Effective Palm Tracking with Integrated Tracker and Offline Detector

Zhibo Yang, Yanmin Zhu and Bo Yuan

Intelligent Computing Lab, Division of Informatics
Graduate School at Shenzhen, Tsinghua University, Shenzhen 518055, P.R. China
yangzhibo450@gmail.com,zhuym111@gmail.com,
yuanb@sz.tsinghua.edu.cn

Abstract. In this paper, we propose a vision based palm tracking method with three inherently connected components: i) an offline palm detector that locates all possible palm-like objects; ii) a SURF-based tracking module that identifies the tracked palm's location using historical information; iii) an adaptive skin color model and a patch similarity calculation module. The outputs from the last component can effectively eliminate false detections and decide which palm is under tracking and also provide updated information to the first two modules. In summary, our work makes the following contributions: i) an effective offline palm detector; ii) a benchmark dataset for training and testing palm detectors; iii) an effective solution to tackling the challenges of palm tracking in adverse environments including occlusions, changing illumination and lack of context. Experiment results show that our method compare favorably with other popular tracking techniques such as Camshift and TLD in terms of precision and recall.

Keywords: Palm Tracking, Detection, SURF, Skin Color Model, Integrator

1 Introduction

In recent years, vision based dynamic hand gesture recognition has received growing interests from the computer vision community. Due to its ease of use and power of metaphor, hand gesture has long been serving as an essential communication tools since the beginning of human society and has great potential in human computer interaction (HCI) [1]. Palm tracking is one of the most important components of dynamic hand gesture recognition. Typical applications include vision based remote control, sign language recognition and palm localization in palm print recognition. However, palm tracking in unconstrained environments still remains as a challenging task due to factors such as changing illumination, occlusions, deformable palm shape and the lack of body or arm context information.

One of the key steps in palm tracking is finding a proper set of features so that the palm region can be extracted as accurately as possible. Recent studies [2] show that, for hand detection, color is an effective feature compared to other texture features such as LBP (Local Binary Pattern), Haar-like [3] and gradient features. However, the color feature can fail when the illumination condition is poor. In this paper, we will

investigate the possibility of adopting non-color features such as LBP and normalized HOG (Histogram of Oriented Gradient) in the training of offline detectors while using the color feature as complementary information.

Another issue in palm tracking is how to use historical cues to increase the precision of tracking. In complex situations, offline detectors may often produce false or multiple detections and it is also difficult to obtain satisfactory tracking results consistently as the feature points may be unstable. To address this issue, there is a need to build a link between the detector and the tracker. In our work, we will show how a SURF (Speeded Up Robust Features) based tracking module can be used to propagate the information of accurately tracked palm on a frame to frame basis.

Furthermore, a representative human palm dataset is essential in the research of palm tracking. Unfortunately, there is a lack of publicly available datasets that are well suited for this purpose. As a result, a fully annotated palm dataset containing samples from various sources is introduced, as shown in Fig. 1.

Fig. 1. Some positive hand samples in our benchmark dataset

2 Related Work

The issue of segmenting hands from images or videos in constrained environments has been extensively studied in the literature. However, in complicated real-world scenarios, methods independently using skin detection [4-6] or cascade detectors based on wavelet features [7, 8] often fail in tracking. Some approaches treat human pictorial structure as the spatial context for locating hand positions [9, 10]. However, these methods require the location information of certain human body parts (e.g., head and arms), which are not always visible in the image, especially when the user is close to the camera. In [11], a novel approach to hand segmentation is based on creating hand regions from contours, but extra information is still required in this method. There are also some studies in which different techniques are combined to achieve improved performance. For example, in [12], three detectors and two segmentation methods were used to locate the hand in images. However, this method cannot be extended to hand tracking in video due to its high computational complexity. By combining traditional tracking methods with the skin color features, it is possible to achieve fast tracking but the system is not robust enough against the interference of hand-like objects [13, 14].

Other well known tracking methods such as block tracking and the state-of-the-art TLD algorithm [15] have been tested in our preliminary experiments. The block tracking can fail to track the palm for many reasons. For example, the face can be accidently tracked after the hand moves across it. The failure of TLD is because there is not sufficient training data for the online detector and the deformation of palm can produce many problematic patches that may compromise the performance of detector. Motivated by the methods mentioned above, the proposed method not only detects the palm using multiple descriptors but also integrates the detections and tracking results, resulting in significantly better tracking performance.

