Real Time Face Tracking and Pose Estimation Using an Adaptive Correlation Filter for Human-Robot Interaction

Vo Duc My and Andreas Zell

***Abstract*— In this paper, we present a real time algorithm for mobile robots to track human faces and estimate face poses accurately, even when humans move freely and far away from the camera or go through different illumination conditions in uncontrolled environments. We combine the algorithm of an adaptive correlation filter with a Viola-Jones object detection to track the face as well as the facial features including the two external eye corners and the nose. These facial features provide geometric cues to estimate the face pose robustly. In our method, the depth information from a Microsoft Kinect camera is used to estimate the face size and improve the performance of tracking facial features. Our method is shown to be robust and fast in uncontrolled environments.**

I. INTRODUCTION

Both face tracking and face pose estimation play key roles for human-robot interaction which can be used as essential preprocessing steps for robust face recognition or facial expression recognition.

Tracking faces in uncontrolled environments still remains a challenging task because the face as well as the background changes quickly over time and the face often moves through different illumination conditions. Moreover, previous tracking methods have significant drift due to sudden changes of the face and background. In this paper, we propose an algorithm of tracking faces based on the combination of an adaptive correlation filter [1] and a Viola-Jones face detection [2]. This combination utilizes the advantages of adaptive correlation filters to adapt to face changes of rotation, occlusion and scales as well as adapt to complex changes of background and illumination. Its computational cost is only 7 ms per frame. Furthermore, it can also remove drift effectively by detecting the face and correcting its position after a period of time. In addition, we utilize the depth information from the Microsoft Kinect camera to estimate the corresponding size of the face. Our tracker successfully runs on a mobile robot when both the humans and the robot move and rotate quickly with different angles and directions.

The problem of face pose estimation for human-robot interaction also has some significant challenges. First, the resolution of faces is very low when the humans move far away from the robot. Most existing methods are accurate for estimating poses in high-resolution face images, but their performance is much worse or they completely fail to estimate poses in low-resolution face images. Second, while both the humans and the robot move in complicated backgrounds and under different illumination conditions, facial features change very quickly and the face also changes in a variety of poses. Additionally, when the face changes by large angles of rotation, some parts of the face are visible while the rest is occluded. In order to find a robust method of face pose estimation for human-robot interaction, we track some key facial features, including the two external eye corners and the nose. These features provide geometric cues to estimate precisely the yaw angle and the roll angle of the face which are important for the improvement of uncontrolled face recognition. Similar to our method of face tracking, we combine an adaptive correlation filter and a Viola-Jones object detection to track these features which are robust to face rotation, face deformation, occlusion and complicated illumination. As a result, the face pose is also estimated efficiently based on some geometric computation among the face features, such as shown in Figure 1. Our feature based method of face pose estimation is shown to be robust running on a mobile robot in uncontrolled illuminations and environments while our whole system is operating at a speed of 21 ms per frame.

The remaining part of this paper is organized as follows. In Section II we present the most relevant algorithms of face tracking and face pose estimation which motivated our research. In Section III, our method is presented in detail. In Section IV the experimental results obtained from databases are presented and in Section V we conclude this paper and

V. D. My is with the Chair of Cognitive Systems, headed by Prof. A. Zell, Computer Science Department, University of Tübingen, Sand 1, D-72076 Tübingen, Germany {duc-my.vo, andreas.zell@uni-tuebingen.de}

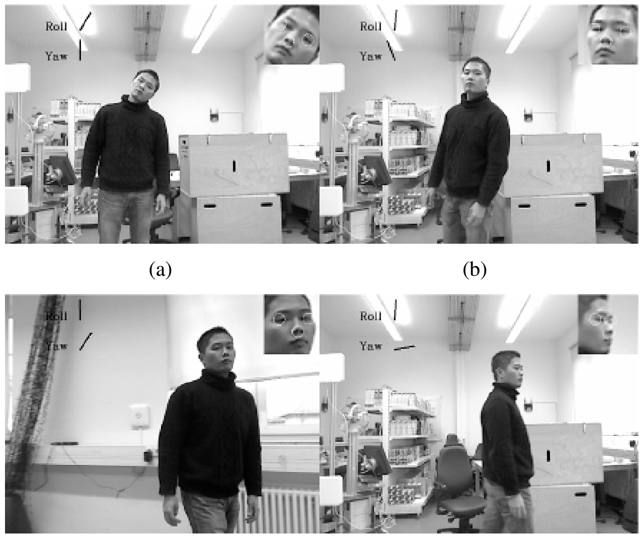


Fig. 1. Examples of our face tracking and pose estimation on a moving mobile robot. The white circles indicate the locations of facial features

mention our future work.

