photographic image repository [8] and other personal collected images . This method can partially detect multiple cloned regions, but it missed some objects and falsely detected forgery in some cases. Moreover, the authors did not consider forged images with rotation in their work.

Ryu et al. [9] divided the image into overlapping 24×24 blocks and calculated the Zernike moments for each block. They sorted the Zernike feature vectors lexicographically and computed the Euclidean distance between adjacent stored blocks. If the distance is smaller than a specific threshold, they consider these blocks as cloned. They conducted their experiments with 12 TIFF images from their personal collection and other papers. They considered the Copy-Rotate Move (CRM) with rotations in the range of 0° to 90° in 10° steps. In their follow-up paper [10], they computed the 5th order Zernike moments from overlapping block to generate their feature vectors. Locality sensitive hashing with Euclidean distance was used to find similar feature vectors. The authors applied RANSAC at the feature level to remove false matches. Then, they tested their work on their forged images, which were rotated between 0° to 90°, in steps of 10°. They built their forged images by duplicating random square patches, with different sizes, on original images. This makes CMF/CRM detection much easier and produces unrealistic forged images. Moreover, their method generates a considerable rate of Pixel False Positive (PFP) values.

Li et al. [11] followed a different approach. They segmented the image into more than 100 patches, extracted the SIFT features from the whole image and found possible matches using a kd-tree and KNN. They estimated the transformation matrix using RANSAC. Then, they refined their results using an EM-based algorithm. They tested their work on two datasets and were unable to detect forgery in some plain forged images. Moreover, they identified some unforged images as counterfeit.

3 Forgery detection algorithms

This section first describes the steps taken to improve the DSIFT descriptor, and then presents our suggested algorithm for CMF detection using our improved DSIFT descriptor. Subsequently, three different methods of removing false matches are discussed, and we propose an algorithm to remove false matches using neighbourhood clustering within a radius.

3.1 Steps to improve DSIFT

CRM forgery detection requires a rotation-invariant descriptor; thus, we improved the DSIFT descriptor to make it rotation invariant. Based on local image properties, SIFT assigns a dominant orientation for each keypoint. The keypoint

descriptor rotates each patch according to this orientation so that the subsequent descriptor is robust to rotation [12].

We improved the DSIFT descriptor in two steps; first, we used a different method to compute the dominant orientation, and second, we used circular blocks instead of square ones.

Step one: SIFT uses the following approach to detect dominant orientation for each patch. For each keypoint, we compute the gradient orientations in its 16×16 neighbourhood. Build orientation histogram has 36 bins covering 360°. Each value added to the histogram is weighted by its gradient magnitude. The peak in the orientation histogram represents the dominant directions of the keypoint. The standard setting of SIFT uses 36 bins, which causes a quantisation of the estimated dominant orientation, and this error in the orientation will cause problems in the CMF stage. It is possible to increase the number of bins to 360, but this would substantially increase the run time. Obviously, there is a trade-off between quantisation error and robustness. Therefore, we used the following method to detect the dominant orientation in our work.

In our suggested method to improve the detection of dominant orientation, we used the second order and the third order central moments to detect the dominant orientation. This method is more accurate and faster than the SIFT's method for detecting dominant orientation. The second order central moment (moment of inertia) can be used to detect the principle axes of the patch, the region around keypoint. The angle of the principle axis of the least inertia is used to describe the object orientation. This angle has a 180° ambiguity; the third central moment (projection skewness) was used to solve this ambiguity. The rotation of an object by 180° changes the sign of the projection's skewness on either axis. In other words, the sign of $\mu 30$ was used to differentiate between the possible orientations [13]. This method works very well and is much faster than the SIFT method (see section 4.3).

Step two: SIFT considers a square region around the keypoint, which increases the border effects on this region. Simply, we considered a circular area instead of square area to reduce the border effects. Each block within a radius of 8 is divided into 4×4 sub-regions. A comparison between circular and square neighbourhoods will be described in section 4.3.

