Robust Hand Tracking in Realtime Using a Single Head-Mounted RGB Camera

Jan Hendrik Hammer[1] and Jürgen Beyerer[1,2]

[1] Karlsruhe Institute of Technology (KIT)
jan.hammer@kit.edu
[2] Fraunhofer Institute of Optronics, System Technologies and Image Exploitation, Karlsruhe, Germany
juergen.beyerer@iosb.fraunhofer.de

Abstract. In this paper novel 2D-hand tracking algorithms used in a system for hand gesture interaction are presented. New types of head-mounted Augmented-Reality devices offer the possibility to visualize digital content in the user's field of view. To interact with these head-mounted devices hand gestures are an intuitive modality. Generally, the recognition of hand gestures consists of two main steps: The first one is hand tracking and the second step gesture recognition. This paper concentrates on the first step: Hand tracking. Due to the wearing comfort of the glasses-like systems these only use a single camera to capture the field of view of the user. Therefore new algorithms for hand tracking without depth data are presented and compared to state-of-the-art algorithms by utilizing a thorough evaluation methodology for comparing trajectories.

1 Introduction

The development of mobile glasses-like Augmented-Reality (AR) devices is striding along. Different companies are working on these so called *high-tech glasses* or *cyberglasses*. Besides the capability of offering optical see-through AR the only head-mounted device (HMD) having eye tracking functionality is the *Interactive See-through HMD*[1]. The HMD has further been extended with a scene camera for capturing the user's field of view and serves as core device of the European project *ARtSENSE*[2]. The goal of *ARtSENSE* is to develop a system enhancing the experience of a museum visit by providing the visitor with digital content adapted to his personal interest [5]. Gaze is used to implicitly detect the visual attention [17] that heavily contributes to the decision of what is of interest to the visitor. Hand gesture recognition is used to detect intuitive and easy to learn wiping- and pointing-gestures making explicit interaction with the

[1] http://www.interactive-see-through-hmd.de/

[2] Augmented RealiTy Supported adaptive and personalized Experience in a museum based oN processing real-time Sensor Events, funded by the European Commission under the 7th Framework Program, Grant Agreement Number 270318.

M. Kurosu (Ed.): Human-Computer Interaction, Part IV, HCII 2013, LNCS 8007, pp. 252–261, 2013.

system and visual AR content possible. Basis of a gesture recognition is the hand tracking that we will focus on in this paper. Since depth sensors are too heavy to be attached to an HMD, we concentrate our work on 2D-hand tracking with a single RGB camera. The structure of this document is as follows: In Sec. 2 we present related work. Afterwards, in Sec. 3 we detail our developed methods and in Sec. 4 the used evaluation methodology. Before the conclusion and outlook in Secs. 6 we give information on the real-time capability in Sec. 4.

2 Related Work

In mobile applications the following challenges are prevalent: Lighting conditions may change constantly and – since the sensor is head-mounted – the background is not static. Both make the process of hand localization more difficult as in stationary applications, where simple frame differencing yields high quality hand segmentations [2] or trustable motion information [18]. That is why in mobile applications researchers use gloves [21], markers [11], accelerometers [16] or thermal cameras [1]. 3D-sensors are widely and successfully used for hand tracking [13]. The problem is that all these approaches are not appropriate for the given scenario. Nothing shall be attached to the hands of a museum visitor and the head-mounted device shall be lightweight in order to be as non-intrusive as possible. Pisharady et al. [15] detect hand postures against complex backgrounds, but their method is far away from being real-time capable. An application similar to ours is that of Kölsch and Turk [9].

3 Hand Tracking

In this section the hand tracking algorithms evaluated in this paper are described. Using only one visual camera we take into account two cues – as most recent procedures do: Skin color and motion information [20]. A rough overview of our system is depicted in Fig. 1. The input images are fed into a segmentation process which deals with the localization of the hand using skin color detection and motion information. Afterwards, the hand is tracked using different approaches including particle filters or Flocks of Features [9]. As result the tracking has a trajectory used e.g. for gesture recognition. In Sec. 3.1 we describe our approach for skin color detection and in Sec. 3.2 for motion computation. Then we detail our hand tracking methods in Sects. 3.3, 3.4 and 3.5.

