Comparison of Matching Methods for Copy-Move Image Forgery Detection

Osamah M. Al-Qershi and Bee Ee Khoo

Abstract Copy-Move is one of the most common image forgery types, where a region of an image is copied and pasted into another location of the same image. Such a forgery is simple to achieve but hard to be detected as the pasted region shares the same characteristics with the image. Although plenty of algorithms have been proposed to tackle the copy-move detection problem, those algorithms differ in two things; matching method and type of features. In this paper, we focus on analyzing and comparing four matching methods in terms of accuracy and robustness against different image processing operations. Such analysis and comparison provide indispensable information for the design of new accurate and reliable copy-move detection techniques.

Keywords Copy-move · Digital image forensics · Image forgery

1 Introduction

Digital images became an important source of information in our digital world. It has been said that a picture is worth a thousand words and seeing is believing. But those sayings seem not to be completely acceptable in the presence of photo editing software. Popular and simple computer software can be used by average computer users to manipulate digital images without leaving a noticeable trace. Although manipulated, or forged, images can be shared using social media for fun, they can be used in many serious situations such as journalism, criminal investigation, and surveillance systems [1]. Copy-move is one of the most popular methods for manipulating a semantics image [2].

O.M. Al-Qershi (✉) · B.E. Khoo (✉)
School of Electrical and Electronic Engineering,
Universiti Sains Malaysia, Penang, Malaysia
e-mail: osamahqershi.lm07@student.usm.my

B.E. Khoo
e-mail: beekhoo@usm.my

H. Ibrahim et al. (eds.), *9th International Conference on Robotic, Vision, Signal Processing and Power Applications*, Lecture Notes in Electrical Engineering 398,
DOI 10.1007/978-981-10-1721-6_23

It can be achieved by copying a region from an image and pasting it into the same image with the intent of hiding undesired objects or replicating objects. In copy-move forgery, the tampered region still shares most of its inherent characteristics, such as the color palette or pattern noise, with the remainder of the image. Most of the copy-move detection algorithms adhere to a common pipeline [3]. First, the image is optionally pre-processed (downscaling and/or conversion to greyscale). It is then subdivided into overlapping blocks of pixels. From each of these blocks, a feature vector is extracted. Highly similar feature vectors are matched as pairs. The similarity of two features can be determined by different similarity criteria, e.g. Euclidean distance. In the verification step, outliers are removed and holes are filled which may be achieved using a basic filtering such as morphological operations.

The overall performance of copy-move detection methods depends mainly on two stages of that pipeline; the type of the features that are extracted from image blocks and the matching method [3]. There are two main differences between the existing methods; the type of the features that are used for matching image blocks, and the method of matching. In spite the wide range of features which have been used for copy-detection algorithms, only a few matching methods have been exploited in those algorithms. Researchers always justify employing certain types of features that are invariant to geometrical operations or noise, but not all of them do same when they chose a matching method. The limited number of matching methods that have been used so far gives the impression that matching methods do not have the same impact on detection accuracy that features have. In this paper, we compare between four matching methods in order to study and understand the effect of matching method on the performance of copy-move detection algorithm. For fair comparison, the same feature, which is Zernike moments, is used with the four features. In other words, this paper compares between four Zernike moments-based algorithms. The paper is organized as follows. Section 2 introduces the Zernike moments and the four matching method Sect. 3 gives the setup of the experiments. Section 4 gives the results and the comparison between the four matching methods. The conclusion is drawn in Sect. 5.

2 Background

In this section we briefly introduce the methods and techniques that will be used in the comparison. For more details about those methods, the reader can refer to references that are mentioned in each subsection.

2.1 Zernike Moments

Moments and invariant functions of moments have been extensively used for invariant feature extraction in a wide range of pattern recognition, digital watermarking applications and etc. [4]. Among the various types of moments found in the literature, Zernike moments have been proved to be superior to the others in terms of their insensitivity to image noise, information content, and ability to provide faithful image representation [4, 5].