II. Related Work

A. Face tracking

Although there are many publications of face tracking, it still remains a huge challenge, because of changes of face appearance, occlusions, rotation, complex changes of the surrounding background, partial or full changing illumination conditions. Most existing methods build a face model for tracking which does not adapt to the large variation of face pose as well as illumination. These algorithms include some typical methods of correlation-based visual tracking [3], condensation algorithm [4] or the method using intensity gradients and color histograms [5]. Additionally, recent research focuses on online learning methods to handle the complex appearance variation of human faces. Some examples of these algorithms include incremental learning [6], online random forests [7], online multiple instance learning [8] and visual tracking using L1 minimization [9]. Despite their high efficiencies, most of the online learning methods fail due to the drift problem. Moreover, computational requirements of these methods are usually huge which is not suitable for real time tasks of the robot. Although not successfully eliminating drifts, the Minimum Output Sum of Squared Error (MOSSE) filter attracts the attention of researchers. It adapts to wide variations of object poses, illumination, and is able to run at high frame rates. In order to eliminate the drift problem we combine the algorithm of a MOSSE filter with a Viola-Jones object detection. While the MOSSE filter is able to track the face during a long period of time, the face detector is responsible for correcting the face position in a constant period of time if it detects exactly the face location.

B. Face pose estimation

During the past 20 years, there has been a huge number of papers in the field of face pose estimation [10]. Among them, geometric methods have proven to be fast, simple and suitable for real time applications. Gee *et al.* [11] presented two simple methods based on detecting and tracking some facial features such as the far corners of the mouth and eyes, and the tip of the nose. Face model ratios were built to compare with the real face ratios seen in images to calculate the face normal. This method is very fast and accurate when the human face is near the camera. Horprasert *et al.* [12] estimated the face pose using five points, the inner and outer corners of each eye, and the tip of the nose. The angle yaw is estimated based on the difference of the distance between the left and right eye. The roll angle is calculated by the arctangent of the slope between two eyes. The pitch angle can be found easily based on the distance between the nose tip and the eye lines. All these geometric methods have some common drawbacks. First, face features in their methods must be detected and tracked very precisely, which is not easy when the humans move far away from the camera and the face resolution is very low. Furthermore, some facial features can be missing when the face changes by large angles of rotation. Therefore these methods can be much worse or can completely fail. In addition, the depth information is used to improve the accuracy of head pose estimation. Newman *et al.* [13] combined techniques of stereo matching and feature matching to track the three dimensional positions of six face features and maps positions of these features to a head model for estimating the face pose. This method is able to estimate the face pose when the humans stand near the camera. Fanelli *et al.* [14] applied the method of discriminative random regression forests in the depth image of a Microsoft Kinect camera to estimate location and orientation of the head. Their system is able to run in real time and is relatively accurate while the head changes with a large variation of poses or is occluded partly. But this method fails when the humans move farther than 1 meter from the Microsoft Kinect camera. Cascia *et al.* [15] used a manually initialized cylindrical head model and applied recursive least squares optimization to track the head. Xiao *et al.* [16] used dynamic templates to recreate the face model. The drawback of these methods is the requirement of an accurate initialization of the face location. Additionally, these methods are only applicable for near-field images and are very time-consuming.

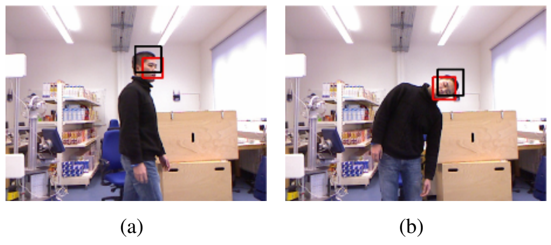


Fig. 2. Examples of face tracking through poses. Our face tracker is marked by the red rectangle and the original MOSSE filter is marked by the black rectangle.

