Multimed Tools Appl (2018) 77:15093–15110
DOI 10.1007/s11042-017-5094-y

An improved block-based matching algorithm of copy-move forgery detection

Yuecong Lai[1] · Tianqiang Huang[2,3] · Jing Lin[1] · Henan Lu[1]

Received: 6 September 2016 / Revised: 21 March 2017 / Accepted: 9 August 2017 /
Published online: 19 August 2017

Abstract Copy-move forgery is a common way of image tampering. Matching algorithm is the key step in copy-move forgery detection. Usually, the classical block-based matching algorithm (CBMA) can't find all matched sub-blocks. In this paper, we propose an improved block-based matching algorithm (IBMA) to solve the problem. Firstly, we put the sum of feature vectors in the first column to get a new matrix. Secondly, the matrix is sorted by first column. Finally, every row of the matrix will search the following rows until the difference in the first column is larger than the threshold value. Experiment results show that the improved block-based matching algorithm is better than the classical block-based matching algorithm when an image was distorted by Gaussian noise, salt-pepper noise, or JPEG compression. The reason is that improved block-based matching algorithm can look for all matched sub-blocks, which makes copy-move forgery detection methods more robust.

Keywords Digital image forensics · Copy-move forgery · Region duplication detection

1 Introduction

Today, owing to powerful computers, advanced photo-editing software packages and high resolution capturing devices, digital images can be easily manipulated and edited [1, 15].

The original version of this article was revised: Reference citations in Fig. 4 were incorrectly written as [5], [6], [7], [10] and [11]. It should be written as [14], [9], [11], [7] and [6].

✉ Huang Tianqiang
fjhtq@fjnu.edu.cn

[1] School of Mathematics and Computer Science, Fujian Normal University, Fuzhou 350007, China

[2] Faculty of Software, Fujian Normal University, Fuzhou 350007, China

[3] Fujian Engineering Research Center of Public Service Big Data Mining and Application, Fujian Normal University, Fuzhou 350007, China

(a) original image (b) forgery image

Fig. 1 An example of copy-move forgery from CASIA database

An image maybe changed inadvertently or intentionally when it spread through the Internet. Digital image forensics aims to verify their authenticity. Image authentication solution is classified into two types: active methods and passive methods [1]. In contrast with the active methods, the passive methods needn't additional information and used more widely.

There are many ways to falsify an image. Copy-move forgery is a common and no perceptible method of image tampering. As shown in Fig. 1, it means that parts of an image are copied and pasted another part of the same image [5]. As a result, some information of the image will be appended or hidden. Copy-move forgery detection is one of the passive methods.

Various methods have been proposed to detect copy-move forgery. Most of them follow a common pipeline [4], as shown in Fig. 2. Copy-move forgery detection methods are categorized either keypoint-based methods or block-based methods [4], which have their respective pros and cons. For feature extraction, block-based methods used different feature vectors to represent the sub-blocks, like images' brightness [14], Polar Sine Transform (PST) [9], Zernike moments [11], Discrete Cosine Transform (DCT) [2] and so on. However, for matching, the block-based matching algorithm often fixed (e.g., [2, 9–11, 14]). The step of algorithm as follows:

Step1. Lexicographically sort the matrix that is formed by feature vectors.

Step2. Calculate the Euclidean distance between adjacent feature vectors in the matrix. If the results are less than a preset threshold value V_{eur}, two sub-blocks represented by the adjacent feature vectors will classify the matching sub-blocks.

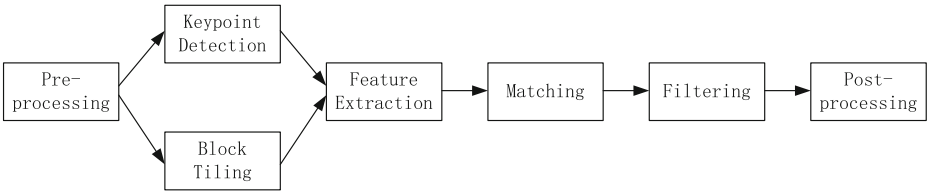

Fig. 2 General pipeline of copy-move detection methods [4]

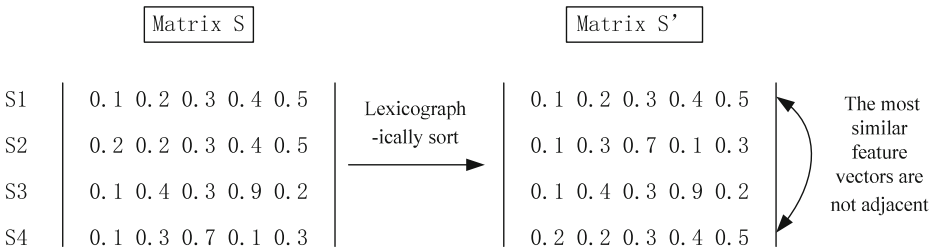

Fig. 3 An example of classical block-based matching algorithm

We call above-mentioned algorithm the classical block-based matching algorithm (CBMA). Figure 3 is an example of CBMA. S is the matrix that is formed by feature vectors. After lexicographically sorting matrix S, we can get S'. For $S1$ (the first row of matrix S), we respectively compute the Euclidean distance between $S1$ and $S2$, $S1$ and S3, $S1$ and S4. The Euclidean distance between $S1$ and $S2$ is the smallest. But $S1$ and $S2$ isn't adjoining in the matrix S'. As a result, CBMA will most probably miss many matching sub-blocks.

In order to solve this problem, we devise an improved block-based matching algorithm (IBMA). IBMA can pick out all matched sub-blocks. Either CBMA or IBMA is one step of copy-move forgery detection methods. Compared with CBMA, IBMA is able to detect more matched sub-blocks. Hence, IBMA let copy-move forgery detection methods more robust. Besides, the threshold V_{eur} is very important, because it determines whether two sub-blocks are matched. We put forward a scheme to get the threshold V_{eur}.

