DETECTING MULTIPLE COPIES IN TAMPERED IMAGES

E. Ardizzone, A. Bruno, G. Mazzola

Dipartimento di Ingegneria Informatica (DINFO) dell'Università di Palermo.
Viale delle Scienze – Ed. 6, 90128, Palermo, Italy
ardizzon@unipa.it, bruno@dinfo.unipa.it, mazzola@dinfo.unipa.it

ABSTRACT

Copy-move forgeries are parts of the image that are duplicated elsewhere into the same image, often after being modified by geometrical transformations. In this paper we present a method to detect these image alterations, using a SIFT-based approach. First we describe a state of the art SIFT-point matching method, which inspired our algorithm, then we compare it with our SIFT-based approach, which consists of three parts: keypoint clustering, cluster matching, and texture analysis. The goal is to find copies of the same object, i.e. clusters of points, rather than points that match. Cluster matching proves to give better results than single point matching, since it returns a complete and coherent comparison between copied objects. At last, textures of matching areas are analyzed and compared to validate results and to eliminate false positives.

Index Terms— Image Forensics, SIFT, Clustering

1. INTRODUCTION AND RELATED WORKS

Digital Image Forensics[1] deals with the problem of certifying the authenticity of a picture, or its origin, without explicit a priori information, e.g. using watermarks. Three are the main branches in this research field: image source identification; discrimination of computer generated images; image forgery detection. In this paper we focused with the problem of detecting image forgeries. Images can be doctored in several ways[2]: photo-compositing, re-touching, enhancing are only some examples of typical image alterations. Copy-Move[3] is one of the most common forgery used for image tampering, typically to delete some objects from the scene, and to substitute information with some other from the same image. The most simple approach to find copies is exhaustive search, to compare an image with every cyclic-shifted version of itself. This approach is computationally very expensive, then many methods use block-matching approaches[4][5]. These approaches proved to be robust to noise addition, compression, and retouching, but very few of them are effectively robust against geometrical attacks (rotation, scaling, distortion), and none deals with the problem of multiple copies. In these paper we use SIFT [6] (Scale Invariant Feature Transform) descriptors, which are invariant to scale and rotation, and relatively robust to perspective changes.

The paper is organized as follows: section 2 describes a SIFT-based detection method, which inspired our approach, and its limitations; in section 3 we present our SIFT-point cluster matching approach; section 4 shows compared results; a conclusive section ends the paper.

2. SIFT-POINT MATCHING METHOD

In [7] Huang et al. uses SIFT descriptors to detect copy move forgeries in an image, by matching key-points in order to detect similar points in the scene. A key-point matches if the ratio of its nearest neighbor, according to the Euclidean distance between descriptor vectors, and the second-closest neighbor is less than a threshold. Due to the strong stability of SIFT feature descriptors this method has a good performance on different kind of image post-processing (JPEG compression, rotation, noise, scaling etc.) and is robust to changes in noise, illumination, distortion and point of view. However, this approach presents several drawbacks. First, it detects many false matches between close but distinct points (points A in fig. 1.a). They can be easily removed setting a distance threshold between points coordinates, but this does not completely remove the problem of false matches. In fact, matches can be found between points that are similar, but that are not part of copied objects (points B in fig. 1.a) or, in case of multiple copies, points of the same object may match with different copies (see fig. 4.a). These are incoherent matches. In some cases, copies are not detected because better matches are chosen. Last, analysis of the results is left to the user.

3. SIFT-POINT CLUSTER MATCHING

Our approach aims to a more challenging goal, with respect to the point-matching approach: we search for objects that match, rather than points. A similar approach was proposed in [8], in which a supervised method, based on SIFT features, is used to find commercial logos in sport videos and in [9], for robotics application. Our method is fully automatic and can be divided into three sub-steps: SIFT point clustering, Cluster matching and Texture analysis.