III. Approach

A. Face detection

Face detection is an important component in our algorithm which allows mobile robots to quickly locate the position of the face in the initial step and relocate it if our system loses tracking. For this paper, we use the face detection method mentioned in our previous work [17], which is very fast and accurate and runs in real time. By using geometric constraints, navigation and depth-based skin segmentation, the average processing time of this method is only around 8 ms. It is also more reliable than the unmodified OpenCV face detector. Our face detection involves five basic steps: First, we collect data from a small set of sampling points which span both the color image and depth image. This step is to reduce computational costs. Second, we evaluate these sampling points under constraints of geometry and navigation information to remove the background. Third, we apply a robust technique of skin detection around filtered sampling points. In the fourth step, a method of depth-based skin segmentation is used to find the potential face regions and estimate the face size. In the last step, we apply the Viola-Jones method to detect the face. We also use the technique mentioned in [17] to limit the range of scales to

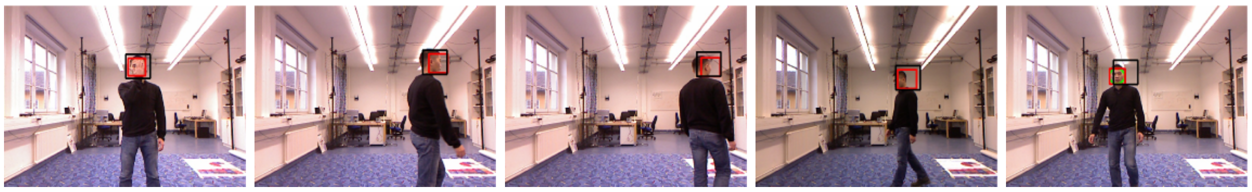

(a) Frame 885 (b) Frame 1295 (c) Frame 1313 (d) Frame 1381 (e) Frame 1410

Fig. 3. Examples of face tracking through occlusion and drift. We compare our face tracker, which is marked by the red rectangle, and the original MOSSE filter, which is marked by the black rectangle. 3(a): The face is occluded. 3(b), 3(c), 3(d): The drift problem occurs when the human turns around. 3(e): Our face tracker is recovered while the original MOSSE filter mostly fails when the face reappears.

detect the face for the next steps. The average width of the human face is about 0.15 meters. We denote s_f as the average width of faces; d is the distance from those to the camera and f is the focal length of depth camera. Then we use the following formula to estimate the size of faces in images

$$s_f = \frac{0.15f}{d} \tag{1}$$

B. Face tracking

The tracking algorithm we propose is based on the combination of a tracking method using the MOSSE filter and a Viola-Jones face detection. By using the Microsoft Kinect camera, our algorithm is able not only to track the position of a face but also to estimate its corresponding size based on the formula (1). The face position, which is located by the face detector, is the initial position of the face tracker.

The filter is initialized by training eight randomly affine transformed versions (f_i) of a search window with a fixed size of 64×64 in the initial position. Training outputs (g_i) are generated from 2D Gaussian images, of which peaks are in their centers. We denote the 2D Fourier transform of a training image f_i as F_i, of the filter h as H, and of a training output g_i as G_i. In the initial position, the filter H can be found based on the following formula

$$H^* = \frac{1}{N} \sum_i \frac{G_i \odot F_i^*}{F_i \odot F_i^*} \tag{2}$$

where $*$ indicates the complex conjugate, $\odot$ is the operation of element-wise multiplication and N is the number of the training images.

In the next frames, the face is tracked by the search window in the center. By correlating the filter over the search window, we can find the new position of the face in the current frame, which is the area corresponding to the maximum value in the correlation output. In addition, every search image is multiplied by a log function to reduce the effect of illumination. Then it is multiplied by a cosine window to increase the effect of pixels near the center of the search window. In order to compute the correlation operation, all the search images and filter are transformed to Fourier space by using a Fast Fourier Transform. We denote the 2D Fourier transform of a search image f as F = F(f). The correlation output G takes the form

$$G = F \odot H^* \tag{3}$$

In every 30 frames, the Viola-Jones face detector is applied for correcting the positions of faces. The search window is scanned with the scale estimated based on the formula (1) while the depth information in the face center is known. It significantly reduces the processing time of face detection to an average of 3 ms. In the case that the face detector finds a face, its position is considered as the new tracking position instead of that predicted by the MOSSE filter. Therefore the drift problem can be solved efficiently, such as shown in Figure 3.

