SEGEMENTATION-BASED IMAGE COPY-MOVE FORGERY DETECTION METHOD WITH CONSIDERATION OF SPATIAL CORRELATION

[1] *Yi-Jia Peng* (*彭奕嘉*), [2] *Chiou-Shann Fuh* (*傅楸善*)

[1] Dept. of Computer Science and Information Engineering,
National Taiwan University, Taiwan
E-mail: r03944006@ntu.edu.tw

[2] Dept. of Computer Science and Information Engineering,
National Taiwan University, Taiwan
E-mail: fuh@csie.ntu.edu.tw

ABSTRACT

In this paper, we improve an existing scheme of detecting the copy-move forgery. The main difference between our method and prior arts is that we seek spatial correlation in the test image. Our method has 4 stages. In the first stage, we segment test image. In second stage, we find the suspicious pairs and group them into suspicious groups by their spatial correlation, and then find a transform matrix between them. In third stage, we refine the transform matrix from step two by an iterated algorithm. In the last step, we use mix DSIFT feature approach and ZNCC feature approach and get a better estimation of copy-move forgery regions.

Keywords *Copy-move forgery detection; Image forensics; Segmentation; Spatial correlation*

1. INTRODUCTION

With the aid of image editing tools (e.g. Photoshop), we have more and more images that are forged artifacts. These forgeries can be applied to many illegal things, and make a large loss of economy, thus a forgery detection scheme become popular nowadays.

Since there are too many forgery methods, we cannot detect all of them. Thus in this paper we focus on copy-move forgery detection (CMFD). Copy-move forgery (CMF) contains at least a couple of identical regions. Forgers use CMF either to cover the truth or to enhance the visual effects of the image. As we can see, in Figure 1 there are two couples of images from Christlein et al.'s database [1]. Images in the first row use CMF technique to create a new tower, and images in the second row use it to hide buildings.

In the literature there are mainly two classes of CMFD algorithms [1]. One is based on block-wise division, and the other on keypoint extraction. The

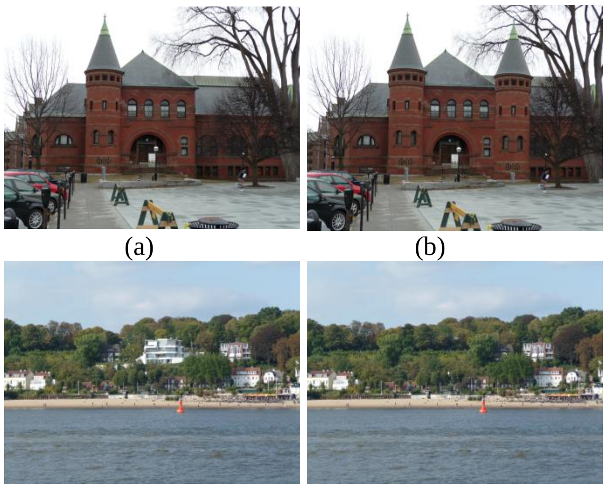


Fig. 1: (a)(c) are original images. (c)(d) are CMF images.

former divides the image into overlapping blocks and then finds the CMF by looking for the similar blocks. The later detects the CMF through observing the keypoints in the image.

Block-wise division seeks for similar blocks, this causes their computational complexity much higher than keypoint-based approach. Thus how to an effective and robust description of each block is needed. On the other hand, keypoint based approach can significantly reduce computation time and gives robust matching results of CMF detection. Because the keypoints lying spatially close to each other may be naturally similar, there should be a proper distance between two key-points that is comparable to each other. Most prior arts empirically select this threshold but neglect its relation-ship with the image size and content.

Since both of them have some defects, in [2] they proposed a method that is based on image segmentation, and use two stages approach to find CMF regions. They successfully overcome the defects in keypoint-based approach and block-based approach. Thus in this paper, we mainly focus on this approach and proposed an improved scheme to get better results.

The rest of the paper is organized as follows. In Section 2, we revisit [2]. In Section 3, we will give our new scheme for CMFD. The results are given in Section 4, followed by conclusion and discussion in Section 5.