3.1. SIFT-point Clustering

Given an input image, after gray-scaling, we first compute SIFT points and descriptors for the whole image. Each keypoint is therefore described by its coordinates and by its vector of SIFT descriptors. Points are grouped using an agglomerative hierarchical-tree cluster method. We build the matrix of Euclidean distances between each point coordinates, and then we use the weighted center of mass distance (WPGMC) as linkage method. For each hierarchy of the tree, if r and s are two clusters to be combined, we compute the Euclidean distance between their weighted centroids:

$$d(r,s) = \|\tilde{x}_r - \tilde{x}_s\|_2 \quad (1)$$

where $\tilde{x}_r$ (and as well $\tilde{x}_s$) is defined recursively as:

$$\tilde{x}_r = \frac{1}{2}(\tilde{x}_p + \tilde{x}_q) \quad (2)$$

and $\tilde{x}_p, \tilde{x}_q$ are the weighted centroids of the clusters p and q from the lower step. The "best" number of clusters is automatically detected with a posteriori analysis. The cluster tree is formed, once for all, and then we test a set of candidate numbers of clusters (n_i i=2,..nmax). For each candidate n_i, we compare all the possible pairs of clusters within the correspondent clustering and we count the number of matching points (see section 3.2) for each pair of clusters N(j,k). We consider only those matches where the number of matching points is higher than a threshold:

$$M(j,k) = \begin{cases} 0 & N(j,k) < th \\ N(j,k) & otherwise \end{cases} \quad (3)$$

and we compute the score for the candidate n_i as:

$$S_i = \sum_{j=1}^{n_i} \sum_{k=j+1}^{n_i} M(j,k) \cdot \quad (4)$$

We select the "best" number of clusters that maximizes this score, i.e. that gives the highest number of significant matching points (see fig. 2.c,d). In case of multiple maxima, we select the lowest number of clusters, for efficiency.

3.2. Cluster Matching

In this step we match clusters of points, rather single points. If C_i and C_j are the two clusters to be matched, for each point $i \in C_i$ we compare its vector of descriptors $\bar{d}_i$ with the vectors $\bar{d}_j$ of all the points $j \in C_j$. For efficiency, rather than using Euclidean distance, we compute the angle between the two vectors:

$$\alpha_{ij} = \arccos(\bar{d}_i \cdot \bar{d}_j) \quad (5)$$

and

$$\alpha_1 = \min_{j \in C_j} \alpha_{ij} \quad \alpha_2 = \min_{j \in C_j - j_1} \alpha_{ij} \quad (6)$$

where j_1 is the point in C_j which vector $\bar{d}_j$ has the minimum angle with vector $\bar{d}_i$. To increase robustness, matches are accepted only if the ratio of the two minimum angles, α_1 and α_2, is less than a threshold (0.6, as in the reference method described in section 2). Note that, for small angles, this is a close approximation to the ratio of Euclidean distances. To improve the matching process, for the "best" clustering, we apply RANSAC (RANdom SAmple Consensus) to select a set of inliers that are compatible with a homography transform between the two clusters. After RANSAC, outliers are discarded and the corresponding homography is estimated. If less than 4 matches are found, RANSAC cannot be applied, and the match between the two clusters cannot be considered reliable. In case of a small number of matching points, which may occur comparing small clusters, estimated homography does not give good results. Therefore, after discarding outliers, we use the Moore-Penrose pseudoinverse of matrix to solve the system of linear equations of the matching points' coordinates, to estimate the homography. Then the inverse transformation is applied in order to register the two areas, and the two smallest convex hulls, A_1 and A_2 , that enclose matching points (see figg. 2.e,f) are extracted.

3.3. Texture analysis

The content of the two areas is analyzed by texture descriptors in order to remove possible false matches. Prewitt and Laplacian masks are used to extract vertical, horizontal, diagonal (North-East and North West) and not-oriented edges. Descriptors are then computed as the average value of pixels in the area after filtering. If the distance between the descriptor vectors of the two areas is lower than an experimental threshold, they are considered identical, and their match correct. Texture analysis step is needed to validate results from the previous matching process, in order to discard matches between objects that are similar, but not copies. Reference and approximated areas can be slightly different almost for three reasons:

- Interpolation applied by inverse transformation;
- JPEG compression after tampering;
- Approximation errors (SIFT-point extraction, homography estimation, inverse transformation)

With respect to the first problem, experimental tests showed that using bicubic interpolation reduces the difference between reference and approximated area.

With regard to the second one, proposed descriptors extract information from image edges, which are less affected by noise introduced by JPEG compression.