2.2 Matching Based on Lexicographical Sort

This method is the most popular matching method because it is simple, efficient, and straightforward [6]. In this matching method, the set of feature vectors,Z, is sorted lexicographically which is similar to dictionary sort. The sorted set is denoted as $\hat{Z}$. From the set $\hat{Z}$, the Euclidean distance between adjacent pairs of $\hat{Z}$ is calculated. If the distance is smaller than the pre-defined threshold D_1, we consider the inquired blocks as a pair of candidates for the forgery. Due to the fact that the neighboring blocks might result in relatively similar Zernike moments, the distance between the actual blocks corresponding to the pair of vectors is calculated. If the calculated distance is greater than a pre-defined threshold D_2, the corresponding blocks are considered as copy-move blocks. To enhance the performance of the matching process, each vector is compared with the next r vectors.

2.3 Matching Based on Lexicographic Sort and Grouping

The reason behind proposing this method is the fact that vectors corresponding to similar blocks are not always adjacent to each other after lexicographical sorting. It means that matching a vector with the next r vectors may be not enough to find similar vectors, and this may reduce the true positive ratio (TPR) [7]. To overcome that issue, a grouping method introduced in [8]. Instead of matching all vectors with each other, the vectors are first divided evenly into G groups. Then G buckets are created so that the i bucket contains the vectors from group i, group $i - 1$, group $i + 1$. Each vector will be placed into 3 buckets except the vectors in the first and last groups which are placed in only two buckets. The vectors are matched with all vectors within the same bucket. The matching starts with sorting Z using lexicographical sort. Then the resultant $\hat{Z}$ is divided into G groups and G buckets are created. Within each bucket B, vectors are paired, and the actual distance between paired blocks is calculated as D_A. A new set of paired vectors is created as:

$$\mathbf{P}_i = \left\{(\mathbf{B}_{i_j}, \mathbf{B}_{i_k})\right\}, j \neq k \, \forall i = 1 \ldots G \tag{1}$$

$$\text{If } D_A(\mathbf{B}_{i_j}, \mathbf{B}_{i_k}) > D_1 \tag{2}$$

Within each set P_i, the relative error is calculated between vectors of each pair as the ratio of the absolute error and the minimum value of the two components. If all the relative errors are below threshold D_1, the two corresponding blocks are considered as candidate forgeries. Otherwise, the pair of vectors is omitted from P_i.

2.4 Matching Based on k − d Tree

Bentley introduced the $k - d$ tree as a binary tree that stores k-dimensional data. Beside the quite efficient in its storage requirements, a significant advantage of this structure is that a single data structure can handle many types of queries very efficiently [9]. The $k - d$ tree preprocesses data into a data structure that allows making efficient range queries. It stores points of a k-dimensional space in the leaves. In order to overcome the drawbacks of straightforward lexicographic sorting, which is said to be too sensitive to the transformations and yields a lower false positive rate, researcher adopted $k - d$ tree [10]. Compared to lexicographical sorting, $k - d$ tree produces reliable results and lower false negative rates. In addition, researchers utilized $k - d$ tree to reduce the computational cost [11].

2.5 Matching Based on Locality Sensitive Hashing

Locality-sensitive hashing (LSH), proposed by Indyk and Motwani [12], is an approximate similarity search technique that works efficiently even for high-dimensional data. It has gained some popularity for copy-move detection [13, 14], as it is more robust to image processing and can be still quite fast. The function of LSH is to solve the (r, ϵ)-NN similarity search problem in sub-linear time. If, for a point q (query) in d-dimensional space, there exists an indexed point p such that $(p, q)r$, then LSH will, with high probability, return an indexed point p' such that $d(p', q)(1 + \epsilon)r$. If no indexed point lies within $(1 + \epsilon)r$ of q, then LSH will return nothing with high probability. This is achieved by means of a set of special hash functions. The hash functions satisfy the intuitive notion that the probability of a hash collision for two points be related to the similarity (distance) between the points. LSH reduces the false negatives rate by using multiple hash functions in parallel.

3 Experimental Setup

3.1 Image Dataset Preparation

We started preparing the dataset with 20 BMP authentic images from a personal collection. The images were selected carefully such as they have relatively similar regions to make copy-move detection quite challenging which simulates real situations. The size of the images is 400×300 pixels, and the size of the copied region is 80×80, which is about 5 % of the image size. The copied regions are pasted in different locations for each image in order to provide kind of spatial synchronization and homogeneity between the copied region and its neighbors. Copy-move forgery with various combinations of manipulations such as scaling, rotation, JPEG compression, Additive White Gaussian Noise AWGN, blurring were performed.