2 Methods

Flow diagrams of the proposed algorithm can be seen from Fig. 4. First, if the tested image is a RGB color image, we can turn it to grayscale. Second, the input image is divided into sub-blocks. Third, each sub-block will become a feature vector. The means of feature extraction are brightness [14], PST [9], Zernike moments [11], DCT [7] and Hu moments [6]. Fourth, we respectively apply CBMA and IBMA to matching. Finally, the filtering method proposed by [8] is used to remove wrong matched sub-blocks. The whole algorithm aims to compare the different results between CBMA and IBMA.

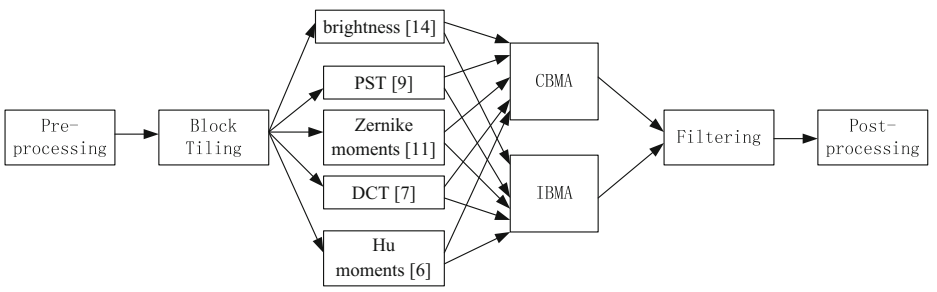

Fig. 4 The algorithm framework of our method

2.1 Feature extraction

If the input image is a RGB color image, we can turn it to grayscale by (1).

$$I = 0.228R + 0.587G + 0.114B \tag{1}$$

We assume the size of tested image is $M \times N$ pixels. Then we divide it into overlapped sub-blocks of $b \times b$ pixels. The number of sub-blocks is N_{all}.

$$N_{all} = (M - b + 1) \times (N - b + 1) \tag{2}$$

To better compare CBMA and IBMA, we take five feature vectors: brightness [14], PST [9], Zernike moments [11], DCT [7] and Hu moments [6]. We extract each feature separately rather than concatenate five different existing features. For every feature, we use CBMA and IBMA to make experiments. It must be added that the quantization table of [7] become improper because the size of sub-blocks is changed. We use the quantization table of [5], and the quantization factor is 80.

2.2 Matching

2.2.1 Classical block-based matching algorithm

The purpose of matching algorithm is to select matched sub-blocks. If a pair of sub-blocks is matching, perhaps they are the blocks in either copy region or move region. After extracting feature vectors of dimension n, we can obtain the matrix M of $N_{all} \times (n + 2)$. For the matrix M, the n-dimensional column vector at the head is the feature vectors and the 2-dimensional column vector at the end is the sub-blocks' coordinate of top left corner. In the chapter Sections 2.2.1 and 2.2.2 of this article, we define the top right corner of matrix notation represents the column of the matrix and the lower right corner represents the row of the matrix. For example, M_1^2 means the first row and the second column of matrix M.

The steps of CBMA are shown as follows. First, lexicographically sort the matrix M that is formed by feature vectors and position coordinates of sub-blocks. Then we can obtain the matrix Q. By this way, similar feature vectors will be adjacent as far as possible. Second, we calculate the Euclidean distance between adjoining two feature vectors in the matrix Q like (3). If the results are less than the threshold V_{eur}, we calculate the distance of the corresponding sub-blocks' position coordinates like (4). If the results are more than the threshold V_{dis}, the position coordinates of sub-blocks will be persisted to matrix T. The threshold V_{eur} ensures that the two sub-blocks are resembled. On account of the copy region and move region have a certain distance, we set the threshold V_{dis}.

We assume $Q1 = \left[I^1, I^2, \ldots, I^n\right]$ and $Q2 = \left[L^1, L^2, \ldots, L^n\right]$. The Euclidean distance between $Q1$ and $Q2$ can be calculated as (3).

$$SIM(Q1, Q2) = \sqrt{\sum_{r=1}^{n} (I^r - L^r)^2} \tag{3}$$

We assume $Q3 = [x, y]$ and $Q4 = \left[x', y'\right]$. The distance between $Q3$ and $Q4$ can be calculated as (4).

$$DIS(Q3, Q4) = \sqrt{(x - x')^2 + (y - y')^2} \tag{4}$$

Algorithm 1 CBMA

Input : the matrix M(the row vectors of M are formed by the feature vectors and the position coordinates of sub-blocks)
Output : the matrix T(the row vectors of T are formed by the position coordinates of matched sub-blocks)

1. $Q \leftarrow sort\ the\ matrix$ M $lexicographically$
2. j=1;
3. $for\ i = 1; i < N_{all}; i++\ do$
4. $\quad Q1 \leftarrow \left[Q_i^1, Q_i^2, \ldots, Q_i^n\right]; Q2 \leftarrow \left[Q_{(i+1)}^1, Q_{(i+1)}^2, \ldots, Q_{(i+1)}^n\right];$
5. $\quad Q3 \leftarrow \left[Q_i^{(n+1)}, Q_i^{(n+2)}\right]; Q4 \leftarrow \left[Q_{(i+1)}^{(n+1)}, Q_{(i+1)}^{(n+2)}\right];$
6. $\quad if\ (SIM\,(Q1, Q2) < V_{eur})\ \&\ \&\ (DIS\,(Q3, Q4) > V_{dis})$
7. $\quad\quad T_j \leftarrow \left[Q_i^{(n+1)}, Q_i^{(n+2)}, Q_{(i+1)}^{(n+1)}, Q_{(i+1)}^{(n+2)}\right]$
8. $\quad\quad$ % record the position coordinates of matched sub-blocks
9. $\quad\quad j++;$
10. $\quad end\ if$
11. $end\ for$

2.2.2 *Improved block-based matching algorithm*

The signs of many feature vectors (e.g., DCT [7] and Hu moments [6]) are positive and negative. We deem that only the signs of the feature vectors are the same can the sub-blocks probably become matched sub-blocks. In other words, if not all values of the feature vectors is either more than zero or less than zero, we firstly classify the matrix M based on the signs of the feature vectors. Then we continue the follow steps of IBMA. Furthermore, for each sub-block, if the number of matching is more than M_{ax}, we abandon them. We set M_{ax} to 10. Namely, if a sub-block matches lots of other sub-blocks, the sub-block is unqualified.

