Face Pose Estimation and Tracking Using Automatic 3D Model Construction

Pedro Jiménez, Jesús Nuevo, Luis M. Bergasa
Department of Electronics
University of Alcala
CAMPUS. 28805 Alcalá de Henares (Madrid), Spain.
E-Mail: pjimenez,jnuevo,bergasa@depeca.uah.es

Abstract

This paper presents a method for robustly tracking and estimating the face pose of a person in both indoor and outdoor environments. The method is invariant to identity and that does not require previous training. A face model is automatically initialized and constructed on-line, when the face is frontal to the stereo camera system. To build the model, a fixed point distribution is superposed over the frontal face, and several appropriate points close to those locations are chosen for tracking. Using the stereo correspondence of the two cameras, the 3D coordinates of these points are extracted, and the 3D model is created. RANSAC and POSIT are used for tracking and 3D pose calculation at each frame. The approach runs in real time, and has been tested on sequences recorded in the laboratory and in a moving car.

1. Introduction

Face detection and tracking is a very active research field in computer vision, and a comprehensive number of methods have been developed [1]. Face detection is also the first step in many other algorithms in face recognition, modeling, expression analysis and other areas of computer vision. Face pose estimation has attracted interest for its usefulness in different applications. It is an important cue of where the person is directing his or her attention, and thus has been widely used in Human-Machine Interface applications, sometimes coupled with gaze estimation [2]. It is a principal component of many driver inattention monitoring systems [3, 4, 5].

Many different approaches have been made to the face pose estimation problem. In recent years, Active Appearance Models [6] have been extended to enable pose estimation, in 2D [7] and 2D+3D spaces[8]. Three dimensional face models [9] include pose estimation as a part of the fitting process. Appearance models process the face as a flexible object, and have been shown to work reliably in many different applications. Unfortunately, their fitting algorithms are computationally expensive, with a few exceptions such as [8], and some of them exhibit fitting convergence problems when the face and illumination change rapidly. They also require a time-consuming training process. Some of these shortcomings have been solved in [10], where detection of non-rigid surfaces is done based on keypoint recognition. This algorithm works in real time, and the keypoint classifier can be trained within minutes [11].

Other methods simplify the processing by considering the face, or at least part of it, as a rigid object. Usually the eyes and nose are tracked in this algorithms. In [12] a method for tracking the upper part of the face is presented. The authors used the reflection of near-infrared light on the user's eyes (*red-eye* effect) as a step of their algorithm. This technique has also been used in [13] to estimate the pose of the face. These works obtained accurate tracking and estimations in real time in indoor tests. However, the *red-eye* effect may not appear in an outdoor environment. Also, continuous exposure to near-IR lighting is a known cause of eye fatigue.

Several approaches use generic machine learning to evaluate the pose. In [14], a method using Support Vector Machines (SVM) is presented. The method accurately discriminated images of faces, but in those images the faces were only in three different poses. In [3], Viola & Jones algorithm [15] is used to locate the face, and a SIFT-like [16] descriptor is calculated over the face area. A SVM trained in regression is used to estimate the pitch and yaw angles. This method yields good results in tests in a moving vehicle, but it is com-

978-1-4244-2340-8/08/$25.00 ©2008 IEEE

putationally too expensive to run in real-time, and it only provides estimates for two angles and not the full 3D pose.

Almost all the works mentioned above use monocular vision, and then estimate the pose using a model or a pre-learned mapping from the computed characteristics to the 3D pose.

In this paper we propose a stereo system that is able to work with different users without prior training. The only requirement is for the user to be in frontal position to the camera system for a few frames during initialization. This approach resembles [13] in the automatic model initialization step, but our system does not require the use of near-IR to locate the eyes, and relies on various facial features to robustly track the person's face, even when the eyes are occluded. The presented algorithm uses stereo vision to construct a 3D model of the face. Adequate features for tracking on the person's face are found using the Harris detector [17]. Using the stereo correspondence of the two cameras, the 3D coordinates of these points are extracted, and the model is created. A modified Simultaneous Modeling and Tracking algorithm (SMAT) [18] is used to track the facial features on the video sequences from the cameras independently, and RANSAC [19] and POSIT [20] are used for a robust 3D pose calculation at each frame. While the 3D model is rigid, it contains enough points so that possible errors introduced by face deformations can be handled by the algorithm.