2. SEGMENTATION-BASED CMFD

In this section, we will talk about the CMFD scheme proposed by [2]. [2] has totally three stages. In first stage, they segment input image into many patches. In second stage, they apply feature extraction and matching technique to find matching patches, and roughly calculate transform matrix between them. In third stage, they use an EM based algorithm to refine their transform matrix from second stage. We will revisit all of the three stage in section 2.1~2.3.

2.1. First stage

In order to separate copying source region from the pasting target region, the image should be segmented into small patches, each of which is semantically independent to the others. Thus in this step, [2] uses SLIC (Scale Invariant Feature Transform) algorithm to segment images [4].

SLIC is a wild used superpixel-based image segmentation method. It has three properties: First, each superpixel's boundary adheres well to image boundaries. Second, the algorithm is simple and the processing speed fast. Third, each superpixel in smooth region is regular lattice.

In practice, each image is segmented into no less than 100 patches, because [2] only consider the case that CMF regions are in different patches. We think it make sense, because if CMF regions are in the same patch, it must be too small to be detected by keypoint-based method since we segment image into many small patches.

2.2. Second stage

In this section, [2] introduce a matching process of segmented patches. It has 3 steps: Keypoint extraction and description step, matching between patches step, and affine transform estimation step. We will go through these steps sequentially.

2.2.1. Keypoint extraction and description

In this step, [2] use a very popular detection algorithm named SIFT (Scale Invariant Feature Transform) [5]. SIFT is a robust feature detection and description algorithm, it is useful in various research field such as image processing, pattern recognition, and so on.

2.2.2. Matching between patches

Next, [2] search for the suspicious pairs of patches by feature matching. Since CMF image has many identical parts, there must be many similar keypoints between two suspicious patches. If a pair of patch has a large proportion of matched keypoints, they consider being suspicious patch pair.

Since matching between any 2 key pair will consume a large amount of time, a useful technique is to build a k-d tree to speed up matching process.

2.2.3. Affine transform estimation

Since we have suspicious patches and matched keypoint between these two patches. We can simply use 3 of them to calculate a transform matrix by minimize the geometric distance $\sum_i \|x'_I - Hx_I\|^2$, where x' , x are locations of matched keypoint pair, and H is the transform matrix between them.

As we know, 3 point is enough of estimate an affine transform. However it is not robust at all. A common strategy is to apply RANSAC (Random Sample Consensus) algorithm for select a better key points [6].

RANSAC is a well-known algorithm for model fitting that can reduce outlier's influence. Thus it is used in various fields of fitting problem, as well as most of the CMFD schemes.

In this step, [2] also mentions an important thing: Some small size regions with limited number of keypoints, say 5, influence the detection accuracy. Clearly, the main reason is because the CMF regions are too small, and a limited number of keypoints cannot resist the possible error in keypoint extraction step. Observing this problem, [2] further come up with an EM-based algorithm for refining the result from second stage of matching.

2.3. Third stage

We know the reason why transform matrix is not robust is that we do not have enough keypoints. Thus [2] proposed to use all of the matched pixels in suspicious region to refine the transform matrix.

In this stage, [2] proposed a two-steps iterative algorithm: Obtaining the new correspondences of the pixels, and re-estimation of the transform matrix.

Finally they proposed a refining algorithm:

1. Obtaining the new matched points.
2. Calculating the new transform matrix.
3. Repeating the above two steps until a convergence condition is satisfied. (either the iterations is enough, or the old transform matrix is similar to the new one)

2.3.1. Obtaining the new correspondence of the pixels

In first step, we find new correspondence matched pixel pair (*x, x'*). Here [2] introduce a new feature named dense-SIFT (DSIFT). DSIFT is dense version of SIFT that can quickly compute descriptors for densely sampled keypoint with identical size and orientation. With DSIFT feature of each pixel, we can use L2-norm to measure the distance between 2 pixels.

Thus in first step, we fix *x*, and find a new *x'* nearby *x* (says 4 neighbors of *x* and *x* itself) with minimum DSIFT difference.