Now we provide the approach of IBMA. Firstly, we can count the sum of the feature vectors in the matrix M and put them in the column vector D. Then we can have a new matrix P by combining M with D. Secondly, we gain Q after sorting the matrix P based on the first column. Thirdly, every row of the matrix Q will search the following rows until the difference in the first column is larger than a threshold value $\sqrt{n} \times V_{eur}$. If the Euclidean distance (calculate by (3)) of the two rows is less than V_{eur} and the distance (calculate by (4)) is more than V_{dis}, the position coordinates of sub-blocks will keep in matrix T.

The key of IBMA is the stopping condition of searching. Without the stopping condition, every row of the matrix Q will search the following rows until the end. Moreover, the threshold $\sqrt{n} \times V_{eur}$ is based on the threshold V_{eur}.

We suppose $Q1 = \left[I^1, I^2, \ldots, I^n\right]$ and $Q2 = \left[L^1, L^2, \ldots, L^n\right]$. The threshold V_{eur} is a constant and more than zero.

Condition $\mathcal{A}$: $\left|\left(I^1 + I^2 + \ldots + I^n\right) - \left(L^1 + L^2 + \ldots + L^n\right)\right| > \sqrt{n} \times V_{eur}$.

Condition $\mathcal{B}$: $SIM\,(Q1, Q2) > V_{eur}$.

We have proved that condition $\mathcal{A}$ can deduce condition $\mathcal{B}$. The details of proof procedure are in Appendix. The proof mainly used average inequality. For two feature vectors, $Q1$ and $Q2$, if the difference value between the sum of $Q1$ and the sum of $Q2$ is more than $\sqrt{n} \times V_{eur}$, we can deduce that the Euclidean distance between $Q1$ and $Q2$ is more than V_{eur}. So the condition $\mathcal{A}$ can turn into stopping condition of searching.

Algorithm 2 IBMA

Input : the matrix M(the row vectors of M are formed by the feature vectors and the position coordinates of sub-blocks)
Output : the matrix T(the row vectors of T are formed by the position coordinates of matched sub-blocks)

1. $for\ k = 1; k < N_{all}; k++\ do$
2. $\quad D_k \leftarrow M_k^1 + M_k^2 + \ldots + M_k^n$
3. $\quad$ % get the sum of the feature vectors
4. end for
5. $P \leftarrow [D, M]$
6. % put the sum of the feature vectors in the first column of P
7. $Q \leftarrow\ sort\ the\ matrix\ P\ based\ on\ the\ first\ column$
8. j==1;
9. $for\ \ i = 1; i < N_{all}; i++\ do$
10. $\quad r \leftarrow (i+1)$;
11. $\quad while\ (1)$
12. $\quad$ % put into circulation
13. $\quad\ if\ \ \left(Q_r^1 - Q_i^1\right) < \sqrt{n} \times V_{eur}\ \ then\ do$
14. $\quad\quad Q1 \leftarrow \left[Q_i^2, Q_i^3, \ldots, Q_i^{(n+1)}\right]; Q2 \leftarrow \left[Q_r^2, Q_r^3, \ldots, Q_r^{(n+1)}\right];$
15. $\quad\quad Q3 \leftarrow \left[Q_i^{(n+2)}, Q_i^{(n+3)}\right]; Q4 \leftarrow \left[Q_r^{(n+2)}, Q_r^{(n+3)}\right];$
16. $\quad\quad if\ (SIM(Q1, Q2) < V_{eur})\ \ \&\ \ \&\ \ (DIS(Q3, Q4) > V_{dis})$
17. $\quad\quad\quad T_j \leftarrow \left[Q_i^{(n+2)}, Q_i^{(n+3)}, Q_r^{(n+2)}, Q_r^{(n+3)}\right]$
18. $\quad\quad\quad$ % record the position coordinates of matched sub-blocks
19. $\quad\quad\quad$ j++;
20. $\quad\quad\quad$ r++;
21. $\quad\quad$ else
22. $\quad\quad\quad$ r++;
23. $\quad\quad$ end if
24. $\quad$ else
25. $\quad\quad$ break;
26. $\quad$ % satisfy the stopping condition , terminate the cycle
27. $\quad$ end if
28. end for

2.2.3 The calculation method of the threshold V_{eur}

The threshold V_{eur} is important because it determines whether two sub-blocks are matched or not. The value of the threshold V_{eur} must be suitable. Most of copy-move forgery detection methods of block-based obtain the threshold V_{eur} through experience. In view of this problem, we put forward a scheme to compute the threshold V_{eur}. After a tampering image was distorted by Gaussian noise, salt-pepper noise, or JPEG compression, the corresponding sub-blocks in copy region and move region may be changed. Our method gains the threshold V_{eur} by calculating the change.

If we should detect an image database, the details of our scheme are as follows. First, 10 images are randomly picked out from the database. For each image, we copy a region (size is 70×70) of the image and paste the region to another part of the same image. At the same

time, we record the locations of the two regions. Second, we respectively add Gaussian noise (variance is 0.001), salt-pepper noise (parameter is 0.003), and JPEG compression (quality factor is 80) to the 10 images. Then we can get 30 images. Third, we take 10 images of JPEG compression as an example. For every JPEG tampered image, we randomly select 100 sub-blocks (size is $b \times b$) from the middle part of copy region. The size of the middle part is 60×60. After that, we calculate the Euclidean distance between the corresponding sub-blocks of copy region and move region. And e_i(i = 1,2,...,100) represent the results of Euclidean distance. Then we sort e_i(i = 1,2,...,100) in ascending order. The front 50 e_i(i = 1,2,...,50) is remained to compute their average value. And E_j(j = 1,2,...,10) represent the results of average values from 10 JPEG images. We can sort E_j(j = 1,2,...,10) in ascending order. The front 5 E_j(j = 1,2,...,5) is remained to get their average value. We can receive the final result V_{eur}^1 from the 10 JPEG tampered images. By the same way, we can obtain the result V_{eur}^2 of 10 tampered images with Gaussian noise and the result V_{eur}^3 of 10 tampered images with salt-pepper noise. Finally, we can get the threshold V_{eur} through reckoning the average value of V_{eur}^1, V_{eur}^2 and V_{eur}^3.