The steps to build our improved DSIFT descriptor are as follows: 1/Transform a colour image into greyscale. 2/At each pixel, consider its 16×16 neighbourhood. 3/Mask each neighbourhood to use only the central disk with a radius equal to 8. 4/Use the moments based method to find the dominant orientation for each circular patch. 5/Rotate each circular patch according to its dominant orientation. 6/Compute the gradient magnitude and orientation for each circular patch. 7/Use the Gaussian function to weight the gradient magnitude. 8/For each 4×4 sub-region in the patch, build an 8 bin histogram. 9/Accumulate each bin according to its gradient magnitude of orientation. 10/Concatenate the 16 histograms to build a 128 feature vector.

3.2 CMF Detection with translation/rotation

Computing DSIFT for a 512×512 image with block size of 16×16 generates 247009 feature vectors. Computing sparse SIFT for the same image size typically generates about 750 to 1350 keypoints/feature vectors. Using SIFT densely increases the running time but provides robust features which are systematically distributed over the whole image.

The full algorithm for CMFD with translation/rotation

```
Input: IRGB % Coloured image
Output: IForgery % Forged image
IG=RGB2Grey(RGB); % Convert coloured image to greyscale image
(M,N)=size(IG);
K=0;
For i=1:M-15
 For j=N-15
  Iij=IG(i:i+15,j:j+15);
  If MAD(Iij)> T1 % Compute Median Absolute Deviation
   K=K+1;
   B(K)=DSIFT(Iij);
  end if
 end for
end for
Tkd=KDtree(B);
For L1=1:K
 V1=B(L1);
 Index=ANN2(V1);
 V2=B(Index);
 If ||V1-V2|| < T2
  List=(V1,V2);
 end if
End for
NCList=Neighbourhood_Clustering (List); % Reduce the false matches
RList=RANSAC(NCList);
IForgery=IRGB.GreenColor(RList);
CC=connected_components_labelling(RList)
For F=1: size(CC)
 If CC(F).area < T3
 IForgery.CC(F).Pixel=IRGB.CC(F).Pixel;%Restore original image colour
 end if
End for
```

Flat regions increase false matches. Such flat regions occur where the pixel intensity values are similar to each other and change smoothly over comparatively large regions (e.g. sky and sea). The similarity between pixel intensity values in a large region produces a large number of similar feature vectors, which are

considered as copy-move regions in the matching step. We used the median absolute deviation (MAD) to reduce the effect of flat regions. We checked the block's MAD value and if it was larger than a threshold, we build the descriptor to those tested block; otherwise, we neglected it. The proposed method reduces the number of the false matches in the flat region(s) and decreases the run time significantly. We considered Neighbourhood Clustering with CRM forgery detection only to decrease the false matches and reduce the number of outliers. Without Neighbourhood Clustering, RANSAC cannot efficiently estimate the transformation because of the large number of outliers. In comparison, the number of outliers is relatively small in translation.

3.3 False match removal

We tested the three following methods to remove potential false matches:

1. Counting shift vectors: This method involved creating a list of coordinates for each potential cloned patch and sorting it. Then, the shift vector (spatial distance) between each related point was computed. If the number of each of the shift vectors was greater than the threshold, the patches were considered to be a forgery. This method is appropriate in the case of translation but not for CRM forgery detection.

2. Neighbourhood Clustering: The copied and pasted blocks each had to comprise at least three neighbouring blocks. This method produced very good result; details of Neighbourhood Clustering are given in section 3.4.

3. RANSAC: This is an iterative method of estimating parameters of a mathematical model from a set of observed data which contains outliers. In the initial stage, RANSAC uses a dataset which is as small as possible; it enlarges this dataset consistently when possible. RANSAC can robustly estimate the geometric transformation between matched points and remove outlier blocks [6]. It can cope with more than 50% outliers, making it more robust than many other parameter estimation techniques (such as the least median of squares) [14]. Figure 3 is an example of using RANSAC for CMF detection to remove false matching.

3.4 Neighbourhood clustering

As the second contribution of this paper, we propose a new method to remove false matches by analysing a potential match's neighbourhood. We extensively experimented and optimised all thresholds and parameters (see Fig. 2).