2.3.2. Re-estimation of the transform matrix

In this step, [2] use an EM-based method, thus they introduce a random variable *z* to indicate whether this pixel is in CMF region.

Finally they derive an equation of new transform matrix:

$$H_n = (X * W * X^T)^{-1} * X * W * \widetilde{X}'^T \qquad (1)$$

Where H_n is transition matrix H after n iteration, X, X' is correspondence pixel pair, W is $P(z = 1 \mid \vec{x_i}, H_{n-1})$ is a probability of whether x is in CMF region.

3. PROPOSED CMFD SCHEME

In this section, we will do some improvement of the CMFD scheme proposed by [2]. Our CMDF scheme has four stages. The details of them will in Section 3.1~3.5.

3.1 is an introduction to a toolbox we use, 3.2~3.5 is about our four stage CMFD scheme.

3.1. Toolbox

We use vlFeat [7] software to do segmentation and feature extraction.

The vlFeat open-source library implements popular computer vision algorithms specializing in image understanding and local features extraction and matching.

3.2. First stage

Exactly the same as [2] method, we use SLIC algorithm to segment images. The parameters we set are regionSize = 50, regularizer = 0.5 when size of input image is about 800*800 pixels also the same as [1].

In Figure 2(b), we can see SLIC can segment images into regular lattice in smooth regions, and can fit edges well.

3.3. Second stage

In this stage, we have 4 steps: Keypoint extraction and description step, matching between patches step, grouping patches step, and affine transform estimation step.

Keypoint extraction, matching patches, affine transform estimation steps are almost the same as [2].

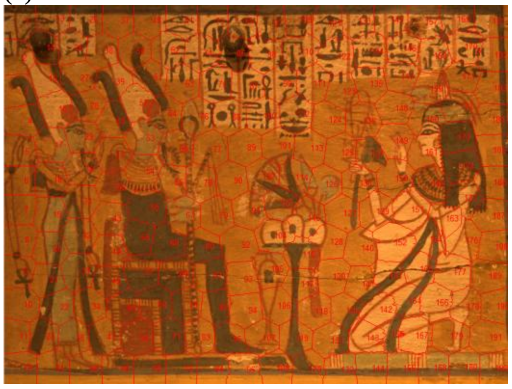
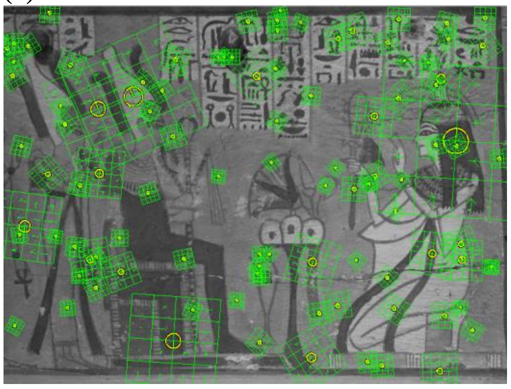
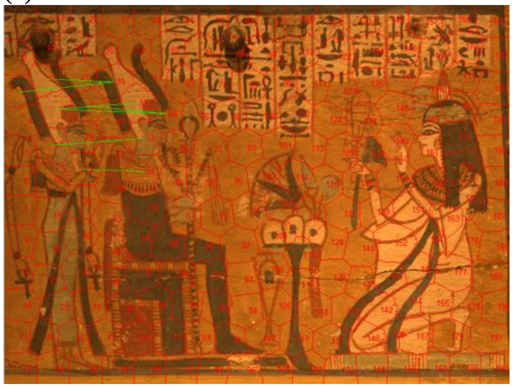
(a)

(b)

(c)

(d)

Fig. 2: (a) is input CMF images. (b) is segmented image. (c) is 100 SIFT feature points of CMF image. (d) is matching result of Section 3.2.2.

Grouping patches step is our new idea for a more robust matching result. We will introduce these four steps sequentially in Sections 3.3.1~3.3.4.

3.3.1. Keypoint extraction and description

Here we use vl_sift function to extract SIFT feature of input CMF image. The result can be seen in Figure 2. In Figure 2(c), we randomly plot 100 keypoints with their scales and orientations.