2.3 Filtering

The aim of filtering algorithm is to reduce the probability of false matches. Our method of filtering is based on the filtering algorithm of [8].

As shown in Fig. 5, for every matched sub-blocks (e.g., a and a'), we can achieve the number of adjacent matched sub-blocks (e.g., b and b', c and c'). Take a pair of matched sub-blocks, a and a', as an example. If a sub-block is located in the circular area whose center is a and the matched sub-block is situated in the circular area that the center is a', we define the pair of matched sub-blocks is an adjacent matched sub-block of a and a'. We set the radius to R_{re}. The threshold of the number of adjacent matched sub-blocks is N_{re}. For a pair of matched sub-blocks (e.g., a and a'), when the number of adjacent matched sub-blocks is bigger than N_{re}, we keep the pair of matched sub-blocks (a and a').

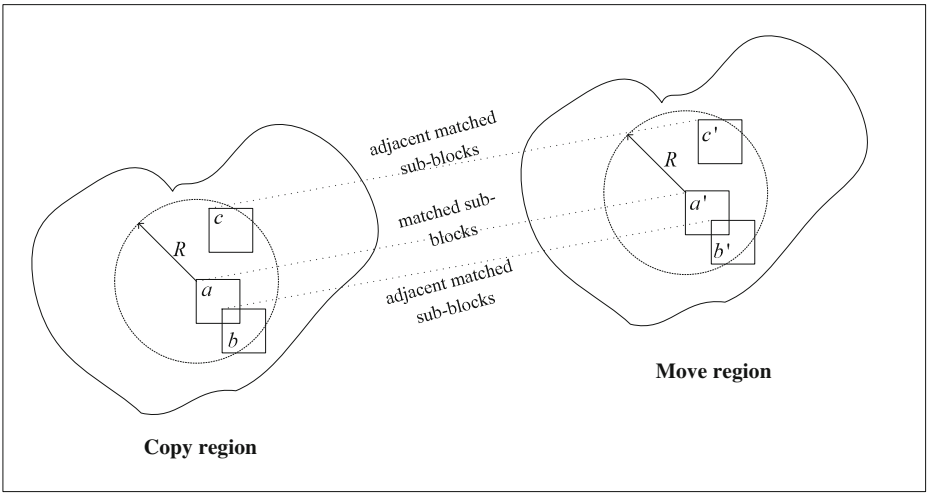

Fig. 5 Adjacent matched sub-blocks

As shown in Fig. 6, for a and a', we define angle β_k ($k = 1, \ldots, N_{re}$) as the included angle between a and the adjacent sub-block of a. The value of β_k is range from $-\pi$ to π. And we define $\beta_k{}'$ ($k = 1, \ldots, N_{re}$) as the included angle between a' and the adjacent sub-block of a'. The value of $\beta_k{}'$ is range from $-\pi$ to π. Then we define:$\theta_k = \left(\beta_k - \beta_k{}'\right) \bmod (2\pi)$ ($k = 1, \ldots, N_{re}$). Ideally, θ_k equal to the rotating angle α of move region. We calculate the variance of the assemblage $\left[\theta_1, \theta_2, \ldots, \theta_{N_{re}}\right]$. If the variance is less than φ_{re}, we remain the pair of matched sub-blocks (a and a').

In summary, for a pair of matched sub-blocks, if the number of adjacent matched sub-blocks is bigger than N_{re} and the variance of the assemblage $\left[\theta_1, \theta_2, \ldots, \theta_{N_{re}}\right]$ is less than φ_{re}, we hold it.

3 Experimental results and analysis

The experiments were performed on a computer with Intel core i5 CPU (3.20 GHz) and 4 GB RAM. The software environment of tests is Matlab R2012a. The images that we used in the experiments were formed by two parts. The one was 100 images from UCID [13]. The size of the images are 512 × 384. In real life, most people tamper with the images deliberately by Photoshop. In order to simulate the situation, we make use of Photoshop CS3 to tamper images. The other was 100 images of CASIA V2.0 [3]. The size of the images range from 384 × 256 to 528 × 318. All the images of CASIA V2.0 are formed by authentic images and tampered images. The tampered ways of CASIA V2.0 are splicing and copy-move. We randomly chose 100 images that the tampered method was copy-move.

3.1 Thresholds setting

The thresholds of our experiments were shown as Table 1. The threshold V_{eur} of UCID and CASIA V2.0 were obtained by the method of chapter Section 2.2.3.