3 Framework

This section gives an overview of our palm tracking method, as shown in Fig. 2. There are four components: pre-processing module, palm detector trained offline, SURF-based tracker and integrator. The pre-processing stage including image smoothing and color space transformation can effectively decrease noises, which may otherwise be falsely treated as tracking points. The task of tracking is actually to track the foreground and a foreground detector can be helpful [16]. In our method, multiple cues are used to detect the foreground and both the offline detector and the tracker take the preprocessed frame and the feedback of the integrator as inputs.

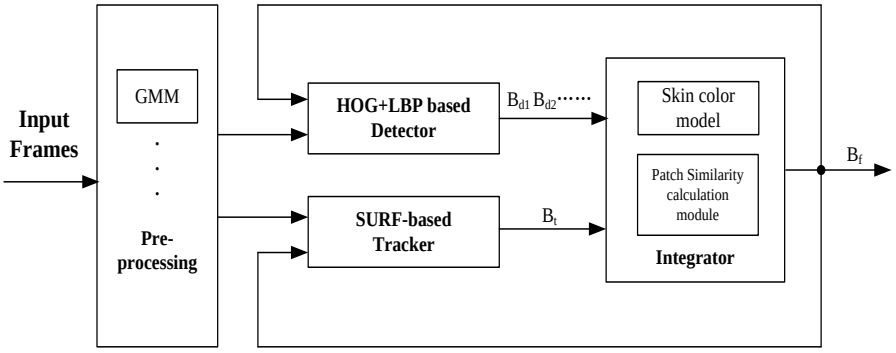

Fig. 2. The framework of the proposed tracking process

If there are multiple palm-like objects in the image, the detector will recognize all of them and output their bounding boxes B_{d1}, B_{d2},⋯. We noticed that classifiers trained offline were unable to detect the palm with large rotation degrees (e.g., more than ± 15 degrees). In practice, the palm is unlikely to rotate sharply between two frames. Therefore, it is possible to inject the rotation information into the detector by analyzing the bounding box in the last frame. The SURF-based tracker takes as input two consecutive frames and the bounding box in the last frame. A number of SURF points within the bounding box are tracked and outliers are filtered out. Remaining points are used to estimate the motion of the box region and the location of the new tracking box is equal to the location of the last bounding box plus the motion offset.

4 Implementation Details

4.1 Offline Palm Detector

We collected 500 palm images as positive training samples, together with their left-right reflections (1000 images in all), as shown in Fig. 1. Some of the images were generated by our lab members while others were retrieved from Cambridge Hand Gesture Data Set [17]. Negative training samples were based on the images used in a face detection project [18] and all images containing hand-like objects were removed, resulting in 6465 normalized images.

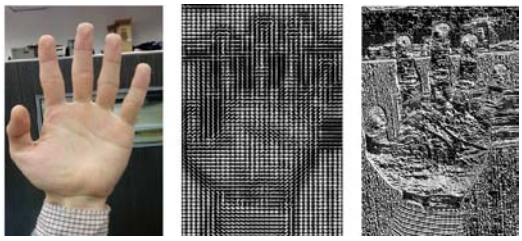

Fig. 3. A palm (left) and its HOG (middle) and LBP (right) reconstruction images

For feature extraction, we tested the discriminative power of different features: HSV, RGB, YCbCr, Gabor, Harr-like, HOG, BRIEF and LBP. In our results, color features often failed when illumination was poor, and the last three features ranked higher than Gabor and Harr-like features. Since HOG describes the shape and contour of palm while LBP is invariant to grey value and rotation, when combined together, they may generate better results. Therefore, HOG and LBP descriptors were selected to form combined feature vectors (Fig. 3).