3.3.2. Matching between patches

In this section, [2] uses a threshold to determine whether the keypoint pair is matched, and uses

$$\emptyset = 10\frac{|\{key\ points\}|}{|\{patches\}|}$$

to be another threshold of matched patches.

However we found it does not make sense. If we use 10 times average keypoints per patches, we will drop out those patches which is in identical but only have less key points to match each other.

To deal with this problem, we proposed a two-step thresholding method for keypoint matching step.

In Step 1, we use a tight threshold for keypoint matching. When a pair of patches that contains more than 2 pair of matched keypoints, they will become suspicious patch pair. In this step we can find several suspicious patch pairs that are very similar to each other, but there are still some outliers. Thus in Step 2, we will focus on these patch pairs.

In Step 2, we use a loose threshold for key point matching, and we will use these matched keypoints to support suspicious patch pairs. If a pair of suspicious patch does not enough "support points", we will drop them. The definition of "support points" and criteria of "enough support points" are described as the follow:

The calculation of support points:

1. For each pair of patch, the support point is
$$\alpha * |\{tightly\ matched\ key\ point\}| + |\{loose\ matched\ key\ point\}|$$
where α is a weighting value. Here we simply set $\alpha = 1$.

The criteria of "enough support points"

1. The number of Support keypoints must larger than 3, since our calculation for affine transform must uses at least 3 key points.
2. If A patch and B patch has enough support key point, then
$$\text{Support point AB} \geq \beta * \max(\text{Support point AC})$$
And
$$\text{Support point of AB} \geq \beta * \max(\text{Support point of BC})$$

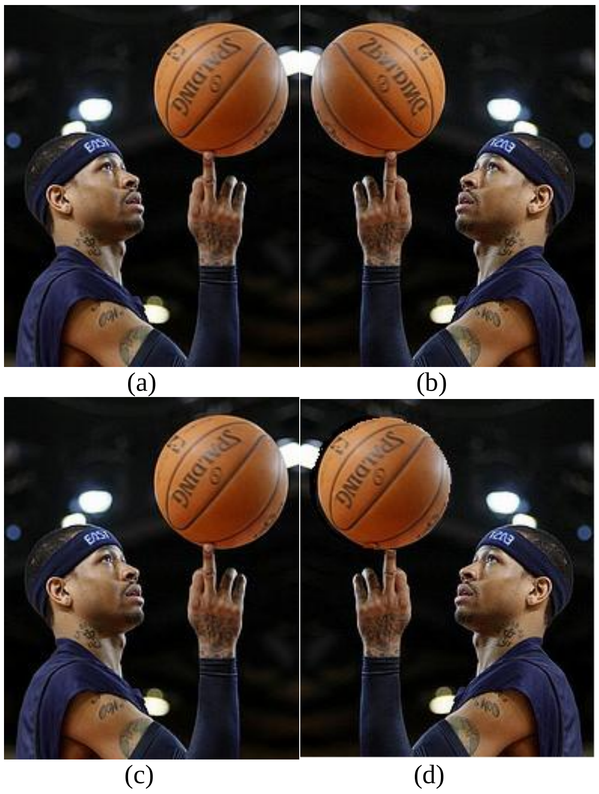


Fig. 3: (a)(b) are a couple of CMF regions. (c)(d) are another couple of CMF regions.

On implementation, we use vl_sift function to extract SIFT feature of input CMF image. The result can be seen in figure 2(d). The green lines indicate matched patch pairs.

3.3.3. Grouping patches

In this step we group nearby patch pairs by their space relationship.

We have 3 reasons to do this step:

1. The CMF region is usually bigger than 1 patch. Thus the matched pair in the same CMF region must have spatial correlation.
2. Since each matched pair contains limited keypoints, it will cause our affine transform estimation to be noisily. If we can group nearby patch pairs, we can have much more keypoint pair to do estimation, and get a better transform matrix.
3. With a better affine transform estimation result, we can have an even better refine result in third stage of matching.