The third problem needs a specific solution: after inverse transformation, the two areas are not perfectly aligned (see fig. 3.b,c). For larger areas this is not a relevant problem, as edge information is averaged. In case of smaller ones, due to this misalignment, some edges in the reference area could be missing in the approximated one, and the two areas may look consistently different, resulting in descriptor values that may not give correct match. Therefore a local alignment is needed. We apply a further rigid translation to the points of the approximated area in a neighborhood of 5x5 pixels centered into original coordinates. For each translation we compare descriptors of the two areas, and whenever their distance is lower than the above experimental threshold,

Table 1. Percentage of correct matches (PC) for Keypoint Matching (KM) and Cluster Matching (CM). For CM we measured also the percentage of false negatives (FN) and of undetected copies (UC)

algorithm	PC (%)	FN (%)	UC (%)
KM	44\72	n.m.	n.m.
CM	92	9	5

local alignment stops, and the two areas are marked as copies. Significant improvements in results justify the increased execution time.

4. EXPERIMENTAL RESULTS

In this section we compare experimental results obtained with the reference (keypoint matching - KM) and the proposed (cluster matching - CM) methods. Both accuracy and execution time are measured. In our implementation we use the Matlab code in [10] for SIFT extraction, and that in [11] for RANSAC. Tests were made on about 100 medium-sized images (500x500 ca.), 80 of which contain multiple copies of an object of the scene, while 20 of them does not have copies, but similar objects. Tests are executed on an Intel Core Duo PC (1,83 GHz, 2 GB RAM). In figg. 4, 5 and 6 visual results are compared for the two methods. Note that it is very difficult to give a numerical comparison, as first method returns points that match, while our method works with sets of points that match. With regard to the percentage of correct matches (PC), as shown in table 1, our method outperforms (92%) the KM method, both when measuring precision before (44%) and after (72%) applying a distance threshold, to discard very close matches. False positives are detected only in case of very similar objects (see fig. 1.b). In case of multiple copies, for CM we measured also the percentage of undetected matches (9%), i.e. links between copies, and of undetected copies (5%), i.e. copies that do not match with any other ones in the image. For the KM approach false negatives (keypoints which do not match with other points) are not measurable.

As discussed in section 2, for KM, different points from the same object can match with different objects (fig.4.a). This is not a problem with our method (fig.4.b), because it compares all the possible cluster pairs. Our method is also able to detect deleted objects (fig. 6), in case of textured background, but it fails whenever no SIFT points can be extracted from copied objects, e.g. in case of homogeneous areas. In this case block matching approaches are preferred.

Regarding efficiency, time spent for SIFT extraction is the same for the two methods (2,8s). Moreover for KM average time to match keypoints is 1,3 s. For CM we measured also time to build cluster tree(1-30s), to find best clustering (3-120s), average time for each cluster matching (0,01s), and average time for description(1,4s) (for each translation during the local alignment phase). Only partial times for matching and analysis are shown because total time strongly depends on the number of clusters. Our tests showed also that CM is robust up to JPEG quality 30.

5. CONCLUSIONS

In this paper we proposed a method to detect copied objects in a digital image, by matching clusters of keypoints, extracted by SIFT. Isolated matches are discarded and only coherent matches are found: sets of neighbor points that match with other sets of points, according to a projective transformation. Furthermore, for each match, we detect the specific transformation that has been applied to create the copy. Texture analysis is used to compare the content of the two matching objects, in addition to their shape. This makes our method very robust against false matches. Only objects which have both similar shape and texture can be considered as real copies. With regard to common attacks, our method inheriting its robustness mainly by SIFT properties, work as well in case of JPEG compression, added noise and geometrical transformation (rotation, scaling, limited distortion). Another important issue is the number of clusters. A low number of clusters makes the matching phase unable to detect multiple objects, if they are part of the same cluster. Setting an higher value, clusters may be made of too few points, and not enough matches can be found to estimate the relationship between them. Our solution (selecting the number of clusters which maximize the number of significant matching points), is the best trade-off between these two aspects.

6. REFERENCES

[1] Sencar, T., and Memon, N.,"Overview of State-of-the-art in Digital Image Forensics". *World Scientific Press*, 2008

[2] H. Farid: "A Survey of Image Forgery Detection", *IEEE Signal Processing Magazine*, 26(2):16-25, 2009

[3] Bayram, S., Sencar, T., and Memon, N., "A Survey of Copy-Move Forgery Detection Techniques", *IEEE Western New York Image Processing Workshop*, Sept. 2008, NY

[4] Fridrich, J., , Soukal, D., and Luk, J., "Detection of Copymove Forgery in Digital Images," *Proc. Digital Forensic Research Workshop, Cleveland, OH*, August 2003.