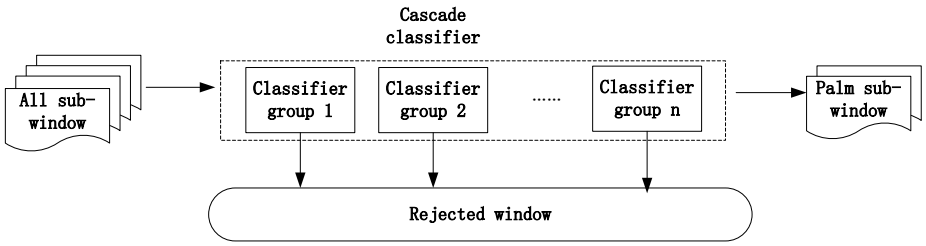

Fig. 4. The workflow of cascade classifiers

Cascade classifiers were used in the classifier training process, as shown in Fig. 4. The number of classifier stages depends on the specific training process. In each stage, the GAB (Gentle AdaBoost) algorithm boosted a strong classifier consisting of a group of decision trees (weak classifiers). The maximum tolerable false positive rate FPR (50% in our experiments) and minimum tolerable true positive rate TPR (99.9% in our experiments) influenced the number of weak classifiers in each stage. During

testing, all sub-windows went through stages one by one, and more than half of the remaining negative sub-windows were rejected at each stage. In our experiments, the cascade had 20 stages and the number of weak classifiers was between 5 and 28. The overall false positive rate was $1-0.5-0.5^2\cdots-0.5^{20}\approx 0.0001\%$ and the overall true positive rate reached $0.999^{20}\approx 98\%$. Finally, the cascade classifiers identified all palm-like objects and generated a bounding box for each of them.

4.2 SURF-Based Tracker

There are two key factors for achieving reliable tracking: the selection of feature points and the point matching strategy. Different from the standard KLT tracker [19], we used SURF feature points [20] and, for each SURF point within the bounding box B_f in the frame F^t, its location in F^{t+1} was tracked. Next, the same procedure was conducted in the opposite direction, which means that we tracked backward the point that has already been tracked in F^{t+1}. For example, in Fig. 5 (right), given P_0 in frame F^t, its corresponding point P_1 in frame F^{t+1} is located. When we track P_1 backward, P_2 is identified. The Euclidean distance between these two points is then calculated. If the distance between P_1 and P_2 is smaller than a predefined threshold, P_1 will be treated as an effective tracking point. Otherwise, P_1 will be abandoned.

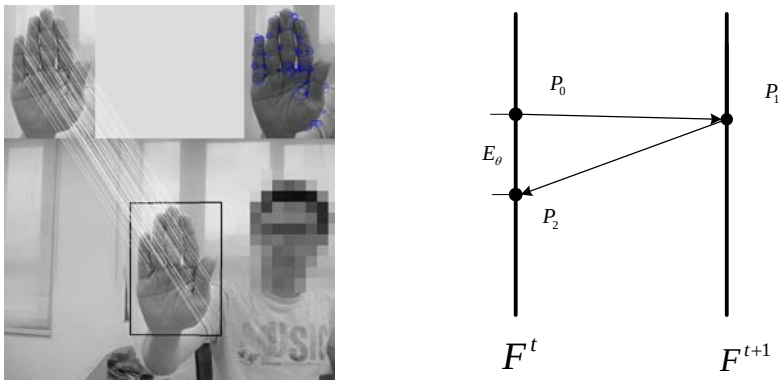

Fig. 5. Examples of extended KLT tracking based on SURF points and the forward-backward point matching strategy

The implication of our forward-backward tracking is as follows. For points that are visible in both frames, it is usually easy for the tracker to locate them correctly. If a point is invisible in the next frame, the tracker may locate a different point, leading to false tracking. Fig. 5 (left) gives an example of tracking SURF feature points, which are labeled as blue circles over the palm in the upper-right corner.

In summary, the tracking precision of the standard KLT tracker can be improved due to the use of SURF tracking points and the specific point matching strategy. The SURF points are invariant to rotations and scale change while falsely tracked points can be effectively removed by tracking them back to the last frame. The details of the error estimation of this bi-directional tracking can be found in [21].

4.3 Integrator

The integrator combines the outputs of the offline detector and tracker. It provides the rotation information to the detector and the object to be tracked in the next frame. In the following, our work is conducted in the perspective of patch. A single instance of the object's appearance is represented by an image p called patch, which is either described by color statistics or texture statistics. We generated all possible patches by shifting an initial box with the following parameters: scale step = 1.2, horizontal step size = 10% of the width, vertical step size = 10% of the height and minimal box size: 20×30 pixels for a 640×480 image. Finally, all these grids were normalized to create uniform patches with 40×60 pixels.