In the tracking position, the MOSSE tracker must be updated online in order to quickly adapt to the appearance changes of the face. To update online in frame i the MOSSE filter is computed as follows

$$H_i^* = \frac{A_i}{B_i} \tag{4}$$

$$A_i = \eta G_i \odot F_i^* + (1-\eta) A_{i-1} \tag{5}$$

$$B_i = \eta F_i \odot F_i^* + (1-\eta) B_{i-1} \tag{6}$$

where η is the learning rate, H_i^* consists of the numerator A_i and the denominator B_i, F_i and G_i are the 2D Fourier transforms of the training image f_i and of the training output g_i, respectively. The MOSSE filter combines the computation of previous frames and the current frame to adapt quickly and robustly to the changes of face pose, rotation, deformation and illumination. Furthermore, it is possible to detect the failure of tracking and to stop updating the face appearance by measuring peak strength called the peak to sidelobe ratio (PSR) [1]. As a result, the face tracking is possibly recovered when the face reappears.

C. Facial feature tracking

After successfully tracking the face and estimating the face size using the formula (1), we can resize and copy the face to a second image called facial feature image. In the new image, the size of the face is fixed at 120×120 pixels; therefore, the sizes of facial features including eyes and nose are easily estimated. As a result, we can detect and track the facial features in the same way as detecting and tracking the face. There are three crucial features which are necessary to be tracked: the two external eye corners and the nose. These features provide geometric cues to estimate the facial pose across a wide variety of face rotations and scales. Figure 4 shows the result of facial feature tracking in which white circles indicate their locations. In this figure the yaw and roll angles of the human face can be found simply and quickly based on these features. Basically, the new facial feature positions are efficiently tracked based on the combination

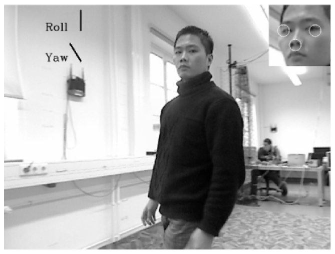


Fig. 4. Example of facial feature tracking. White circles indicate locations of facial features.

of the MOSSE filter and a Viola-Jones object detection. The two basic steps of prediction and online update for tracking facial features are the same as those used above for tracking faces.

D. *Face pose estimation*

Based on the tracked facial features we can estimate the yaw angle of the face pose denoted as γ and the roll angle of the face pose denoted as α. In our paper, we did not estimate the pitch angle of the face because it is not an easy task when the humans move far away from the camera and the face resolution is low. The pitch angle is usually estimated with large errors. Estimating the pitch angle of a face in uncontrolled environments is our future work.

In order to estimate the roll angle of the face pose, α, we calculate the angle of the line joining the two external eye corners, which is the arctangent of the slope between these corners. We denote the coordinates of the left external eye corner and the right external eye corner as (x_1, y_1) and (x_2, y_2), respectively. The roll angle of the face pose can be calculated as follows

$$\alpha = tan^{-1}\left[\frac{y_2 - y_1}{x_2 - x_1}\right] \tag{7}$$

In addition, we apply a simple technique to estimate the yaw angle of the face based on the relative positions of three tracked points. We denote the distance between the left external eye corner and the nose as L, the distance between the right external eye corner and the nose as R. Because the size of the face is fixed in the feature image, we can estimate the yaw angle by a function of R and L as follows

$$\gamma = \begin{cases} \frac{aR - bL}{L} & if\ R > L \\ \frac{-aL + bR}{R} & otherwise \end{cases} \tag{8}$$

where a = 33.3, b = 33.3.

IV. Experimental Setup

For evaluating the accuracy of our face tracking we used a collection of four log files recorded from a Microsoft Kinect camera mounted on our mobile robot SCITOS G5. Every log file includes color and depth images and is recorded from two to three minutes at 30 frames per second. We compared our method with the original MOSSE filter in these challenging log files in which the humans move freely in front of the Microsoft Kinect camera and rotate the face quickly in a wide variety of the poses in an uncontrolled environment. Figure

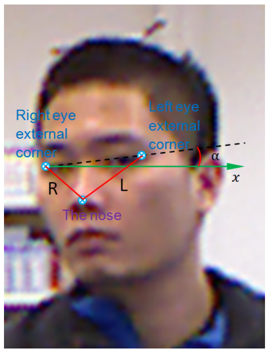


Fig. 5. Ilustration of geometric constraints based on three facial features which are the two the external eye corners and the nose.

2 and Figure 3 are examples extracted from our experiments in which our method outperforms the original MOSSE filter.