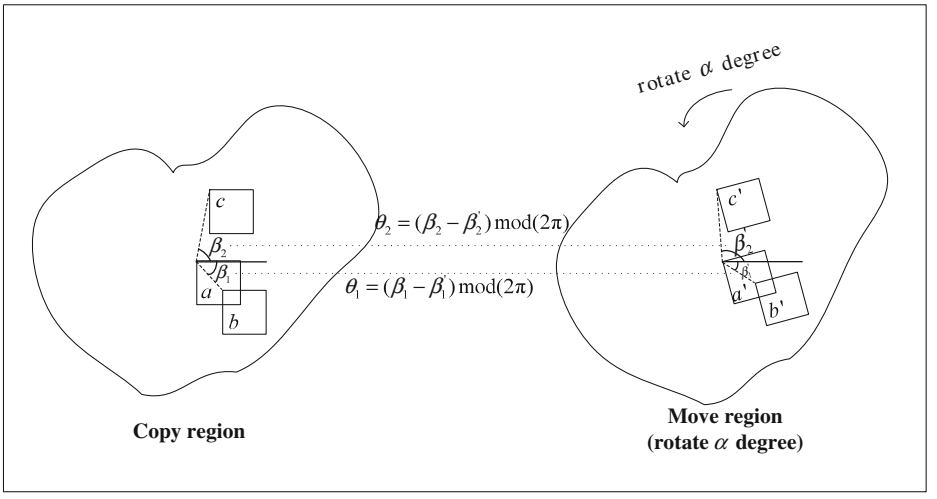

Fig. 6 The angle of adjacent matched sub-blocks

Table 1 Thresholds setting

Method	Blocksize $b \times b$	Feature length (n)	Matching		Filtering			
			UCID's V_{eur}	CASIA's V_{eur}	V_{dis}	R_{re}	N_{re}	φ_{re}
brightness [14]	25 × 25	4	0.0019	0.002	36	6	4	0.01
DCT [7]	24 × 24	8	0.93	0.9				
Hu moments [6]	24 × 24	7	0.069	0.091				
PST [9]	24 × 24	9	0.089	0.081				
Zernike moments [11]	24 × 24	12	116.9	109.5				

3.2 UCID data

For the 100 images of UCID, we randomly copied an area which size is 70 × 70 from an image and pasted it to another part of the same image. Moreover, the tampered images were added with Gaussian noise, salt-pepper noise, and JPEG compression. We detected these images by CBMA and IBMA to compare the results.

In order to display the results objectively, we adopted the True Positive Rate (TPR) and False Positive Rate (FPR) of [12].

$$TPR = \frac{|TP|}{|R_{clone}|} \tag{5}$$

$$FPR = \frac{|FP|}{|R_{nomal}|} \tag{6}$$

In (5), $|TP|$ represents the number of pixels correctly classified as tampered in the copy region and the move region. And $|R_{clone}|$ represents the number of real tampered pixels in the copy regions and the move regions. In (6), $|FP|$ represents the number of pixels wrongly classified as tampered in the normal region that isn't tampered. And $|R_{normal}|$ represents the number of real normal pixels in the region that do not belong to the copy regions or move regions. In general, the higher are the TPR and the lower are the FPR, the better are the copy-move detection methods.

Table 2 shows the tampered images' TPR and FPR of CBMA and IBMA. The tampered images without post-processing are from UCID. From Table 2, we can see that the TPR of CBMA is slightly higher than IBMA and the FPR of CBMA is slightly lower than IBMA. In short, CBMA performs slightly better than IBMA. IBMA can detect all matched sub-blocks that contain false matches and right matches. Because more false

Table 2 The TPR and FPR of UCID images without post-processing

Method	brightness [14]		DCT [7]		Hu moments [6]		PST [9]		Zernike moments [11]	
	CBMA	IBMA	CBMA	IBMA	CBMA	IBMA	CBMA	IBMA	CBMA	IBMA
TPR	0.9963	0.9310	0.9684	0.8983	0.9717	0.9619	0.9694	0.8701	0.9953	0.9216
FPR	0.0096	0.1201	0.0035	0.0082	0.0015	0.0105	0.0019	0.0347	0.0053	0.0192

Table 3 The TPR of UCID images with JPEG compression

Method \ JPEG(QF)		60	70	80	90
Brightness [14]	CBMA	0.0422	0.0393	0.0719	0.2479
	IBMA	0.8237	0.8729	0.9170	0.9252
DCT [7]	CBMA	0.0265	0.0361	0.0670	0.1833
	IBMA	0.8113	0.9093	0.9096	0.9005
Hu moments [6]	CBMA	0.0220	0.0278	0.0355	0.1078
	IBMA	0.7129	0.7830	0.8858	0.9316
PST [9]	CBMA	0.0201	0.0216	0.0241	0.0380
	IBMA	0.5766	0.7025	0.8559	0.8658
Zernike	CBMA	0.0264	0.0374	0.0521	0.1202
moments [11]	IBMA	0.8694	0.9400	0.9296	0.9319

matches are detected, a part of right matches are deleted through filtering algorithm in chapter Section 2.3.

Table 3 is the tampered images' TPR of CBMA and IBMA. The tampered images had been handled by JPEG compression. The JPEG quality factors (QF) are 60, 70, 80 and 90. When the QFs are less than 90, CBMA almost can't detect the tampered regions. But the change of QF has influenced IBMA not too much. Table 4 shows the JPEG images' FPR of CBMA and IBMA. On the whole, the FPR of CBMA is slightly lower than IBMA. If we consider the TPR, it's acceptable.

Table 5 is the tampered images' TPR of CBMA and IBMA. The tampered images were added with Gaussian noise. The variances (V_{ar}) of Gaussian noise are 0.0001, 0.0003, 0.0005 and 0.0007. With different variances, the most TPR of CBMA close to zero. However, the most TPR of IBMA are higher than 0.8. Table 6 shows the FPR of CBMA and IBMA. In general, the FPR of CBMA is slightly lower than IBMA. From the TPR and FPR, we can see that IBMA is more robust than CBMA when the tampered images are distorted by Gaussian noise.