3.1 Skin Color Detection

Skin color detection is well known especially by research topics like face detection from the last decades. The first question is the one for the color space to be used, the second one for the model representing the skin color distribution. For this purpose parametric models like single Gaussian distributions, Gaussian mixture models (GMM) or non-parametric histograms have been tested.

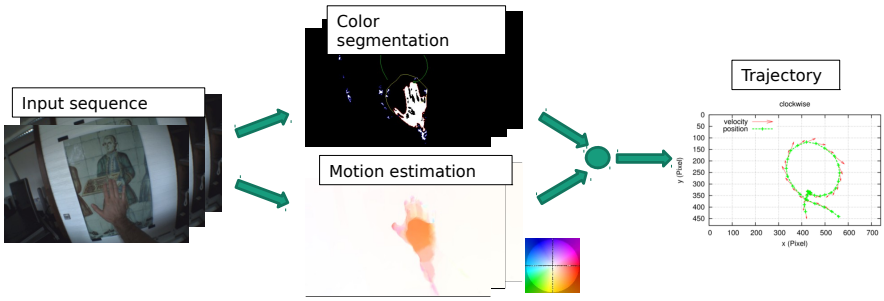

Fig. 1. General overview of our hand tracking algorithms and the used cues

Regarding the color space it has been shown in our experiments as well as the literature ([8], [14]) that 3D color spaces are the ones to choose compared to 2D color spaces. We decided to use RGB since there exists no preference whether RGB or HSV is more suitable.

Histograms do not make an assumption about the distribution of the color to be tracked. Gaussian distributions only work well if the tracked color is distributed normally. GMMs can adapt to not-normally distributed color distributions better, but the number of Gaussian distributions and the weighting factors have to be estimated using an Expectation Maximization step. In comparison, it can be said, histograms can adapt better to special color distributions, but their generalization capability is not as high as that of parametric models.

Independent of the type of model chosen for color representation, training data is needed to fill a histogram or compute the parameters of a Gaussian distribution. In some databases 1000 of images have been annotated according to "what is skin" and "what is not skin". Furthermore, GMMs have been generated out of this data. Models trained like this show a high generalization capability but tend to be not accurate enough in special situations [7]. We can confirm that circumstance for our case.

A Bayesian classifier along with a threshold [14] is used to produce a binary mask as seen in Fig. 1. For training of the models we use an image patch of a skin-colored region, which can be determined in a calibration phase. The binary mask is afterwards processed with morphological operations to reduce the noise of the segmentation.

3.2 Robust Motion Information

With only one camera and an inhomogeneous background it is not possible to perfectly segment the hands based on color information [1]. By using motion information it is possible to distinguish between different objects in the scene. Motion information is mostly produced by simple frame differencing in stationary applications [18]. Since this does not work for mobile applications, Koelsch and Turk[9] used KLT-features [10] to compute the optical flow for their Flocks of

Features tracker. We estimate motion by computing the optical flow between two images using the FlowLib [22] producing much more precise optical flow [3]. It contains the movement of each pixel to its position in the next frame. The right picture of Fig. 2 displays, how optical flow can be color-coded [3]. The direction of a motion vector is determined by the hue and its length by the saturation. As it can be seen in Fig. 2, the hand moves almost vertically upwards. At the same time, that part of the arm, which is at the lower right margin of the image, moves more to the right. One of the biggest problems concerning optical flow is that algorithms computing accurate flow fields in realtime are rarely available. In Secs. 3.4 and 3.5 is described how motion information is utilized for tracking.

Fig. 2. Left: Hand with overlayed motion vectors. Middle: Color-coded dense flow field. Right: Color wheel for color-coding flow vectors [3].