The criteria for grouping match pair is very simple. Assume AB, and CD are matched patch pair. If A, C is connected and B, D is connected, we will group than into a bigger group. Most of the time it is enough for grouping patch pairs, however it sometimes will make mistakes.

For example: Figure 3 (a)(b) are a couple of CMF region with the man and the basketball have the same transform. However, Figure 3 (c)(d) are another couple of CMF region, as we can see the man is rotated, but the basketball is only shift to another side.

Thus if we simply use only spatial correlation sometimes it is not enough, the main problem is we do not consider the affine transform between two nearby patch pairs. We can simply modify our approach to consider the affine transform between the two nearby patches.

Though considering the same affine transform method can have a better result, we finally decide to use spatial correlation only. The reasons are:

1. Same affine transform method takes too much time.
2. Patches with less keypoint to estimate affine transform is noisy for same affine transform method.
3. If we use same affine method, we must decide a threshold for determining whether these two transform matrix are the same or not, it can be another problem.

3.3.4. Affine transform estimation

The same as [2], given any 3 matched keypoint pairs, we can calculate an affine transform between them. For robustness, we also adopt RANSAC algorithm to filter outliers. Practically after grouping matched patches, we can have enough keypoints for RANSAC to select inliers.

There is only one problem in this step: If keypoints is less than 5, it will be hard to get a robust affine transform. Even worst if the keypoints is too few, sometime it will cause a singular matrix that does not have an inverse matrix. For this case, we simply use a translation model to estimate transform between two patches.

3.4. Third stage

In this stage, we have 2 steps: Correspondence adjustment, and transform matrix re-estimation.

These steps are almost the same as [2]. We will talk about them in Section 3.4.1~3.4.2

3.4.1. Correspondence adjustment

The same as [2], we use vl_dsift function in vlFeat library to obtain DSIFT value, and find new correspondence pixel pair by a 4-neighbor minimum algorithm.

3.4.2. Transform matrix re-estimation

[2] derives Equation (1) and uses an EM-based algorithm to re-estimate transform matrix. We use another way to represent their derivation of Equation (1) as a quadratic optimization problem.

The objective function is given as:

$$\min_{H_n} \left\| \widetilde{X'} - H_n * X \right\|^2 \tag{2}$$

Use differential on $\mathrm{H_n}$ we get a solution:

$$H_n = (X * X^T)^{-1} * X * \widetilde{X}'^T \tag{3}$$

Compare with Equation (1), we can find that the difference is only a weighting *W*. It seems that Equation (1) is a quadratic optimization with each (*x, x'*) pair has a weighting on them. Thus in this step, we will propose a new weighting of *W* to get a better result.

The original *W* only use the L2-norm difference between DSIFT value of *H*X* and *X'* (or simply use intensity difference for speed up). However we think it is not enough, for instance: A pair of correspondence pixels on edge is more useful than on plane area.

Thus we must consider the spatial correlation between current pixel and its neighborhood. If current pixel is similar to its neighborhood, than it must in less texture area. If current pixel is dissimilar to its neighborhood, than it must in texture area.

Based on this observation, we use L2-norm DISFT difference between current pixel and its neighbor-hood's average to indicate the similarity. Thus our W term turn out to be:

$$W = e^{-(Hx-x')^T(Hx-x') - (x'-\overline{x'})^T(x'-\overline{x'})} \tag{4}$$

Same as [2], we can use intensity of *x, x'* instead of DSIFT value *x, x'* to calculate *W* for speed up.

3.5. Fourth stage

In this stage, we will introduce [3]'s method for CMF region identification, and use it to our region extension method.

3.5.1. Prior art for CMF region identification

In this section we will talk about [3]'s method for CMF region identification.

Since we have transform matrix for each suspicious group of patches. [3] uses a block-wise correlation measure based on zero mean normalized cross-correlation (ZNCC) between the gray-scale of the original image *I* and the warped image *W*. It is computed as

$$\mathrm{ZNCC}(x) = \frac{\sum_{v\in N(x)}(I(v)-\bar{I})(W(v)-\bar{W})}{\sqrt{\sum_{v\in N(x)}(I(v)-\bar{I})^2 \sum_{v\in N(x)}(W(v)-\bar{W})^2}} \tag{5}$$

where N(x) is a 7 pixels neighboring area centered at x; I(v), W(v) denote the pixels intensities at the location v; $\bar{I}, \bar{W}$ are the average pixels intensities of *I* and *W*.