To evaluate the accuracy and the speed of our face pose estimation technique, we use two data collections. The first one is the Tuebingen dataset which consists of 15 log files of 15 people spanning around two minutes. Each of these log files recorded color and depth images at 30 frames per second at a resolution of 640×480 pixels. Since our goal is to evaluate the performance of face pose estimation in uncontrolled environments, selected log files must contain faces in a wide variety of poses: looking left or right, up or down, or tilting left or right while the humans are moving freely in front of the camera under different illumination conditions. To measure the ground truth data, we used an external tracking system, "Optitrack" by Natural Points including 12 infrared cameras. This tracking system is able to measure six degrees of freedom of the face. Because the working space of the tracking system is limited, the human has to sit on a chair and move freely in a range from 1 to 3 meters away from the camera while the face is allowed to change the pose in different angles. For evaluation, we only focused on the yaw and roll angles which are very important for the application of uncontrolled face recognition. Figure 6 shows some sample images extracted from our dataset.

In order to compare our method with other state-of-the-art methods, we used the Boston University dataset (www.cs.bu.edu/groups/ivc/HeadTracking) with the associated ground truth which was measured by a Flock of Birds 3D tracker. Each video contains 200 frames and has a resolution of 320×240. In this dataset, we can not use depth information to estimate the scale of the face. But human faces do not change the scale too much; therefore, facial features are still tracked well. We compared our results of accuracy and processing time with results of the methods proposed in [15] and [16].

We used a PC with a 2.4 GHz Intel Core 2 Duo CPU to test our algorithms in these experiments.

A. *Evaluation of Face Tracking*

We evaluated the tracking quality of our method and the MOSSE filter in four challenging videos. The tracking output was manually labeled as good tracking, bad tracking in which the tracking bounding box overlaps below 50 % of the ground truth bounding box, and a lost track. Generally,

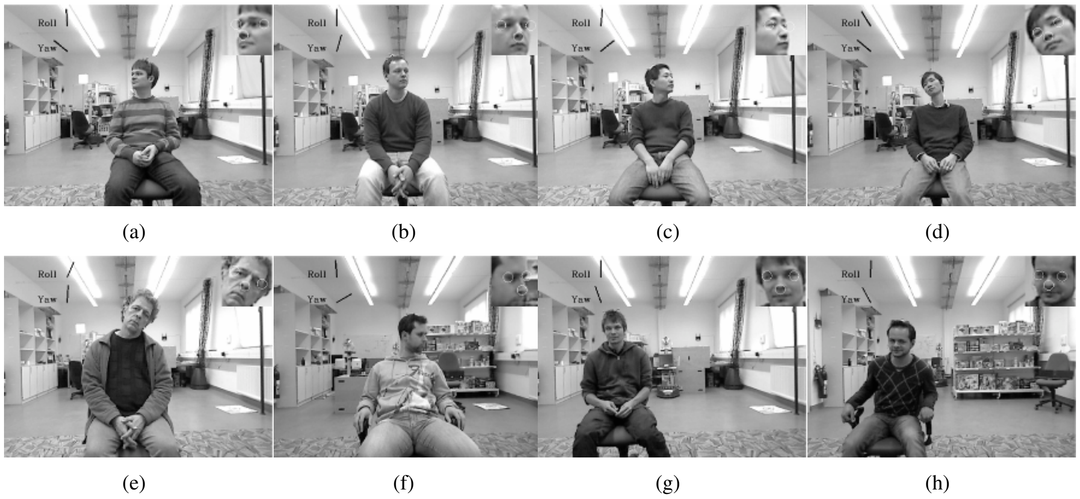


Fig. 6. Sample images from the Tuebingen dataset. The human face can rotate in a wide variety of poses and the humans can move to the left, right side or move backward and forward.

TABLE I
COMPARISON OF TRACKING QUALITY BETWEEN OUR METHOD AND THE MOSSE FILTER METHOD

video	MOSSE filter			Our method		
	good	bad	lost	good	bad	lost
1	26.9 %	73.1 %	0 %	90.6 %	9.4 %	0 %
2	56.8 %	43.2 %	0 %	90.9 %	9.1 %	0 %
3	100 %	0 %	0 %	100 %	0 %	0 %
4	32.4 %	0.4 %	67.2 %	100 %	0 %	0 %

TABLE II
MEAN ABSOLUTE ERROR (MAE) AND STANDARD DEVIATION (STD) OF THE ERRORS OF OUR SYSTEM ON THE TUEBINGEN DATASET.