The final dataset has 900 images. Table 1 shows the settings used to generate those images. The level of blur in the Table 2 is equal to the radius of the disk-shape filter used to generate the blur. The level of Gaussian noise in the table is used to calculate the variance of the Gaussian Noise according to the following equation $v = 25 \times 10^{l-5}$, where v is the variance and l is the value in the table. For Multiple operations images, only one level of blur and noise was used because higher levels affect the overall quality of the images dramatically.

Table 1 The settings used for preparing the dataset

		Levels							
No processing		–	–	–	–	–	–	–	–
Single operation	Scaling (percentage)	80 %	90 %	110 %	120 %	–	–	–	–
	Rotating (angle)	15°	45°	90°	135°	–	–	–	–
	Blur (radius)	2	3	4	5	–	–	–	–
	Gaussian Noise (variance)	0	1	2	3	–	–	–	–
	JPG compression (quality)	100 %	90 %	80 %	70 %	–	–	–	–
Multiple operations	Scaling	90 %	90 %	110 %	110 %	90 %	90 %	110 %	110 %
	Rotating	15°	45°	15°	45°	15°	45°	15°	45°
	JPG comp.	100 %	100 %	100 %	100 %	70 %	70 %	70 %	70 %
	Scaling	90 %	90 %	110 %	110 %	90 %	90 %	110 %	110 %
	Rotating	15°	45°	15°	45°	15°	45°	15°	45°
	Blur	2	2	2	2	2	2	2	2
	JPG comp.	100 %	100 %	100 %	100 %	70 %	70 %	70 %	70 %
	Scaling	90 %	90 %	110 %	110 %	90 %	90 %	110 %	110 %
	Rotating	15°	45°	15°	45°	15°	45°	15°	45°
	Noise	0	0	0	0	0	0	0	0
	JPG comp.	100 %	100 %	100 %	100 %	70 %	70 %	70 %	70 %

Table 2 Parameters and thresholds used for evaluation

	LEX sort	LEX sort with grouping	$k-d$ tree	LSH
Block size	16	16	16	16
$N_{moments}$	12	12	12	12
D_1	50	0.25	–	–
D_2	32	32	–	–
D_3	2	20	4	11
r	7	–	–	–
G	–	256	–	–
Bucket size	–	–	50	
Table Length	–	–	–	20
Hashing key size	–	–	–	24

3.2 Metrics

In this paper, we adopted Precision (P), Recall (R), and F_1-measure which are often-used measures in the field of information retrieval [15]. The Precision is a measure for the probability that a detected region is correct, while the Recall is the ratio of True Positive components to elements inherently ranked as the positive class. They can be calculated as follows:

$$P = \frac{True\ positive}{True\ positive + False\ positive} \tag{3}$$

$$R = \frac{True\ positive}{True\ positive + False\ negtive} \tag{4}$$

Due to the trade-off between Precision and Recall, the accuracy of detection is measured using $F_{1-measure}$, which is the harmonic-mean of Precision P and Recall R:

$$F_{1-measure} = \frac{2PR}{P+R} \tag{5}$$

3.3 Verification Step

To refine the matching results, usually a verification step is used. For fair comparison, the same simple verification step is used with four matching methods. The verification step is based on a histogram that counts the number of matching blocks separated by the same distance. To do so, the shift vector s between the two matching blocks is calculated. At the end of the matching process, the counter C indicates the frequencies with which different shift vectors occur. Finally, a threshold D_3 is used to group shift vectors corresponding to regions that are located within the same distance if:

$$s_1, s_2 < D_3 \tag{6}$$

3.4 Experimental Setup

Several experiments have been carried out in order to setup the initial values and the thresholds required for all methods. The initial values the thresholds are in Table 2.

4 Experimental Results and Discussion

The dataset which we prepared includes different images with various operations are applied on the copied region to create the copy-move forgery. Table 3 illustrates a comparison between the four methods when no processing is involved in creating the copy-move forgery and the overall accuracy (F_1) for the 900 images. For the other types processing, Fig. 1 shows comparisons between the four methods for each group of processing.