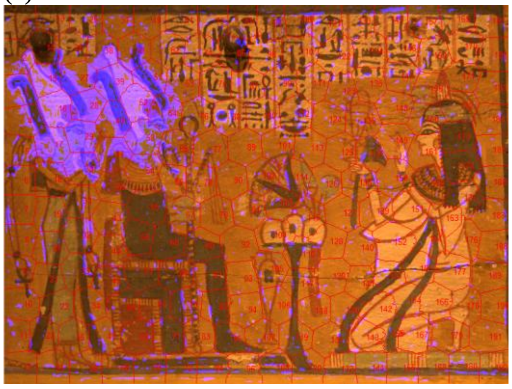
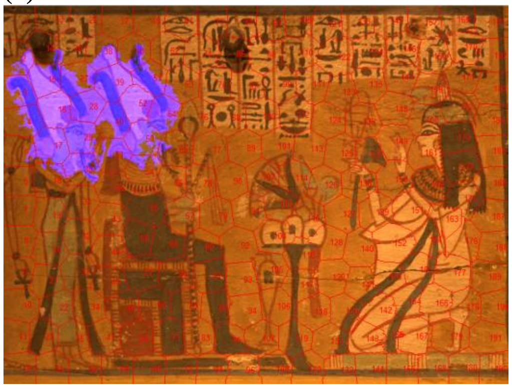
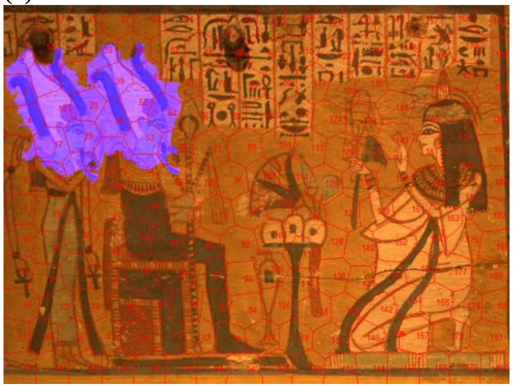
(a)

(b)

(c)

(d)

Fig. 4: (a) is original image. (b) is ZNCC region (c) is ZNCC region with hole filling and filtering (d) is our method

After calculate ZNCC of each pixel, they apply a Gaussian filter of size 7 pixels with standard deviation 0.5 to reduce the noise. Next they thresholding ZNCC image into binary by threshold=0.55. Thus they can get an image like figure 4(b).

As we can see the ZNCC method can successfully capture the edge of CMF regions, but contain many noise area, and holes in CMF regions. Thus they apply a mathematical morphological operation naming hole filling to fill holes and filter small area, and area does not contain corresponding feature points. The result is show in figure 4(c).

3.5.1. Our region extension method

ZNCC method uses a carefully selected threshold to binarize image. If we use a lower threshold value we will include too many pixels that are not in CMF region, on the other hand if we use a higher threshold value we will increase miss detection rate of CMF region.

In the previous section we use DSIFT feature to adjust correspondence pixels. [2] says DSIFT feature is robust to noise, but not discriminative enough. Thus if we use only DSIFT value difference to identify CMF region is not enough. Why not combine DSIFT feature and ZNCC feature together to get a better result?

Thus we come up with our region extension method. Let define some variables to be used in our method:

Z region:
: CMF region that detected by ZNCC method. This region is accuracy but has large miss detection area.

D region:
: CMF region that detected by DSIFT difference and restrict in. This region is accuracy and has low miss detection area if we carefully restrict it in a proper area.