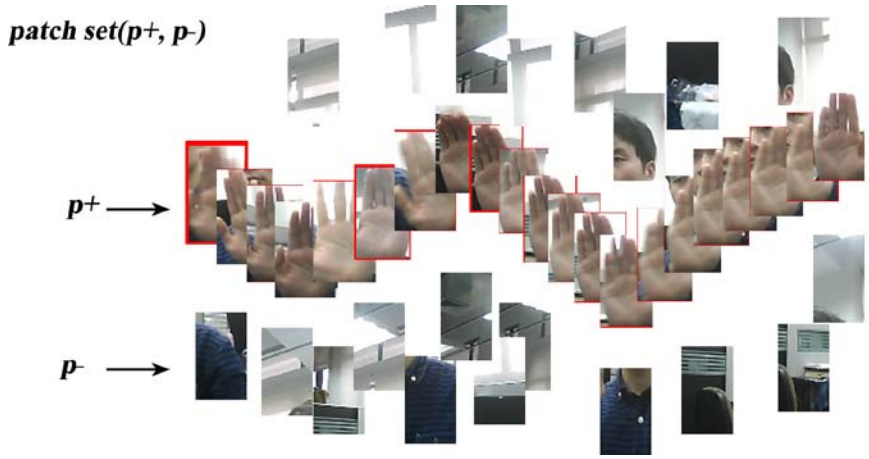

Fig. 6. An example of the patch set. Positive patches consist of correctly tracked palms in the history while negative patches are areas without the palm.

A patch set M includes both positive patch p^+ and negative patch p^-, as illustrated in Fig. 6. Next, the skin color model extracts the color information from the positive patches while the patch similarity calculation module mainly considers the texture information of both positive and negative patches.

Skin color model can work well or badly depending on the illumination condition. To initialize the skin color model in the beginning of tracking, a face detector is employed to locate the human face, which provides the required skin color information. If there is no face in the frames, empirical threshold values are used. The model is then used to reject the inputs (bounding boxes) if they violate the skin color information. This model can also increase *recall* by finding skin regions outside the input bounding box. During the tracking process, the model keeps updating itself by extracting skin color information from positive patches in the latest frame.

Patch similarity calculation module takes the outputs from the skin color model. As mentioned above, we define positive patches as those tracked correctly in the history and their background neighborhoods are regarded as negative patches (sampled from regions close to the trajectory). The similarity $S(p_i, p_j)$ between two patches is defined as (1).

$$S(p_i, p_j) = 0.5(NCC(p_i, p_j) + 1), \tag{1}$$

In (1), *NCC* is the Normalized Correlation Coefficient. Given an arbitrary patch *p* and a patch set *M* represented by (p_i^+ , p_i^-), the similarity between *p* and negative patches is defined as (2). A large value of S^- means that the patch shares similar appearance with the background. The similarity between *p* and the last 10% positive patches is given by (3), where *m* is the total number of patches. In general, it is more accurate when considering only the last 10% patches as older patches of the palm may differ a lot from the newly generated ones.

$$S^-(p, M) = \max_{p_i^- \in M} S(p, p_i^-), \tag{2}$$

$$S^+_{10\%}(p, M) = \max_{p_i^+ \in M \wedge i > \frac{9m}{10}} S(p, p_i^+), \tag{3}$$

The conservative similarity S^c ranging from 0 to 1 is defined in (4). A large value of S^c implies high confidence that the patch resembles appearance observed in the last 10% of the positive patches.

$$S^c = \frac{S^+_{10\%}}{S^+_{10\%} + S^-}, \tag{4}$$

The overlap rate O^θ is defined as the intersection area of two rectangles divided by their total area. According to (5), if the O^θ value between one of the detected bounding boxes (*n* is the number of boxes) and the tracked bounding box is greater than 50%, it implies a perfect tracking result and we can simply output the average of the two boxes. Otherwise, the integrator calculates the conservative similarity S^c between each input patch and the patch set *M*, and outputs the one with the maximum S^c.

$$B_f = \begin{cases} average_{i \in n}(B_d^i, B_t) & if \quad O_\theta > 0.5 \\ \max(S^c(p, M)) & else \end{cases} \tag{5}$$

If none of the tracker, the detector and the skin color model is able to generate a bounding box, the palm is declared as being invisible or a system failure is reported. In this situation, the skin color model will be re-initialized.