	MAE	Std
Yaw (deg)	7.97	6.89
Roll (deg)	4.85	4.49

the MOSSE filter is able to track the face well unless the face pose changes by very large angles of rotation due to the drift problem. And because the MOSSE filter is not able to recover automatically after drifts, it completely fails to track the face in some of our testing videos.

Table I shows a distinguished difference between our tracker and the MOSSE tracker. It shows that our tracker is able to track faces longer and more accurately than the MOSSE tracker because it can correct the tracking position when the tracker drifts. The processing time of our tracker is about 7 ms. Our tracker adapts to drastic changes of illumination, background as well as face pose.

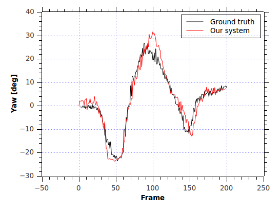
B. *Evaluation of Face Pose Estimation*

We evaluated the system in two experiments. First of all, we used our dataset for evaluating the quality of face pose estimation in uncontrolled environments. Table II shows the mean absolute error and standard deviation of the errors which are measured for our system of face pose estimation in the Tuebingen dataset. As can be seen in this table, our system estimates the face pose robustly while the humans are moving freely in 3D translation and 3D rotation at near or far distances. When the face moves far away from the camera, the face image is much more noisy and blurred. Moreover, changing illumination conditions also produce a lot of noise on the face. But our system can track facial features quite well in such blurred images; therefore the face pose is still estimated relatively reliably in such conditions. With the mean absolute error and standard deviation values of the yaw angle and the roll angle as shown in Table II, our system can meet the requirements of many applications, such as a reliable preprocessing step of uncontrolled face recognition in surveillance systems or on mobile robots.

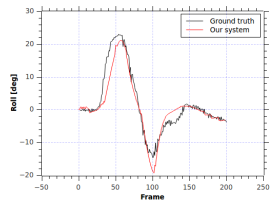
Some results of our system are plotted in Figure 8, which show the estimated yaw and roll angles compared to ground truth. The curve of the estimated yaw angle is quite consistent with the curve of ground truth. In this figure, the errors which are in the estimation of the roll angle mostly result from the deformation of the face when it rotates in 3D space at a far distance. In general, the result of this estimation is robust for real time application of mobile robots.

In addition, our system of face tracking and pose estimation can run at 21 milliseconds per frame on average which meets the real time requirement of a mobile robot.

In controlled environments with uniform illumination conditions, our method is able to track the face more accurately. Figure 7 shows the roll and yaw angles which are estimated by our method and are compared with the ground truth. The curve of our estimation is quite consistent with the curve of ground truth. Additionally, Table III shows the comparison of the accuracy and processing time between our method and state-of-the-art methods [15] and [16]. As can be seen in Table III the accuracy of our proposed approach is slightly worse than two others but it is much faster and can run in real time. While the methods proposed in [15] and [16] run at a speed of 15 frames per second in images which have the resolution of 320×240, our system is able to run at a speed of 50 frames per second even when the resolution of the image is 640×480. In addition, the methods proposed by Xiao *et al.* and La Cascia *et al.* develop a robust cylindrical model which must be initialized and recovered in near distance. This

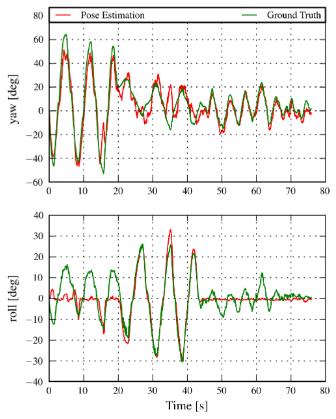


(a) Yaw



(b) Roll

Fig. 7. Comparison of the estimated poses and the ground truth on the Boston University dataset.