After the tampered images were added with salt-pepper noise, we calculated their TPR of CBMA and IBMA, as shown in Table 7. The noise densities (d) of salt-pepper noise are

Table 4 The FPR of UCID images with JPEG compression

Method \ JPEG(QF)		60	70	80	90
Brightness [14]	CBMA	0.0176	0.0143	0.0065	0.0023
	IBMA	0.1296	0.1263	0.1258	0.1214
DCT [7]	CBMA	0.0061	0.0033	0.0031	0.0023
	IBMA	0.0087	0.0084	0.0086	0.0078
Hu moments [6]	CBMA	0.0070	0.0039	0.0020	0.0004
	IBMA	0.0177	0.0145	0.0132	0.0110
PST [9]	CBMA	0.0052	0.0028	0.0021	0.0011
	IBMA	0.0370	0.0383	0.0359	0.0344
Zernike	CBMA	0.0044	0.0034	0.0025	0.0021
moments [11]	IBMA	0.0182	0.0173	0.0202	0.0193

Table 5 The TPR of UCID images with Gaussian noise

Method \ Gaus.(Var)		0.0001	0.0003	0.0005	0.0007
Brightness [14]	CBMA	0.1279	0.0033	0.0013	0
	IBMA	0.9170	0.8742	0.8129	0.7544
DCT [7]	CBMA	0.0773	0.0172	0.0026	0.0011
	IBMA	0.9103	0.8915	0.7579	0.5476
Hu moments [6]	CBMA	0.0227	0.0114	0.0036	0
	IBMA	0.8796	0.7864	0.6999	0.6221
PST [9]	CBMA	0.0112	0.0049	0	0
	IBMA	0.8647	0.8811	0.8639	0.7972
Zernike	CBMA	0.0480	0.0128	0.0015	0.0017
moments [11]	IBMA	0.9367	0.9325	0.8516	0.6407

0.001, 0.002, 0.003 and 0.004. The noise densities (d) mean that the percentage of affected pixels. When the feature vectors are brightness [14], the TPR of either CBMA or IBMA are very high. For the other feature vectors, the most TPR of CBMA are close to zero with different noise densities. But the most TPR of IBMA are higher than 0.7. Table 8 shows the FPR of CBMA and IBMA. In a word, the FPR of CBMA is slightly lower than IBMA. When the tampered images are distorted by salt-pepper noise, we can see that IBMA is more robust than CBMA except for the feature vectors are brightness [14].

For an image, the running time of copy-move forgery detection methods are different when the matching algorithm is either CBMA or IBMA. As shown in Table 9, the running time of copy-move forgery detection methods using CBMA is less than using IBMA. There are two reasons to consider. First, is that the time complexity of IBMA is higher than CBMA. Second, when the matching algorithm is IBMA, the running time of filtering algorithm is more than CBMA because IBMA will detect more matched sub-blocks.

In all, although the running time of copy-move forgery detection methods using IBMA is more than using CBMA, IBMA is more robust than CBMA when the tampered images are distorted by JPEG compression Gaussian noise or salt-pepper noise.

Table 6 The FPR of UCID images with Gaussian noise

Method \ Gaus.(Var)		0.0001	0.0003	0.0005	0.0007
brightness [14]	CBMA	0.0002	0	0	0.0003
	IBMA	0.1038	0.0856	0.0726	0.0626
DCT [7]	CBMA	0.0006	0.0003	0	0
	IBMA	0.0066	0.0062	0.0043	0.0044
Hu moments [6]	CBMA	0	0	0	0
	IBMA	0.0099	0.0086	0.0076	0.0065
PST [9]	CBMA	0.0003	0.0001	0	0
	IBMA	0.0241	0.0168	0.0126	0.0090
Zernike	CBMA	0.0007	0.0002	0.0001	0.0001
moments [11]	IBMA	0.0174	0.0158	0.0134	0.0096

Table 7 The TPR of UCID images with salt-pepper noise

Method \ slat-pep.(d)		0.001	0.002	0.003	0.004
brightness [14]	CBMA	0.9829	0.9490	0.9251	0.8510
	IBMA	0.9054	0.8697	0.8518	0.7887
DCT [7]	CBMA	0.2590	0.0494	0.0164	0.0073
	IBMA	0.8709	0.7663	0.6196	0.4421
Hu moments [6]	CBMA	0.2277	0.0197	0.0014	0
	IBMA	0.8454	0.7237	0.6379	0.5358
PST [9]	CBMA	0.2743	0.0240	0.0014	0.0032
	IBMA	0.8653	0.8431	0.7783	0.6446
Zernike	CBMA	0.5176	0.1320	0.0299	0.0015
moments [11]	IBMA	0.9031	0.8515	0.7347	0.5743

3.3 CASIA data

We randomly got 100 images that the tampered way was copy-move. The sizes of the tampered regions are different and irregular. Some of the tampered images had multiple tampering regions. From CASIA V2.0, we can only extract the information of move regions and can't obtain the copy regions. What's more, the tampered images were added with Gaussian noise, salt-pepper noise, and JPEG compression. We detected these images by CBMA and IBMA to contrast their results.

Due to only have the information of move regions, we define Referenced True Positive Rate ($RTPR$).

$$RTPR = \frac{\left|TP'\right|}{\left|R'_{clone}\right|} \tag{7}$$

In (7), $\left|TP'\right|$ represents the number of pixels correctly classified as tampered in the move regions. And $\left|R'_{clone}\right|$ represents the number of real tampered pixels in the move regions. Thinking from a certain degree, the higher are the $RTPR$, the better are the copy-move detection methods.

Table 8 The FPR of UCID images with salt-pepper noise

Method \ slat-pep.(d)		0.001	0.002	0.003	0.004
brightness [14]	CBMA	0.0072	0.0053	0.0045	0.0032
	IBMA	0.1065	0.0974	0.0883	0.0817
DCT [7]	CBMA	0.0009	0.0003	0.0001	0.0001
	IBMA	0.0084	0.0073	0.0065	0.0049
Hu moments [6]	CBMA	0.0004	0.0004	0.0003	0
	IBMA	0.0079	0.0086	0.0066	0.0070
PST [9]	CBMA	0.0005	0.0003	0	0
	IBMA	0.0255	0.0186	0.0132	0.0110
Zernike	CBMA	0.0018	0.0009	0.0006	0.0002
moments [11]	IBMA	0.0187	0.0165	0.0168	0.0130

Table 9 The running time of using CBMA and IBMA

Method	Brightness [14]		DCT [7]		Hu moments [6]		PST [9]		Zernike moments [11]	
	CBMA	IBMA	CBMA	IBMA	CBMA	IBMA	CBMA	IBMA	CBMA	IBMA
Runtime(S)	4	68	24	48	35	66	8	120	8	127

Table 10 shows the tampered images' $RTPR$ of CBMA and IBMA. The tampered images without post-processing are from CASIA V2.0. From Table 10, we can see that the performances of CBMA and IBMA are almost equal. To compare with CBMA, IBMA has some advantage when the tampered regions are irregular.