The LSH-based method has the highest accuracy among the methods that we investigate as shown in Table 3. The straightforward lexicographical method has the lowest accuracy due to the high False Positive Ratio (FPR). However, adding some scaling to blocks before pasting may affect the accuracy of detection dramatically especially for the two methods which involve lexicographical sort. The accuracies of LSH and $k-d$ tree methods seem to be the almost the same and invariant to small percentage of scaling (110 %). When rotation is used to create the forgery, the accuracy of detection decreases in proportion to the angle of rotation for all four methods. Nevertheless, the LSH-based method is almost invariant to rotation with small angle (15°). When the copied region is blurred before pasting, the four methods show different patterns of performance.
LSH-based method shows high accuracy when higher levels of blur, 4 and 5, is involved. Its accuracy is even higher than what it achieves when no processing is involved to create the forgery, 82.66 compared to 72.23. The two lexicographical sort-based methods show the same performance but with lowest level of blur. The

Table 3 A comparison between the four methods ($F_1 \times 100$)

	LEX sort	LEX sort with grouping	$k-d$ tree	LSH
No processing	74.46	70.02	82.61	77.23
Over all	34.88	48.93	57.88	67.38

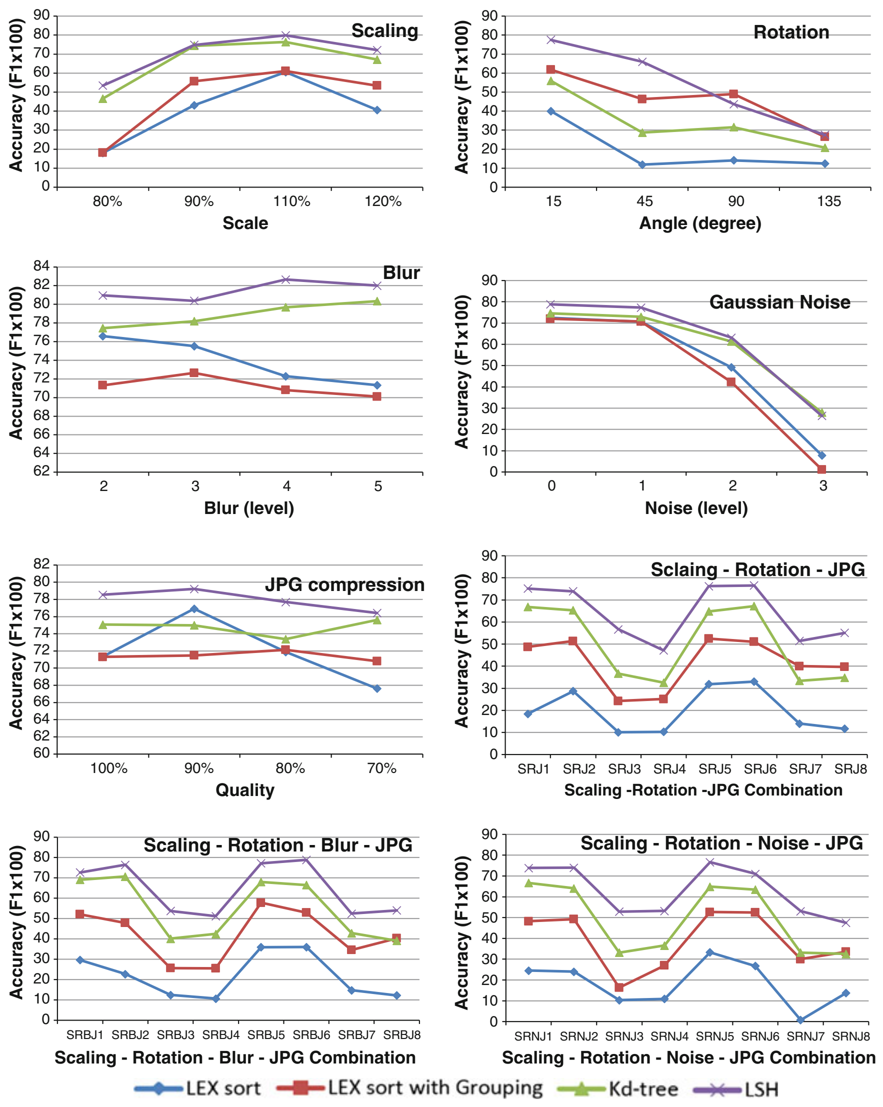

Fig. 1 Accuracy of detection of the four methods after applying different operations

accuracy of the k-d tree-based method increases gradually in proportion to the level of blur, but the accuracy is less than what it achieves when no processing is involved. We can conclude from such an unusual performance that adding some levels of blur, low levels in some cases and high levels in other cases, may enhance the accuracy of copy-move detection.