3.3 Region-Based Hand Tracking

Region-based hand tracking is a simple algorithm relying on skin color detection as described in Sec. 3.1. At first the biggest area of contiguous skin pixels is determined. The *center*-tracking approach only determines the mass value of this biggest skin-colored blob as visualized by the green dot in Fig. 3 on the left. The problem with *center*-tracking can already be seen in that image: If the arm is skin-colored, the hand position determined is distracted from the center of the back of the hand, so tracking becomes inaccurate or fails. To solve this problem we developed *tip*-tracking shown in Fig. 3 on the right. First, the pixel of the skin-colored blob with the smallest y-coordinate is determined. This smallest y-coordinate will be the y-coordinate of the final hand localization. Then, a special hand height corresponding to the dimensions of the hand in the image is chosen determining the height of the green bar shown in the image. The x-coordinate of the hand localization is then computed as the mass value of all blob-pixels under the green bar. This localization is done in each frame resulting in the trajectory of the hand.

3.4 Hand Tracking Using a Particle Filter

A particle filter [6] is a stochastic tracking algorithm. Three things have to be defined: the state, the motion model and the observation model. The simplest

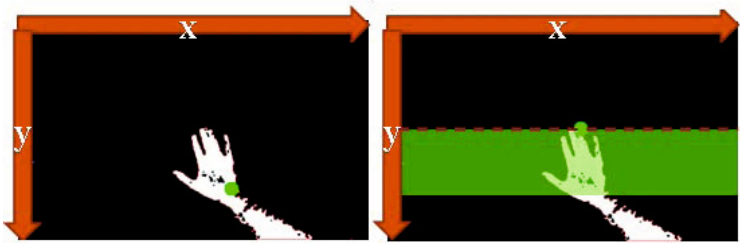

Fig. 3. Left: *center*-tracking. Right: *tip*-tracking.

state of a particle we tested contains one position and one velocity. Often only the skin color probability of the pixel at the particle's position is utilized as observation model, but we found out that using the sum of skin-colored pixels in a local square neighborhood leads to a much higher tracking robustness of the particle filter. This is called the *window*-observation model. As for the *center*-tracking of the previous section, the hand again can be lost because this observation model also computes high weights respectively quality for particles being on the skin-colored arm. Therefore we developed the *shape*-observation model, which is visualized in Fig. 4. The number of skin-colored pixels occluded by the green inner half circle increases and occluded by the yellow outer half circle decreases a particle's weight. Accordingly, the particle on the left in Fig. 4 has a higher weight than the particle on the right. Using this *shape*-observation model particles are prevented from staying on a skin-colored arm. The actual

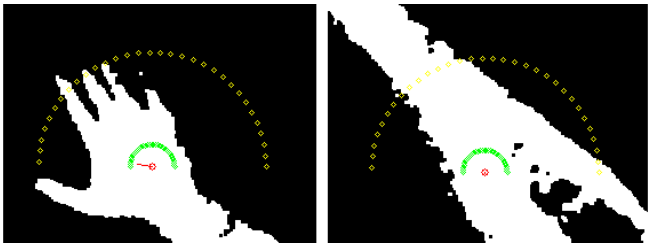

Fig. 4. Left: *shape*-particle with high weight. Right: *shape*-particle with low weight.

velocity as part of the state is determined by the difference vector of the actual and previous position. This motion model is below called *std*-motion model. To further improve the motion of particles we make use of the displacement vectors computed by the FlowLib (cf. Sec. 3.2). This is called the *flow*-motion model.

3.5 Adapted Flocks of Features Tracker

Flocks of Features tracking [9] is one of the state-of-the-art algorithms for hand tracking. This algorithm uses a Viola-and-Jones like detector [19] for the first

localization of the hand. Tracking goes on by using skin color and motion information. Therefore features are placed on the initially found location of the hand and are coupled in a flock by using specific conditions imposed on the feature positions [9]. Motion information is estimated by using KLT-features [10] and skin color probabilities are only considered at the corresponding feature locations.

The Flocks of Features algorithm has been adapted as follows: First, the weight computation of the features has been changed. Originally this computation only considers the skin color probability at the feature's position (*point*-mode). Our version uses all skin color probabilities in a local square neighborhood (*window*-mode) similar to the *window*-observation model of the particle filter described above. Second, instead of KLT-features we use optical flow estimated by the FlowLib [22].