The full algorithm for Neighbourhood Clustering within a radius

```
Input:A={A1,A2,..,An},B={B1,B2,..,Bn};%A & B are lists of coordinates
                                      where Bi is a potential match to Ai.
Output:AB={Am1,Bm1,Am2,Bm2,...Amq,Bmq};
% q is the number of matching pairs.
[Tkd,Ind]= KDtree(A);
```

```
For i=1:n
 Aa=Tkd(i)=A(Ind(i));
 ai=ANNk(Aa) % Find k Approximate
              Nearest Neighbour{ai1,ai2,..aik} ∈ A
 Count=0;
 For j=1: k
  e1=ai(j);
  d1=||Aa-e1|| ;
  if d1 < r % r = specific Radius
   e2= match(B,e1) % Find the matching coordinates of e1 in B.
   Bb=B(Ind(i));
   d2=||Bb-e2||;
    If d2 <= r
     count=count+1;
    End if
   End if
  End for j
  If Count > T
   Add {Aa,Bb} into AB.
  End if
End for i
```

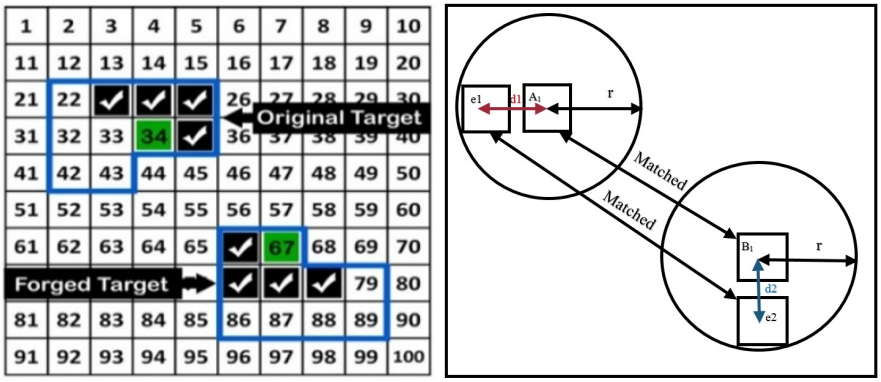

Fig. 2. An example of matching two blocks using the Neighbourhood Clustering method (left), a block diagram of two matched blocks using Neighbourhood Clustering within a radius (right).

4 Experiments and evaluation method

4.1 Datasets and evaluation method

We tested our method using the image database for Copy-Move Forgery Detection (CoMoFoD) [15]. CoMoFoD consists of 260 forged images categorised into

two categories (small 512×512, and large 3000×2000). The small category consists of 200 original images with different types of forgery. We considered only the small images in our work. In the small category, images are divided into 5 different groups according to the applied manipulation, as follows: translation, rotation, scaling, distortion and a combination of all previous manipulations. Moreover, different types of post-processing methods (e.g. blurring, brightness change, colour reduction, JPEG compression, contrast adjustments and added noise), are applied to all forged and original images in each group. The total number of images in the small group is 10400 images with different types of manipulations.

As an example of alternative datasets, CASIA [?] has realistic forgery images which are categorised according to their contents into 9 categories (scene, animal, architecture, character, plant, article, nature indoor and texture). However, it contains limited post-processing methods, specifically JPEG compression and blurring and the majority of the images are small (384×256). The MICC-F2000, MICC-F220, MICC-F8multi and MICC-F600 datasets [7] contain 2000, 220, 8 and 600 images, respectively. These datasets have unrealistic forged images and in most cases, the forgery is perceptually very obvious.

We used the F measure at the pixel level to evaluate the accuracy of our results.

4.2 Experiments and results for CMF/CRM forgery detection

A. Experiment to compute the F measure with plain CMF Detection (translation)

We tested 40 different images with plain CMF and could detect forgery in all images, but some false detections were incurred (see Fig. 3). To remove the false matching, we tested three different methods; RANSAC produced the best results with a very short run time, as shown in Table (1).