Table 11 is the tampered images' $RTPR$ of CBMA and IBMA. The tampered images had been processed by JPEG compression. The JPEG quality factors (QF) are 60, 70, 80 and 90. CBMA almost can't detect the tampered regions when the QFs are less than 90. However, the change of QF has influenced IBMA not too much.

Table 12 is the tampered images' $RTPR$ of CBMA and IBMA. The tampered images from CASIA V2.0 were added with Gaussian noise. The variances (V_{ar}) of Gaussian noise are 0.0001, 0.0003, 0.0005 and 0.0007. With different variances, the most $RTPR$ of CBMA nearly equal zero. But the most TPR of IBMA are higher than 0.7. From the $RTPR$, we can think that IBMA is more robust than CBMA when the tampered images are distorted by Gaussian noise.

After the tampered images from CASIA V2.0 were added with salt-pepper noise, we calculated their $RTPR$ of CBMA and IBMA, as shown in Table 13. The noise densities (d) of salt-pepper noise are 0.001, 0.002, 0.003 and 0.004. When the feature vectors are brightness [14], the TPR of either CBMA or IBMA are very high. For the other feature vectors, CBMA almost can't detect the tampered regions when the noise densities are more than 0.001. And the change of noise densities has influenced IBMA not too much. When the tampered images are distorted by salt-pepper noise, we can prefer that IBMA is more robust than CBMA except for the feature vectors are brightness [14].

In brief, although tampered regions are different and irregular, IBMA is more robust than CBMA when the tampered images are distorted by JPEG compression, Gaussian noise or salt-pepper noise.

From all experiments of UCID and CASIA V2.0, we can get the conclusion that IBMA is more robust than CBMA. For every sub-block, we can search their all matched sub-blocks by using IBMA. CBMA only search the sub-block represented by adjacent feature vector after lexicographically sorting the feature vectors' matrix. When an image was distorted by Gaussian noise, salt-pepper noise, or JPEG compression, the matched sub-blocks probably are not adjacent in the feature vectors' matrix. This leads to CBMA become invalid.

Table 10 The $RTPR$ of CASIA images without post-processing

Method	brightness [14]		DCT [7]		Hu moments [6]		PST [9]		Zernike moments [11]	
	CBMA	IBMA	CBMA	IBMA	CBMA	IBMA	CBMA	IBMA	CBMA	IBMA
$RTPR$	0.9020	0.8753	0.7669	0.8088	0.7780	0.8398	0.7744	0.8166	0.8458	0.8382

Table 11 The $RTPR$ of CASIA images with JPEG compression

Method \ JPEG(QF)		60	70	80	90
brightness [14]	CBMA	0.0479	0.0451	0.1195	0.3785
	IBMA	0.7787	0.8052	0.8604	0.8741
DCT [7]	CBMA	0.0227	0.0331	0.1067	0.2401
	IBMA	0.6459	0.7720	0.8064	0.8093
Hu moments [6]	CBMA	0.0239	0.0257	0.0623	0.1791
	IBMA	0.6988	0.7327	0.7947	0.8301
PST [9]	CBMA	0.0194	0.0159	0.0289	0.0616
	IBMA	0.3780	0.4584	0.7733	0.8148
Zernike	CBMA	0.0272	0.0334	0.0923	0.2287
moments [11]	IBMA	0.7084	0.8208	0.8418	0.8437

Table 12 The $RTPR$ of CASIA images with Gaussian noise

Method \ Gaus.(V_{ar})		0.0001	0.0003	0.0005	0.0007
brightness [14]	CBMA	0.1600	0.0083	0.0029	0
	IBMA	0.8517	0.8138	0.7853	0.7273
DCT [7]	CBMA	0.0886	0.0102	0.0039	0.0003
	IBMA	0.8074	0.7621	0.6330	0.4591
Hu moments [6]	CBMA	0.0393	0.0145	0.0025	0.0022
	IBMA	0.7955	0.7380	0.6837	0.6495
PST [9]	CBMA	0.0219	0.0046	0.0003	0
	IBMA	0.8113	0.7996	0.7583	0.6170
Zernike	CBMA	0.0904	0.0102	0.0018	0.0003
moments [11]	IBMA	0.8500	0.8279	0.7215	0.5025

Table 13 The $RTPR$ of CASIA images with salt-pepper noise

Method \ slat-pep.(d)		0.001	0.002	0.003	0.004
brightness [14]	CBMA	0.8875	0.8587	0.8332	0.8014
	IBMA	0.8412	0.8237	0.7973	0.7741
DCT [7]	CBMA	0.2262	0.0494	0.0075	0.0024
	IBMA	0.7476	0.6493	0.4890	0.3431
Hu moments [6]	CBMA	0.1983	0.0277	0.0053	0
	IBMA	0.7573	0.6993	0.6264	0.5943
PST [9]	CBMA	0.2377	0.0284	0.0047	0.0009
	IBMA	0.7997	0.7475	0.6393	0.4751
Zernike	CBMA	0.4458	0.1383	0.0306	0.0078
moments [11]	IBMA	0.8121	0.7487	0.6151	0.4794

4 Conclusions

We have two contributions in copy-move forgery detection. Firstly, IBMA can detect all matched sub-blocks. The reason is that IBMA is based on strict mathematical proofs. In the step of matching, most existing copy-move forgery detection methods can't detect all matched sub-blocks, such as [9, 11, 14] and [2]. Secondly, the method we propose to calculate the threshold V_{eur} is reasonable. For different image databases and feature vectors, we can obtain relatively high TPR or $RTPR$.