4 Evaluation of Hand Tracking

No common benchmark for hand tracker comparison exists. Because of that we recorded several wiping gestures under different lighting conditions and implemented an evaluation methodology based on the metrics for trajectory comparison found in Needham and Boyle [12]. In Sec. 4.1 the evaluation methodology is described and in Sec. 4.2 the evaluation results are summarized.

4.1 Evaluation Methodology

Our evaluation methodology is based on the metrics for trajectory comparison of Needham and Boyle [12]. Using these makes a thorough evaluation of tracking results possible, since not only detection rates, like the hit rate, false alarm rate or precision of the detection results can be compared. Statistical measures as the median or mean of the deviations between two trajectories allow for precise conclusions. The ground truth trajectories were labeled manually. When labeling is repeated several times, the same person produces similar but slightly different trajectories for the same sequence. Therefore we computed the average distance of manually labeled ground truth trajectories of the same video. The result is an average distance of around five pixels between such trajectories. If an algorithm produces this result, the tracking can be considered as very accurate.

If two different tracking algorithms are compared, it has to be regarded that some track the center of the back of the hand and some the most upper skin pixel (cf. Sec. 3.3). The resulting trajectories are almost the same but shifted by a constant displacement. Hence, one must compensate for this offset by shifting one of the sequences according to this average displacement. After that, scores like the average distance become meaningful. The same yields for the recognition of unreferenced gestures, where not the exact positions are required but the relative trajectories. In this case, since a trajectory can also be seen as sequence of velocities or displacement vectors, the absolute difference of corresponding velocities of compared trajectories should be as small as possible. In the evaluation presented below we consider these velocity deviations as quality metric.

Our benchmark consist of four videos recorded at 25 fps and a resolution of 752×480 pixels. The videos were recorded under dark and light lighting conditions, in front of a complex background. Each contains sixteen wiping-gestures performed by one person. The sequences were recorded for two persons performing gestures with different speeds resulting in more than 5600 frames in total with one hand visible in approximately 50% of the time. The tracking methods were evaluated on these sequences of gestures, in which the hand is entering the field of view of the camera, performing the gesture and leaving the field of view for every gesture. This adds additional difficulty because this entering and leaving must be detected correctly.

4.2 Evaluation Results

In this section we present a comparison of the algorithms described above. All of these show good tracking results on single gestures, but some fail on sequences of gestures. Our evaluation of the Flocks of Features variants revealed improved tracking robustness when using *window*-mode and FlowLib-motion. Still, our best Flocks of Features implementation sometimes fails in detecting the hand leaving the image. As consequence the method starts tracking the background. The adjustment of the Camshift algorithm for properly detecting hands leaving the image is difficult, but Camshift handles this difficulty much better and produces only a few false positives. Camshift further has to be carefully adapted to the hand size and image resolution, which is the same for the *tip*-tracking. Additionally, both solely rely on skin color detection. This is their biggest disadvantage, because if skin color detection fails, they fail tracking. Below we present the results on one of videos containing a sequence of gestures. The following methods have been compared:

1. Region-based hand localization with *tip*-tracking (cf. Sec. 3.3)
2. Particle filtering with 500 and 5000 particles (cf. Sec. 3.4):
   (a) *std*-motion and *shape*-observation model
   (b) *flow*-motion and *window*-observation model
3. Camshift [4]
4. Flocks of Features tracking with *window*-observation mode and Flow-Lib-motion (cf. Sec. 3.5)

In Fig. 5 we see the visualized velocity deviations. It can be seen that all methods can handle this long video with good accuracy. The region-based method *tip*-tracking works very good with a hit rate of above 95 % and a false alarm rate of below 2 %. The Camshift algorithm produces also good accuracy but has slightly worse hit rates of below 90%. The particle filter variants produce hit rates of above 90 % and false alarm rates below 2 %. Generally, it has to be underlined that the *window*-observation mode in combination with the *std*-motion model fails tracking but in combination with the *flow*-motion tracking succeeds. The *shape*-observation model even shows similar accuracy without using optical flow. The particle filter variants with 5000 particles produce more accurate results