Table 1. The results of experiments in translation

Post-processing Method to Remove False Matching	F Measure	Running Time
Without Post-processing	0.8764	155 sec
Shift Vector	0.8792	45 min
Neighbourhood Clustering	0.9123	6.4 min
RANSAC	0.9367	170 sec

B. Experiments to compute the F measure with CMF Detection (translation) and attacks

To create more realistic CMF images and to hide the traces of forgery, the forger could use some post-processing methods. In our work, we considered different types of attack (image blurring, brightness change, colour reduction, JPEG compression, contrast adjustments and added noise). We used our suggested method to test 200 images with different types of post-processing. Then, we tested the

Fig. 3. From the left: The forged input image, forgery detection with false matches, the result of RANSAC, the generated masks.

same images using Zernike moments. In most cases, our improved DSIFT produced better results than Zernike moments, our method detected the forgery in 198 images out of 200 (see Table 2). However, the Zernike moments is more robust to JPEG compression than our method. This is because the DCT operation in JPEG compression has a strong influence on the gradient magnitude, which effected our improved DSIFT.

Table 2. Comparison between improved DSIFT and Zernike moments for CMF detection with different types of attacks

Attacks	F Measure using improved DSIFT	Detected images with improved DSIFT	F Measure using Zernike moments	Detected images with Zernike moments
Image Blurring, (5×5 average filter)	0.7894	38	0.5632	34
Brightness Change Range (0.01, 0.8)	0.8739	40	0.4544	33
Colour Reduction (32 intensity levels)	0.9045	40	0.4999	33
JPEG Compression (quality factor= 40)	0.3819	40	0.6078	33
Contrast Adjustment Range (0.01,0.8)	0.9033	40	0.4868	32

Then, we used our improved DSIFT to detect CMF with different levels of added noise. Then, we used Zernike moments to detect CMF with the same noisy images. We achieved satisfactory results and we get better results than Zernike moments, see Table 3.

C. Comparison between improved DSIFT, Zernike moments and original DSIFT

Christlein et al. [3] tested the 15 most prominent features to detect CMF, and Zernike moments achieved the most precise detection results. Therefore, we chose to compare Zernike moments with our method and tested our improved DSIFT descriptor. We conducted experiments on 40 different CoMoFoD images with

Fig. 4. Top row: blurred image(left), detected forgery (centre), generated mask (right). Bottom row: contrast-adjusted image(left), detected forgery (centre), generated mask (right).

Table 3. Comparison between improved DSIFT and Zernike moments on detection CMF with different levels of noise

The value of White Gaussian Noise (AWGN)	F Measure using improved DSIFT	Detected images with improved DSIFT	F Measure using Zernike moments	Detected images with Zernike moments
0.000001	0.6846	36	0.6362	36
0.000005	0.5424	32	0.5148	31
0.00001	0.4883	32	0.4188	26

CMF (translation). In the first experiment, we tested our algorithm on translation, both with and without RANSAC post-processing. In both cases, our method achieved higher F measure values than Zernike moments (see Table 4). Moreover, the results of using our improved DSIFT and original DSIFT, with translation, were similar.

Table 4. Comparison between improved DSIFT and Zernike moments in CMF

	The average of F Measure using Improved DSIFT	The average of F Measure using Zernike moments
Without post-processing	0.8764	0.8070
With post-processing	0.9367	0.8865

Another experiment was conducted with 40 different images with CRM forgery. These forged images had object(s) rotated by different angles (e.g. 180°, 90°, 10°, 2°, 4°, -4°, 5°, -7°, -3°, 1°...etc.). In this experiment, we removed false positives in two steps. In the first step, for each potential forgery blocks, we tested the 8 neighbouring blocks and if we found that at least 3 neighbouring blocks were matched, we forwarded these blocks to the next step. In the second step, we used

RANSAC to remove the outliers, which improved the results (see Fig. 5). This experiment illustrated the robustness of our descriptor. We obtained a higher F measure value than we did from Zernike moments and the original DSIFT (see Table 5). Therefore, our suggested improved DSIFT descriptor is more accurate under changes in rotation than Zernike moments and the original DSIFT.