5 Experiments

We implemented the proposed tracking system in C++ (single thread), and tested it on a PC running Windows 7 32-bit version with Intel Core i5 CPU and 4GB RAM. Three video sequences named *Gao*, *Huang* and *Yang* were captured in normal lab working environments with different levels of difficulty. The environment in *Huang* was the simplest, without occlusion and other palms. In *Gao*, the tracked palm was occluded and out of frame sometimes. In *Yang*, there were a variety of challenging factors including occlusion, interference by other palms and discontinuity.

Qualitative studies. In Fig. 7, the top row of images demonstrates the performance of the offline detector in recognizing palm and the back of the hand with rotation and deformation. An example with naked arms (middle of the first row) was also included to show that the exposure of arm skin has no negative effect on our approach. The middle row is an ordinary tracking example without occlusion.

To demonstrate the advantages of the proposed tracking system on dealing with complicated background and topology changes, we compared our system with the standard TLD, as shown by the sequence in the bottom row in Fig. 7. The standard TLD performed poorly when an identical object (e.g., another palm) passed over the object under tracking. For example, the blue bounding circle of TLD was distracted by the appearance of the left palm and stayed with it, losing the real target (right palm). By contrast, due to the patch comparison module in use, our method consistently tracked the right palm (red bounding box) in all frames.

Fig. 7. An illustration of the detection (top) and tracking (middle) performance of our approach and the comparison with TLD (bottom)

Quantitative studies. In the three benchmark video sequences, the position of palm was manually annotated. The tracking performance was evaluated using the *precision* and *recall* measures: *precision* is the number of frames with correctly tracked palm divided by the number of frames with a bounding box; *recall* is the number of frames with correctly tracked palm divided by the number of frames that should be tracked according to the ground truth.

Table 1. Precision and recall of different tracking technqiues

Sequences	*Number of frames*	*Camshift*		*Standard TLD*		*Mittal's* [12]		*Our Method*	
		P	*R*	*P*	*R*	*P*	*R*	*P*	*R*
Huang	3368	0.42	0.37	0.81	0.63	0.98	0.60	0.94	0.85
Gao	4108	0.22	0.16	0.38	0.22	0.95	0.51	0.94	0.87
Yang	5884	0.10	0.08	0.22	0.14	0.97	0.45	0.92	0.80
Average	4453	0.25	0.20	0.47	0.33	0.967	0.52	0.93	0.84

In Table 1, the Mittal's method is a sophisticated hand segmentation method [12]. Its *precision* value was very high but it failed to detect the palm in many frames, leading to a low score in *recall*. The standard TLD performed well in simple situations but was not robust against occlusion or interferences.

As to time complexity, for a 640×480 frame, it generally takes our method 90 ms to 150 ms to process, depending on the number of detections produced by the offline detector. The average processing times on the three video sequences are presented in Table 2. The standard TLD often took significant amount of time in learning new samples especially at the beginning of tracking and the time reported was when the tracking was relatively stable. Note that the Mittal's method took much longer time compared to other tracking techniques.

Table 2. Average computation time of each frame (miliseconds)

Sequences	*Camshift*	*Standard TLD*	*Mittals'* [12]	*Our Method*
Huang	11	134	31856	99
Gao	9	118	31974	113
Yang	13	115	32196	135
Average	11	123	32008	116

6 Conclusions

In this paper, we presented a hybrid method for robust palm tracking in complex scenes. An offline palm detector with HOG and LBP descriptors based on cascade classifiers was trained to extract palm-like objects from the background. Furthermore, the traditional KLT tracker was extended with SURF features and an effective point matching strategy was employed to reduce false tracking. At last, a novel integrator containing the skin color model and the patch comparison module was proposed. It not only combines the offline detector and the tracker but also provides vital feedback information such as rotation degrees.

Experiments on three video sequences showed that our technique achieved better tracking performance in terms of *precision* and *recall*, compared to Camshift and standard TLD, especially in challenging environments. It also features reasonable computational complexity, making it suitable for conducting real-time tracking. A major direction of future work is to further improve the effectiveness of the offline detector so that it can better handle the deformation and rotation of palms.