Acknowledgments This work was supported by the National Natural Science Foundation of China (Grant No. 61070062,61502103), the Industry-University Cooperation Major Projects in Fujian Province (Grant No. 2015H6007), Fuzhou science and technology project (Grant No. 2014-G-76), the Science and Technology Department of Fujian province K-class Foundation Project (Grant No. JK2011007), and The Education Department of Fujian Province A-class Foundation Project (Grant No. JA10064).

Appendix

We assume $Q1 = \left[I^1, I^2, \ldots, I^n\right]$ and $Q2 = \left[L^1, L^2, \ldots, L^n\right]$. V_{eur} is a constant and more than zero.

Condition $\mathcal{A}$: $\left|\left(I^1 + I^2 + \ldots + I^n\right) - \left(L^1 + L^2 + \ldots + L^n\right)\right| > \sqrt{n} \times V_{eur}$.
Condition $\mathcal{B}$: $SIM(Q1, Q2) > V_{eur}$.

$$\because \left|(I^1 + I^2 + ... + I^n) - (L^1 + L^2 + ... + L^n)\right| > \sqrt{n} \times V_{eur}$$

$$\therefore \frac{\left|(I^1 - L^1) + (I^2 - L^2) + ... + (I^n - L^n)\right|}{\sqrt{n}} > V_{eur}$$

$$\therefore \frac{\left((I^1 - L^1) + (I^2 - L^2) + ... + (I^n - L^n)\right)^2}{n} > (V_{eur})^2$$

From average inequality for every real number $x_i, i = 1, 2, ..., n$ we can get: $x_1^2 + x_2^2 + ... + x_n^2 \geq \frac{(x_1 + x_2 + ... + x_n)^2}{n}$ we define: $x_i = (I^i - L^i), i = 1, 2, ..., n$, obviously:

$$(I^1 - L^1)^2 + (I^2 - L^2)^2 + ... + (I^n - L^n)^2 \geq \frac{\left((I^1 - L^1) + (I^2 - L^2) + ... + (I^n - L^n)\right)^2}{n}$$

$$\because \frac{\left((I^1 - L^1) + (I^2 - L^2) + ... + (I^n - L^n)\right)^2}{n} > (V_{eur})^2$$

$$\therefore (I^1 - L^1)^2 + (I^2 - L^2)^2 + ... + (I^n - L^n)^2 > (V_{eur})^2$$

$$\therefore \sqrt{(I^1 - L^1)^2 + (I^2 - L^2)^2 + ... + (I^n - L^n)^2} > V_{eur}$$

References

1. Al-Qershi OM, Khoo BE (2013) Passive detection of copy-move forgery in digital images: state-of-the-art. Forensic Sci Int 231(1):284–295
2. Cao Y, Gao T, Fan L, Yang Q (2012) A robust detection algorithm for copy-move forgery in digital images. Forensic Sci Int 214(1):33–43
3. CASIA tampered image detection evaluation (TIDE) database, v2.0, http://forensics.idealtest.org (2011)
4. Christlein V, Riess C, Jordan J, Riess C, Angelopoulou E (2012) An evaluation of popular copy-move forgery detection approaches. IEEE Trans Inf Forensic Secur 7(6):1841–1854
5. Fridrich J, Soukal D, Lukáarticleš J (2003) Detection of copy-move forgery in digital images. In: Proceedings of Digital Forensic Research Workshop
6. Hu MK (1962) Visual pattern recognition by moment invariants. IEEE Trans Inf Theory 8(2):179–187
7. Huang Y, Lu W, Sun W, Long D (2011) Improved DCT-based detection of copy-move forgery in images. Forensic Sci Int 206(1):178–184
8. Li Y (2013) Image copy-move forgery detection based on polar cosine transform and approximate nearest neighbor searching. Forensic Sci Int 224(1):59–67
9. Li L, Li S, Zhu H, Wu X (2014) Detecting copy-move forgery under affine transforms for image forensics. Comput Electr Eng 40(6):1951–1962
10. Liu G, Wang J, Lian S, Wang Z (2011) A passive image authentication scheme for detecting region-duplication forgery with rotation. Jnetw Comput Appl 34(5):1557–1565
11. Ryu SJ, Lee MJ, Lee HK (2010) Detection of copy-rotate-move forgery using Zernike moments. In: Information Hiding, pp 51–65
12. Silva E, Carvalho T, Ferreira A, Rocha A (2015) Going deeper into copy-move forgery detection: exploring image telltales via multi-scale analysis and voting processes. J Vis Commun Image R 29:16–32
13. Uncompressed Colour Image Dataset. http://homepages.lboro.ac.uk/cogs/datasets/ucid/ucid.html (2003)
14. Wang J, Liu G, Li H, Dai Y, Wang Z (2009) Detection of image region duplication forgery using model with circle block. In: International Conference on Multimedia Information Networking and Security, pp 25–29
15. Zhou L, Chao HC, Vasilakos AV (2011) Joint forensics-scheduling strategy for delay-sensitive multimedia applications over heterogeneous networks. IEEE J Sel Area Comm 29(7):1358–1367

Yuecong Lai was born in 1991 in Ganzhou, Jiangxi, China. He received his bachelor degree of science in information and computing sciences in 2013 from Henan University of engineering, China. He received his Master's degree of science in Software Engineering in 2016 from Fujian normal university, China.His research efforts are mainly focused on image processing, multimedia forensics and data mining.

Tianqiang Huang was born in 1971 in Putian, Fujian, China. He received the Ph.D. degree in computer application from Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2006. He is currently a full professor at Faculty of Software, Fujian Normal University, China. His research interests include data mining, image processing and digital forensics.

Jing Lin was born in 1992 in Putian, Fujian, China. She received her bachelor of engineering in computer science and technology in 2014 from Fujian Normal University, China. She is currently pursuing a Master's degree in Fujian normal university and her research efforts are mainly focused on data mining, image processing, multimedia forensics.

Henan Lu was born in 1988 in Liuan, Anhui, China. He received his bachelor degree of engineering in computer science and technology in 2012 from Huainan Normal University, China. He received his Master's degree of science in 2016 from Fujian normal university, China.His research efforts are mainly focused on multimedia forensics and data mining.