(a)

Fig. 8. Comparison of the estimated poses and the ground truth on the Tuebingen dataset.

means that our approach is more robust as it can be initialized and recovered at larger distances. Therefore, under aspects of performance and real time capabilities on mobile robots, our method is a better choice than the methods proposed by Xiao *et al.* and La Cascia *et al.*

V. CONCLUSIONS AND FUTURE WORK

In this paper, we presented a robust real time method for face tracking and estimating its roll and yaw angles in uncontrolled environments. Our experimental results show that this method is robust to track faces and estimate the face orientation, and suitable for real time applications such as uncontrolled face recognition on mobile robots. For future development, we first try to find techniques to speed up our algorithm to be able to estimate poses of many faces in a group. Second, it is possible to estimate the pitch angle of the face pose, which is necessary for many real applications. Finally, the main goal of our future research is to apply the technique of face tracking and pose estimation for real time and uncontrolled face recognition on mobile robots. Handling the changing poses of the face is one of the major challenges of face recognition because the face image differences caused by rotations are often larger than the inter-person differences used in distinguishing identities. Moreover, recognizing the face in arbitrary poses will be more difficult in uncontrolled environments under varying illumination. In the near future the technique of face pose estimation in this paper will be able to improve the performance of face recognition in our mobile robot systems.

TABLE III

COMPARISON OF ACCURACY AND PROCESSING TIME BETWEEN OUR METHOD AND STATE-OF-THE-ART METHODS ON UNIFORM-LIGHT SET OF THE BOSTON UNIVERSITY DATASET.

	Our approach	Xiao *et al.* [16]	La Cascia *et al.* [15]
Yaw (deg)	5.67	3.8	2.9
Roll (deg)	3.53	1.4	2.9
Frequency (fps)	50	15	15

REFERENCES

[1] B. A. D. David S. Bolme, J. Ross Beveridge and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Computer Vision and Pattern Recognition, 2010 IEEE Computer Society Conference on*, 2010.

[2] P. Viola and M. Jones, "Robust real-time object detection," in *International Journal of Computer Vision*, 2001.

[3] G. Hager and P. Belhumeur, "Real-time tracking of image regions with changes in geometry and illumination," in *Computer Vision and Pattern Recognition, 1996. Proceedings CVPR '96, 1996 IEEE Computer Society Conference on*, 1996, pp. 403–410.

[4] M. Isard and A. Blake, "Contour tracking by stochastic propagation of conditional density," in *In Proc. European Conf. Computer Vision*, 1996, pp. 343–356.

[5] S. Birchfield, "Elliptical head tracking using intensity gradients and color histograms," in *Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on*, 1998, pp. 232–237.

[6] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," 2008.

[7] A. Saffari, C. Leistner, J. Santner, M. Godec, and H. Bischof, "On-line random forests," in *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, 2009, pp. 1393–1400.

[8] B. Babenko, M.-H. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 33, no. 8, pp. 1619–1632, 2011.

[9] X. Mei and H. Ling, "Robust visual tracking using l1 minimization," in *Computer Vision, 2009 IEEE 12th International Conference on*, 2009, pp. 1436–1443.

[10] E. Murphy-Chutorian and M. Trivedi, "Head pose estimation in computer vision: A survey," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 4, pp. 607–626, 2009.

[11] A. H. Gee and R. Cipolla, "Determining the gaze of faces in images," in *Image and Vision Computing*, vol. 1, 1994, pp. 639–647.

[12] T. Horprasert, Y. Yacoob, and L. Davis, "Computing 3-d head orientation from a monocular image sequence," in *Automatic Face and Gesture Recognition, 1996., Proceedings of the Second International Conference on*, 1996, pp. 242–247.

[13] R. Newman, Y. Matsumoto, S. Rougeaux, and A. Zelinsky, "Real-time stereo tracking for head pose and gaze estimation," in *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, 2000, pp. 122–128.

[14] G. Fanelli, J. Gall, and L. Van Gool, "Real time head pose estimation with random regression forests," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, 2011, pp. 617–624.

[15] M. L. Cascia, S. Sclaroff, and V. Athitsos, "Fast, reliable head tracking under varying illumination: An approach based on registration of texture-mapped 3d models," in *In IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000, pp. 322–336.

[16] J. Xiao, T. Kanade, and J. F. Cohn, "Robust full-motion recovery of head by dynamic templates and re-registration techniques," in *Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition*, ser. FGR '02. Washington, DC, USA: IEEE Computer Society, 2002, pp. 163–. [Online]. Available: http://dl.acm.org/citation.cfm?id=874061.875442

[17] M. Vo-Duc, A. Masselli, and A. Zell, "Real time face detection using geometric constraints, navigation and depth-based skin segmentation on mobile robots," in *2012 IEEE International Symposium on Robotic and Sensors Environments*, 2012.