[5] Li, G., Wu, Q., Dan Tu and Sun S. ,"A Sorted Neighborhood Approach for Detecting Duplicated Regions in Image Forgeries Based on DWT and SVD," *ICME*, 2007.

[6] Lowe, D. G., "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision*, 60, 2 (2004), pp. 91-110.

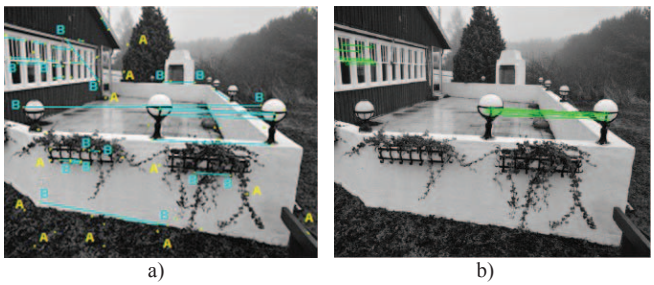
[7] Huang, H., Guo, W. Zhang, Y., "Detection of Copy-Move Forgery in Digital Images Using SIFT Algorithm," *PACIIA 2008. Pacific-Asia Workshop on* , vol.2, no., pp.272-276.

[8] Bagdanov, A. D., Ballan, L., Bertini, M., and Del Bimbo, A., "Trademark Matching and Retrieval in Sports Video Databases", in *Proc. of MIR), 2007* pp. 79—86

[9] Zickler, S., Veloso, M., 2006. "Detection and localization of multiple objects", in *Proc. of 6th IEEE-RAS International Conference on Humanoid Robots*, Genova, Italy, pp. 20-25.

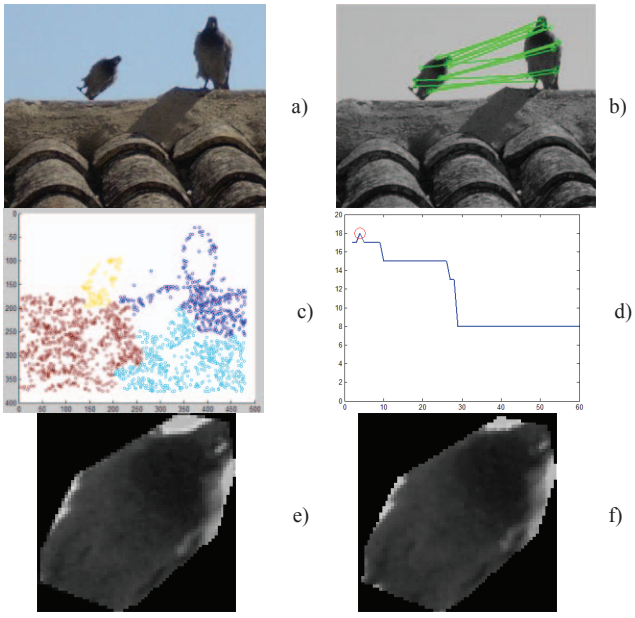
[10] http://www.vlfeat.org/~vedaldi/code/sift.html

[11] http://www.csse.uwa.edu.au/~pk/research/matlabfns



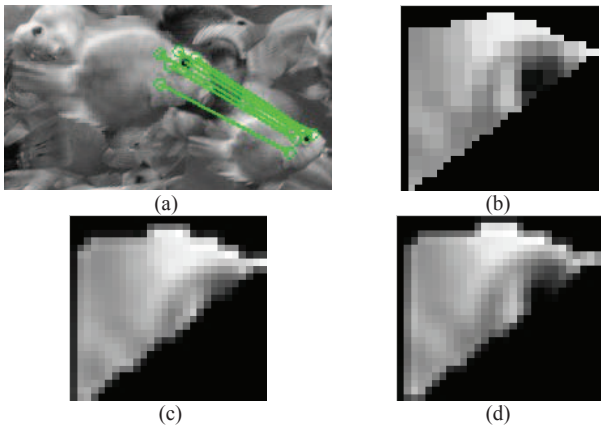
a) b)

Fig. 1. The lamp on the center is a copy of that on the right. Results after keypoint matching a) and cluster matching b). Much less false positives are detected in b) (only the two windows, which have very repetitive structure). Note that in b) the other lamps in the image (not copies) are correctly not detected.