Table 5. Comparison between improved DSIFT, Zernike moment and original DSIFT in 40 images with CRM forgery

	The average of F Measure using Improved DSIFT	**The average of F Measure using Zernike moments**	**The average of F Measure using Original DSIFT**
Without post-processing	0.4005	0.2899	0.3268
With post-processing	0.7613	0.6398	0.5436

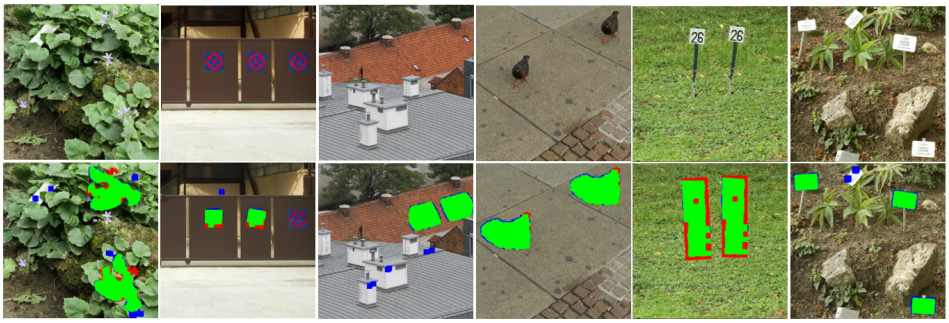

Fig. 5. Examples of using our proposed method for Detection CRM Forgery[3]

4.3 An experiment to test rotation invariant of improved DSIFT

In section 3.1, we described how the level of rotational invariance of the DSIFT descriptor was improved. We also conducted an experiment to test our descriptor.

For 40 different forgery images, we randomly selected 100 blocks from each image and computed our improved DSIFT descriptors for these blocks. Next, we randomly rotated these blocks, considering all possible rotation angles (0°-360°), and computed our improved DSIFT descriptors for these rotated blocks. Then, we computed the Euclidean distance between the descriptors of the original and rotated blocks. The average pairs of Euclidean distance between 4000 improved DSIFT descriptors, built from 4000 different blocks before and after rotation, was 0.0487 (see Fig. 6a).

To compare our improved DSIFT rotation robustness with the original DSIFT, we repeated the previous experiment using the original DSIFT. The average

[3] True Positive (TP),True Negative (TN), False Positive (FP), False Negative (FN)

pairs of Euclidean distance between 4000 descriptors (square block), built from 4000 different blocks before and after rotation, was 0.8787(see Fig. 6b). We then repeated the same experiment using the original DSIFT with circular blocks instead of square ones, and the value of the Euclidean distance was 0.3396(see Fig. 6c).

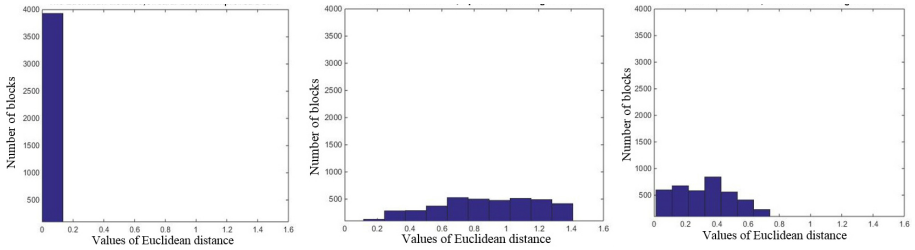

Fig. 6. Histogram of the Euclidean distance between 4000 DSIFT descriptors in: (a) the improved DSIFT (left), (b) the original DSIFT with square blocks (centre), (c) the original DSIFT with circular blocks (right).