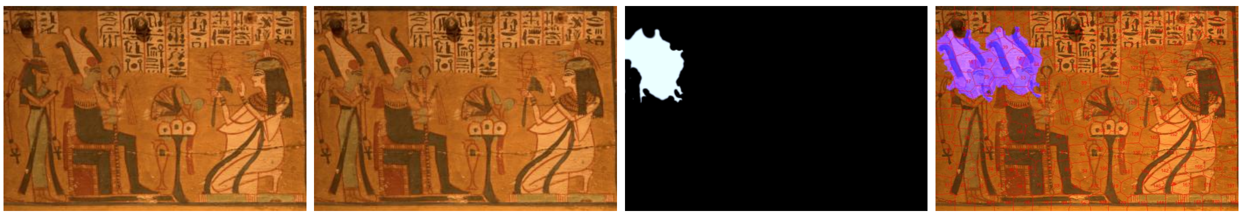
ZD region:
: CMF region that is obtained by or operation on Z region and D region.

Our method has 5 steps.

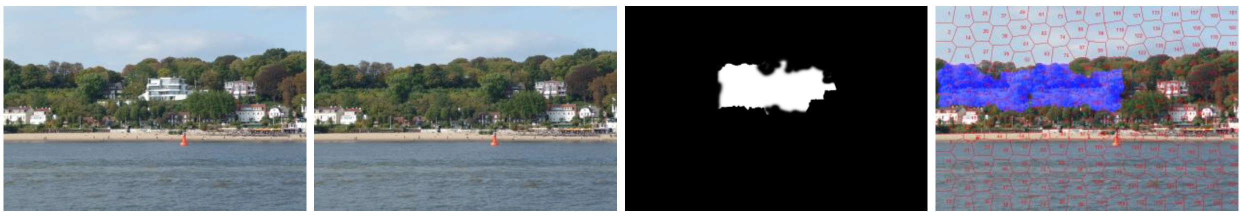
1. We use a higher threshold to obtain *Z* region.
2. We use DSIFT different between I, W image to find CMF region, and restrict its region within the suspicious patches (from Section 3.3) to obtain *D* region.
3. Using Z, D from Steps 1, 2, we can obtain ZD. If patches contain ZD region, and are near suspicious patches, we can extend suspicious patches to include this patch to obtain a bigger suspicious patch set.
4. Iteratively do Steps 1, 2, 3 until we cannot extend our suspicious patch set.
5. Finally we use ZD region as our CMF region identification result.

Table 1: CMFD Evaluation Benchmark Database.

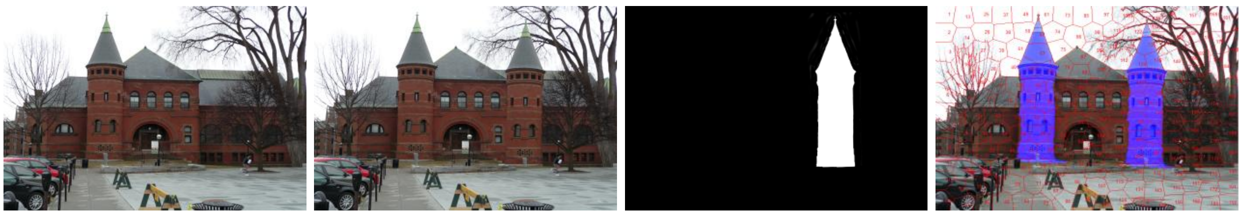
	Description
Source	Camera brand: Panasonic, Cannon, Nikon; Internet: Flickr
Amount	48 group of original image
Size	Camera: about 3000*2000pixels; Flickr: various
Format	PNG
Image Prototype	Original Image + Copy Region + Alpha Mask + CMF Image
Image Topic	Various



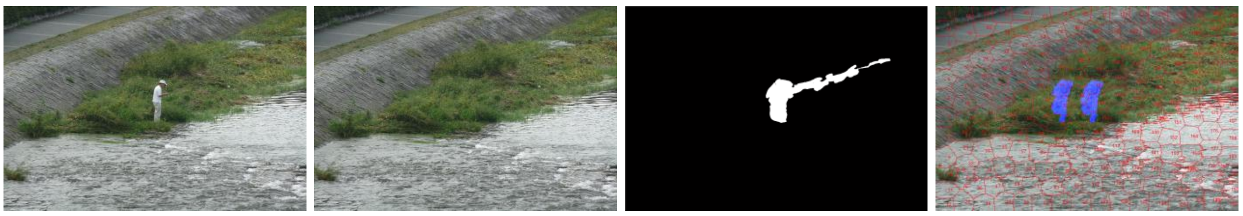
(a)



(b)



(c)



(d)

Fig.5: (a)(b)(c)(d) contain 4 image from left to right are original image, CMF image, ground truth alpha map for pasting region, and our detection results with detected CMF region and segmentation results.