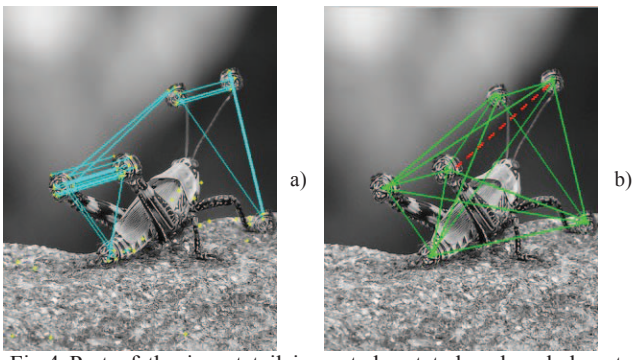
a) b) c) d) e) f)

Fig. 2 (a) input image, the bird on the left is a rotated and scaled copy of that on the right. (c) Clusters of SIFT keyponts (4 clusters). (d) number of significant matching points vs number of clusters. The enveloping convex areas for the two set of points to be analyzed: reference (e) and approximation (f).



(a) (b) (c) (d)

Fig.3 (a) matching points between two objects in the scene (particular). (b) reference area (part of the fish eye). Approximation without (c) and with (d) local alignment.



a) b)

Fig.4 Part of the insect tail is pasted, rotated and scaled, onto other parts of its body. Results with KM (a) and CM (b) (n°clusters=23, only matches between centroids are shown). In both at least one match is found for each copy, but CM detects more matches: only one is missing (red dashed line, superimposed). Furthermore in (a) some matches are weak, since made of only one point.

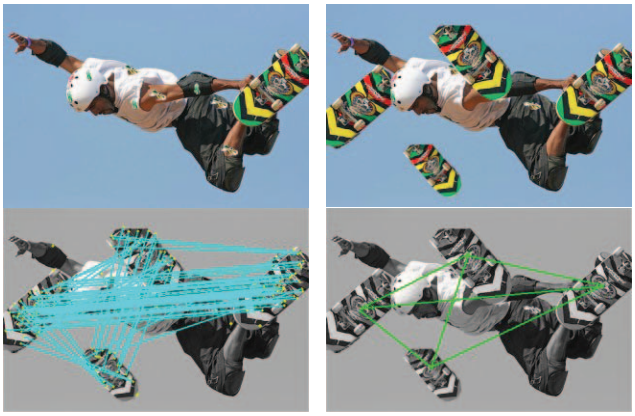
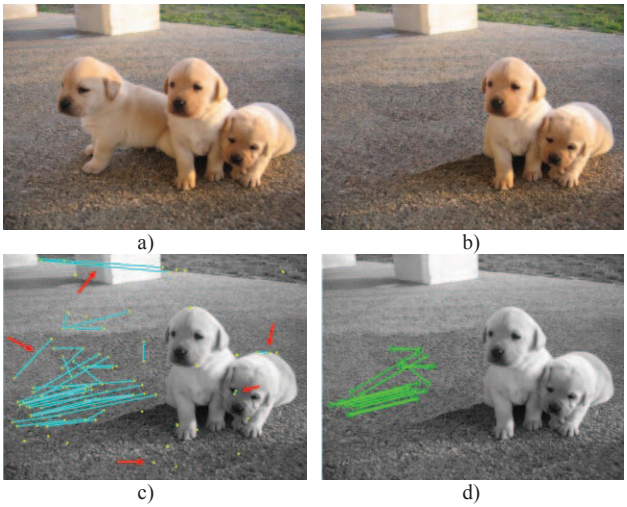


Fig. 5 (a) Original and (b) tampered image. (c) Results with KM and with (d) CM (n°clusters =4, matches between centroids are shown). Note that in (b) only two matches are found between the skateboard in the bottom part of the image and that on the right. In (d) all the copies match each others



a) b) c) d)

Fig. 6 An example of object deletion using textured areas: original a), modified b), detection with KM (c), and CM (d). Some false positives in (c) are indicated by red arrows.