References

1. Zhu, Y.M., Yang, Z.B. and Yuan, B.: Vision Based Hand Gesture Recognition. In: 2013 International Conference on Service Science, pp. 260-265 (2013)
2. Li, C. and Kitani, M.K.: Pixel-Level Hand Detection in Ego-Centric Videos. In: 2013 IEEE Conference on Computer Vision and Pattern Recognition, pp. 3570-3577 (2013)

3. Chen, Q., Georganas, N.D. and Petriu, E.M.: Real-Time Vision-Based Hand Gesture Recognition Using Haar-like Features. In: 2007 IEEE Instrumentation and Measurement Technology Conference, pp. 1-6 (2007)
4. Wu, Y. and Huang, T.S.: View-Independent Recognition of Hand Postures. In: 2000 IEEE Conference on Computer Vision and Pattern Recognition, vol.2, pp. 88-94 (2000)
5. Wu, Y., Liu, Q. and Huang, T.S.: An Adaptive Self-Organizing Color Segmentation Algorithm with Application to Robust Real-Time Human Hand Localization. In: 2000 Asian Conference on Computer Vision, pp. 1106-1110 (2000)
6. Dadgostar, F. and Sarrafzadeh, A.: An Adaptive Real-Time Skin Detector Based on Hue Thresholding: A Comparison on Two Motion Tracking Methods. Pattern Recognition Letters, 27(12), pp. 1342-1352 (2006)
7. Viola, P. and Jones, M.: Rapid Object Detection Using a Boosted Cascade of Simple Features. In: 2001 IEEE Conference on Computer Vision and Pattern Recognition, vol.1, pp. 511-518 (2001)
8. Ong, E.J. and Bowden, R.: A Boosted Classifier Tree for Hand Shape Detection. In: 6th Automatic Face and Gesture Recognition, pp. 889-894 (2004)
9. Karlinsky, L., Dinerstein, M. and Harari, D.: The Chains Model for Detecting Parts by Their Context. In: 2010 IEEE Conference on Computer Vision and Pattern Recognition, pp. 25-32 (2010)
10. Kumar, M.P., Zisserman, A. and Torr, P.H.: Efficient Discriminative Learning of Parts-Based Models. In: 12th International Conference on Computer Vision, pp. 552-559 (2009)
11. Arbelaez, P., Maire, M., Fowlkes, C. and Malik, J.: From Contours to Regions: An Empirical Evaluation. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2294-2301 (2009)
12. Mittal, A., Zisserman, A. and Torr, P.H.: Hand Detection Using Multiple Proposals. In: 22nd British Machine Vision Conference, pp. 75.1-75.11 (2011)
13. Kolsch, M. and Turk, M.: Fast 2D Hand Tracking with Flocks of Features and Multi-Cue Integration. In: IEEE Workshop on Real-Time Vision for Human-Computer Interaction, pp.158-165 (2004)
14. Bretzner, L., Laptev, I. and Lindeberg, T.: Hand Gesture Recognition Using Multi-Scale Color Features, Hierarchical Models and Particle Filtering. In: 5th IEEE International Conference on Automatic Face and Gesture Recognition, pp. 423-428 (2002)
15. Kalal, Z., Krystian, Mikolajczyk, K. and Matas, J.: Tracking-Learning-Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 34(7), pp. 1409-1422 (2012)
16. Rittscher, J., Tu, P.H. and Krahnstoever, N.: Simultaneous Estimation of Segmentation and Shape. In: 2005 IEEE Conference on Computer Vision and Pattern Recognition, vol.2, pp. 486-493 (2005)
17. http://www.iis.ee.ic.ac.uk/~tkkim/ges_db.htm
18. http://tutorial-haartraining.googlecode.com/svn/trunk/data/negatives/
19. Lucas, B.D. and Kanade, T.: An Iterative Image Registration Technique with an Application to Stereo Vision. In: 7th International Joint Conference on Artificial Intelligence, pp. 674-679 (1981)
20. Bay, H., Tuytelaars, T. and Van, G.L.: SURF: Speeded Up Robust Features. In: 9th European Conference on Computer Vision-Part I, pp. 404-417 (2006)
21. Kalal, Z., Mikolajczyk, K. and Matas, J.: Forward-Backward Error: Automatic Detection of Tracking Failures. In: 20th International Conference on Pattern Recognition, pp. 2756-2759 (2010)