The result of our method can be seen in figure 4(d). With our method we can successfully detect the identical necks of Egyptian.

4. RESULTS

In this section we will introduce the database of our CMFD scheme, and show some results.

4.1. Test image database

Table 1 presents our database. The database was constructed by Christlein et al [1]. These CMF image is carefully selected such that the CMF trace is almost unnoticeable. Furthermore we resize those images to 800*600 pixels (a non-uniform image resizing) which make it difficult to extract enough keypoints from the CMF region.

4.2. Results of our method

Show in figure 5, we can find our method can successfully find CMF regions. However we fail to detection the fishing rod in figure 5(d), we think it is too thin to be matched by feature extraction and matching step, and detected by our CMF region identification process. If we want to detect the fishing rod, we can a loosen threshold for feature matching, or set a lower ZNCC threshold and DSIFT difference threshold. But it will cause false alarm for other test data. Thus there is a tradeoff between them.

5. CONCLUSION AND DISCUSSION

In this section we will discuss the limitation and give our conclusion.

5.1. Limitations

Since our scheme is based on keypoint detection and matching, it is hard for small image and texture less image that do not have enough keypoints.

Another limitation is that we first segmentation image into several patches, and matching keypoints only on different patches. If the CMF regions are in the same patch, we cannot detect them. As keypoint based approach, they need to carefully choose a distance between to comparable key points, we need to carefully segment image into several patches. The advantage of ours is we can use adaptive threshold (provided by image segmentation boundary) for comparable keypoint distance selection.

5.2. Conclusion

This paper presented a CMFD scheme based on image segmentation considering spatial correlation. We have 2 contribution in this work.

1. For transform matrix estimation, this algorithm combine block-wise based approach and keypoint based approach. Though it is first seen in [2], we further propose a spatial correlation approach to improve robustness.
2. For CMF region identification, this algorithm mix DSIFT feature, and ZNCC feature to give a more robust result.

REFERENCES

[1] V. Christlein, C. Riess, J.Jordan, C.Riess, and E. Angelopoulou, "An evaluation of popular copy-move forgery detection approaches," IEEE Trans. Inf. Forensics Security, vol. 7, no, 6, pp. 1841-1854, Dec. 2012

[2] Jian Li, Xiaolong Li, Bin Yang, and Xingming Sun, "Segmentation-Based Image Copy-Move Forgery Detection Scheme", IEEE Transaction on Information Forensics and Security, vol. 10, no. 3, pp. 507-518, Mar. 2015.

[3] I. Amerini, L. Ballan, R. Caldelli, A. Del Bimbo, L. Del Tongo, G. Serra. "Copy-Move Forgery Detection and Localization by Means of Robust Clustering with J-Linkage", Signal Processing: Image Communication, vol. 28, iss. 6, pp. 659-669, 2013

[4] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC Superpixels Compared to State-of-the-art Superpixel Methods", IEEE Transaction on Pattern Analysis and Machine Intelligence, vol 34, no. 11, pp. 2274-2282, Nov. 2012.

[5] D. G. Lowe, "Distinctive image features from scale-invariant keypoints", Int. J. Comput. Vis., vol. 60, no. 2, pp. 91–110, Nov. 2004.

[6] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," Commun. ACM, vol. 24, no. 6, pp. 381–395, Jun. 1981.

[7] A. Vedaldi and B. Fulkerson. (2008). VLFeat: An Open and Portable Library of Computer Vision Algorithms. [Online]. Available: http://www.vlfeat.org/