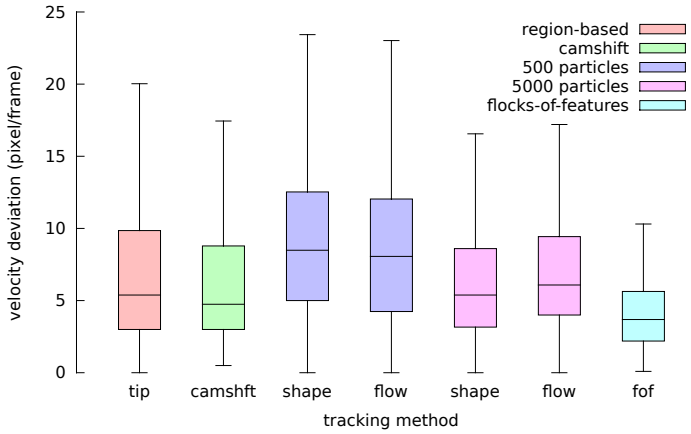

Fig. 5. Comparison of best tracking methods on all gestures with dark lighting conditions

than with 500 particles. The Flocks of Features variant shows the best accuracy on this sequence of gestures but as already mentioned above fails tracking on others videos containing sequences of gestures.

To conclude the evaluation, the developed algorithms have been extensively tested and compared on real sequences using the described evaluation methodology. The results on one sequence of gestures have been presented, which mainly reflect the trackers' overall performance. We could show that our novel observation models incorporated into the particle filter and our adaption of the Flocks of Features tracker as well as the usage of motion information of much higher quality, result in higher tracking accuracy and less tracking failure. However, future tests on other videos with more challenging lighting conditions have to be conducted, because changing lighting conditions directly affect the skin color segmentation. Therefore adaptive skin color models have to be utilized and implemented in all approaches.

5 Realtime Capability

Since the hand tracker is used in an interactive system, it must be realtime capable. *Tip*-tracking reaches 143 fps and Camshift 130 fps. In Sec. 3.2 we mentioned already that robust optical flow unfortunately has a high computational load. The FlowLib in its standard configuration is not realtime capable on the tested image resolution even on modern graphics devices with more than 1000 cores. The *shape*-variant of the particle filter using 500 particles runs with 95 fps. Using 5000 particles reduces the frame rate to 18 fps. The Flocks of Features variant using KLT-features and *window*-observation mode runs with 55 fps.

6 Conclusion and Outlook

To sum up, we have shown new 2D-hand tracking algorithms with increased tracking accuracy and robustness against tracking failure compared to standard approaches as Flocks of Features, Camshift or standard particle filtering. This was demonstrated by realizing a benchmark and an evaluation methodology with different metrics for a thorough trajectory comparison. In the future our focus lies on adaptive skin color models to be able to handle varying lighting conditions and the estimation and fusion of high quality motion information in realtime. To this purpose the benchmark is going to be extended with additional videos containing different people performing gestures in front of different backgrounds under various lighting conditions.