Gaussian noise has the highest impact on detection accuracy due to the distortion that it brings to image. That distortion makes four methods have almost the same accuracy. However, the four methods can achieve quite high accuracy, 72.03 − 78.80, when low level of noise is used during creating the forgery. The LSH-based and $k-d$ tree-based methods are not significantly affected by JPG compression.

When multiple types of operations are exploited to create the forgery, the differences of the four methods become very clear. Again, the LSH-based method has the highest detection accuracy. The plots of the three combinations; scaling-rotation-JPG, scaling-rotation-blur-JPG, and scaling-rotation-noise-JPG have the same pattern. That is because of the high impact of scaling. For example, in scaling-rotation-JPG combination, there is not much difference between SRJ1 and SRJ2 as the scaling ratio is the same. But, there is a big difference between SRJ2 and SRJ3 as the scaling ratios are different, 90 and 110 % respectively. The same thing can be concluded form the plots of the other two combinations.

5 Conclusion

In this paper, we compared between four matching method that have been employed in copy-move forgery detection. For fair comparison, the same features and same verification step have been used in the experiments. The experimental results shows different responses of the four methods based on the type of operations involved in creating the copy-move forgery. Moreover, the four methods showed a wide range of overall accuracy, 34.88–67.38, measured by F_1 measure. We can conclude that matching method has a significant impact on the accuracy of detection of copy-move forgery.

For future work, we are going to compare between the four methods using different types of features in order to study the effect of the matching method along with the type of features. Such a comparison may reveal any possible interaction between the two factors, which may help researcher in selecting the proper matching method for certain types of features.

Acknowledgments The authors would like to acknowledge the financial assistance provided by Ministry of Education Malaysia through FRGS grant number 203/PELECT/6071305.

References

1. Mahdian B, Saic S (2010) A bibliography on blind methods for identifying image forgery. Signal Process: Image Commun 25:389–399
2. Redi JA, Taktak W, Dugelay JL (2011) Digital image forensics: a booklet for beginners. Multim Tools Appl 51:133–162
3. Al-Qershi OM, Khoo BE (2013) Passive detection of copy-move forgery in digital images: state-of-the-art. Forensic Sci Int 231:284–295
4. Kim HS, Lee HK (2003) Invariant image watermark using zernike moments. IEEE Trans Circuits Syst Video Technol 13:766–775

5. Teh CH, Chin RT (1988) On image analysis by the methods of moments. IEEE Trans Pattern Anal Mach Intell 10:496–513
6. Bravo-Solorio S, Nandi AK (2011) Automated detection and localisation of duplicated regions affected by reflection, rotation and scaling in image forensics. Signal Process 91:1759–1770
7. Al-Qershi OM, Khoo BE (2014) Enhanced matching method for copy-move forgery detection by means of zernike moments. In: Digital-forensics and watermarking. Springer, pp 485–497
8. Lynch G, Shih FY, Liao HYM (2013) An efficient expanding block algorithm for image copy-move forgery detection. Inf Sci 239:253–265
9. Bentley JL (1975) Multidimensional binary search trees used for associative searching. Commun ACM 18:509–517
10. Shivakumar B, Baboo LDSS (2011) Detection of region duplication forgery in digital images using surf. IJCSI Int J Comput Sci Issues 8
11. Langille A, Gong M (2006) An efficient match-based duplication detection algorithm. In: The 3rd Canadian conference on computer and robot vision, 2006. IEEE, pp 64–64
12. Indyk P, Motwani R (1998) Approximate nearest neighbors: towards removing the curse of dimensionality. In: Proceedings of the thirtieth annual ACM symposium on theory of computing, ACM, pp 604–613
13. Ryu SJ, Kirchner M, Lee MJ, Lee HK (2013) Rotation invariant localization of duplicated image regions based on zernike moments. IEEE Trans Inf Forensics Secur 8:1355–1370
14. Li Y (2013) Image copy-move forgery detection based on polar cosine transform and approximate nearest neighbor searching. Forensic Sci Int 224:59–67
15. Ryu SJ, Lee MJ, Lee HK (2010) Detection of copy-rotate-move forgery using zernike moments. In: Information hiding. Springer, pp 51–65