4.4 An experiment to find the difference between matching points algorithms in CMF Detection

From previous works, we found that there are two major methods suggested to find similar blocks in CMF Detection: The first method is sorting the feature vectors lexicographically and computing the dissimilarity value between blocks (the Euclidean distance). The second approach is building the kd-tree and finding the 2ANN. We tested both methods and found them to be similar for translation, but the first method failed with rotation and we could not detect forgery with it. To understand the reason for the failure of the first method, we carried out an experiment: We computed the descriptors for two cloned blocks and saved them. Then, we built the descriptors of the whole image, sorted them lexicographically and searched for the two saved descriptors. If the lexicographic sorting worked properly with our method, the two saved descriptors would be adjacent. We found that there were 189 descriptors between the two saved descriptors. The reason for this is that the lexicographical sorting is in a column-wise manner, like a dictionary, so obviously it cannot be used to detect forgery with rotation.

5 Conclusion

In this paper, we considered copy-move forgery incorporating translation and rotation . A new technique was suggested to detect CMF/CRM Forgery. We obtained excellent results on translation and very good results on rotation. We improved the accuracy of the rotation robustness of DSIFT; thus, we achieved better results than for Zernike moment in rotation. A new method of removing false matching was developed and extensively tested.

Acknowledgement This research was supported by the Higher Committee for Education Development (HCED)in Iraqi and the Cardiff University School of Computer Science and Informatics.

References

1. Baboo, S.: Automated Forensic Method for Copy-Move Forgery Detection based on Harris Interest Points and SIFT Descriptors. International Journal of Computer Applications **27** (2011) 9–17
2. Christlein, V., Riess, C., Jordan, J., Riess, C., Angelopoulou, E.: An evaluation of popular copy-move forgery detection approaches. IEEE Transactions on Information Forensics and Security **7** (2012) 1841–1854
3. Huang, H., Guo, W., Zhang, Y.: Detection of Copy-Move Forgery in Digital Images Using SIFT Algorithm. In: 2008 IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application. Volume 2., IEEE (2008) 272–276
4. Dalal, N., Triggs, B.: Histograms of Oriented Gradients for Human Detection. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). Volume 1., IEEE (2005) 886–893
5. Pan, X., Lyu, S.: Region duplication detection using image feature matching. In: IEEE Transactions on Information Forensics and Security. Volume 5. (2010) 857–867
6. Fischler, M.a., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM **24** (1981) 381–395
7. Amerini, I., Ballan, L., Caldelli, R., Del Bimbo, A., Serra, G.: A SIFT-based forensic method for copy-move attack detection and transformation recovery. IEEE Transactions on Information Forensics and Security **6** (2011) 1099–1110
8. Ng, T.t., Chang, S.f., Hsu, J., Pepeljugoski, M.: Columbia Photographic Images and Photorealistic Computer Graphics Dataset. Columbia University, ADVENT Tecnical Report # 205-2004-5 (2005) 1–23
9. Ryu, S.J., Lee, M.J., Lee, H.K.: Detection of Copy-Rotate-Move Forgery Using Zernike Moments. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Volume 6387 LNCS. (2010) 51–65
10. Ryu, S.J., Kirchner, M., Lee, M.J., Lee, H.K.: Rotation invariant localization of duplicated image regions based on zernike moments. IEEE Transactions on Information Forensics and Security **8** (2013) 1355–1370
11. Li, J., Li, X., Yang, B., Sun, X.: Segmentation-Based Image Copy-Move Forgery Detection Scheme. IEEE Transactions on Information Forensics and Security **10** (2015) 507–518
12. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision **60** (2004) 91–110
13. Prokop, R.J., Reeves, A.P.: A survey of moment-based techniques for unoccluded object representation and recognition. In: CVGIP: Graphical Models and Image Processing. Volume 54. (1992) 438–460
14. Zuliani, M.: RANSAC for Dummies With examples using the RANSAC toolbox for Matlab^{TM} & Octave and more (2010). (I Edizione)
15. Tralic, D., Zupancic, I., Grgic, S., Grgic, M.: CoMoFoD - New Database for Copy-Move Forgery Detection. In: Proceedings of 55th International Symposium ELMAR-2013. Number September, IEEE (2013) 25–27