References

1. Appenrodt, J., Al-Hamadi, A., Elmezain, M., Michaelis, B.: Data gathering for gesture recognition systems based on mono color-, stereo color- and thermal cameras. In: Lee, Y.-h., Kim, T.-h., Fang, W.-c., Ślęzak, D. (eds.) FGIT 2009. LNCS, vol. 5899, pp. 78–86. Springer, Heidelberg (2009)
2. Bader, T., Räpple, R., Beyerer, J.: Fast invariant contour-based classification of hand symbols for hci. In: Jiang, X., Petkov, N. (eds.) CAIP 2009. LNCS, vol. 5702, pp. 689–696. Springer, Heidelberg (2009)
3. Baker, S., Scharstein, D., Lewis, J., Roth, S., Black, M., Szeliski, R.: A database and evaluation methodology for optical flow. International Journal of Computer Vision 92, 1–31 (2011)
4. Bradski, G.R.: Real time face and object tracking as a component of a perceptual user interface. In: Proceedings of the 4th IEEE Workshop on Applications of Computer Vision (WACV 1998). IEEE Computer Society, Washington, DC (1998)
5. Damala, A., Stojanovic, N., Schuchert, T., Moragues, J., Cabrera, A., Gilleade, K.: Adaptive augmented reality for cultural heritage: Artsense project. In: Ioannides, M., Fritsch, D., Leissner, J., Davies, R., Remondino, F., Caffo, R. (eds.) EuroMed 2012. LNCS, vol. 7616, pp. 746–755. Springer, Heidelberg (2012)
6. Isard, M., Blake, A.: Condensationconditional density propagation for visual tracking. International Journal of Computer Vision 29, 5–28 (1998)
7. Jones, M.J., Rehg, J.M.: Statistical color models with application to skin detection. International Journal of Computer Vision, 274–280 (1999)
8. Kakumanu, P., Makrogiannis, S., Bourbakis, N.: A survey of skin-color modeling and detection methods. Pattern Recognition 40(3), 1106–1122 (2007)
9. Kölsch, M., Turk, M.: Fast 2d hand tracking with flocks of features and multicue integration. In: CVPRW 2004 Conference on Computer Vision and Pattern Recognition Workshop, p. 158 (June 2004)
10. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of the 7th International Joint Conference on Artificial Intelligence, IJCAI 1981, vol. 2, pp. 674–679. Morgan Kaufmann Publishers Inc., San Francisco (1981)
11. Mistry, P., Maes, P.: Sixthsense: a wearable gestural interface. In: ACM SIGGRAPH ASIA 2009 Sketches, pp. 11:1–11:1. ACM, New York (2009)

12. Needham, C.J., Boyle, R.D.: Performance evaluation metrics and statistics for positional tracker evaluation. In: Crowley, J.L., Piater, J.H., Vincze, M., Paletta, L. (eds.) ICVS 2003. LNCS, vol. 2626, pp. 278–289. Springer, Heidelberg (2003)
13. Oikonomidis, I.: Tracking the articulated motion of two strongly interacting hands. In: Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2012, pp. 1862–1869. IEEE Computer Society, Washington, DC (2012)
14. Phung, S., Bouzerdoum, A.S., Chai, D.S.: Skin segmentation using color pixel classification: analysis and comparison. IEEE Transactions on Pattern Analysis and Machine Intelligence 27(1), 148–154 (2005)
15. Pisharady, P., Vadakkepat, P., Loh, A.: Attention based detection and recognition of hand postures against complex backgrounds. International Journal of Computer Vision 101, 403–419 (2013)
16. Prisacariu, V., Reid, I.: Robust 3d hand tracking for human computer interaction. In: 2011 IEEE International Conference on Automatic Face Gesture Recognition and Workshops (FG 2011), pp. 368–375 (March 2011)
17. Schuchert, T., Voth, S., Baumgarten, J.: Sensing visual attention using an interactive bidirectional hmd. In: Proceedings of the 4th Workshop on Eye Gaze in Intelligent Human Machine Interaction, Gaze-In 2012, pp. 16:1–16:3. ACM, New York (2012)
18. Spruyt, V., Ledda, A., Geerts, S.: Real-time multi-colourspace hand segmentation. In: 2010 17th IEEE International Conference on Image Processing (ICIP), pp. 3117–3120 (September 2010)
19. Viola, P., Jones, M.: Robust real-time object detection. International Journal of Computer Vision (2001)
20. Wachs, J.P., Kölsch, M., Stern, H., Edan, Y.: Vision-based hand-gesture applications. Commun. ACM 54, 60–71 (2011)
21. Wang, R.Y., Popović, J.: Real-time hand-tracking with a color glove. ACM Trans. Graph. 63, 1–63 (2009)
22. Werlberger, M., Pock, T., Bischof, H.: Motion estimation with non-local total variation regularization. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), San Francisco, CA, USA (June 2010)