The rest of the paper is structured as follows: in section 2, we describe our approach to face model building and pose estimation in detail. Test results can be found in section 3. Conclusions and future works are presented in section 4.

2. Face model and pose estimation

In order to obtain the user's face direction, it is first necessary to construct a 3D model of the face. This model is formed by a set of 3D points of the face. The 2D projections of these points on each camera are tracked on each frame, using the Simultaneous Modelling and Tracking (SMAT) [18] algorithm. From the 2D points, the 3D face pose is obtained, using the POSIT algorithm for pose extraction and RANSAC for erroneous point elimination. After a set of correctly tracked points (inliers) is obtained, the position of the outlier points is set accordingly to the estimated pose. A diagram of this process is shown in figure 1.

2.1. 3D Face model creation

To define the model of the face, we use a coordinate system affixed to the right camera. The model points

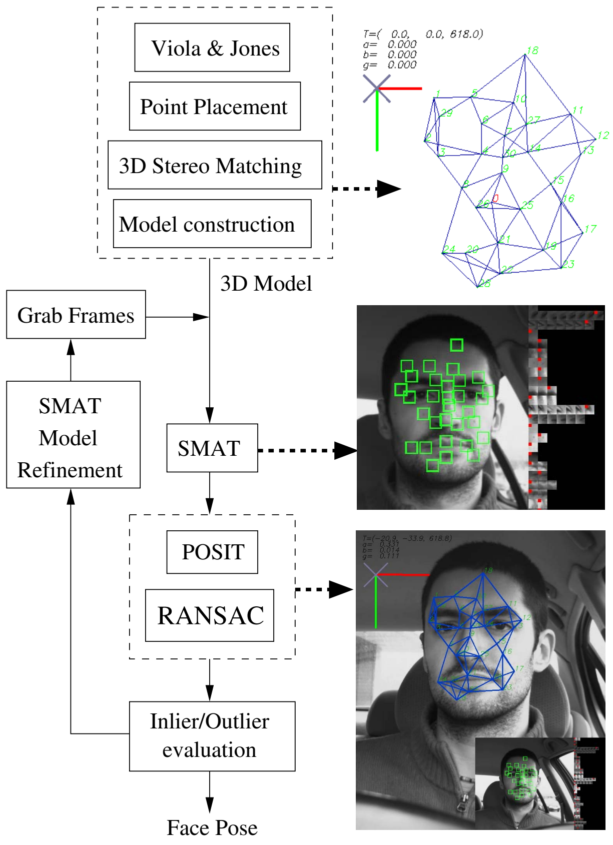


Figure 1. System block diagram

are referenced to another coordinate system, with origin on the central point of the model. $\vec{X}$ axis is the horizontal axis, and grows to the right of the image. $\vec{Y}$ axis is the vertical, and grows down the frame, and the $\vec{Z}$ axis is perpendicular to the image plane and grows to the rear of the scene so that the nose of the driver should have the most negative z value.

The face pose is characterized by a translation within the camera coordinate system, and a pointing vector. This vector is defined as the normal vector to the face, and at the moment of the model creation is set to $\vec{v}_{ini} = (x, y, z) = (0, 0, -1)$, as can be seen in figure 2. The translation vector points to the center of the model.

The first step to create the model is to localize the user's face. Viola & Jones [15] [21] algorithm is used in both cameras to localize the position of a frontal face within the camera frames. This algorithm returns a box that encloses the face on each frame. We reduce the size of this box by 33%, so that it only encloses the face with bigger certainty, as shown in figure 3.

The algorithm initialization requires the person to front at the cameras for a few frames at the beginning of system operation. At this moment, the model is

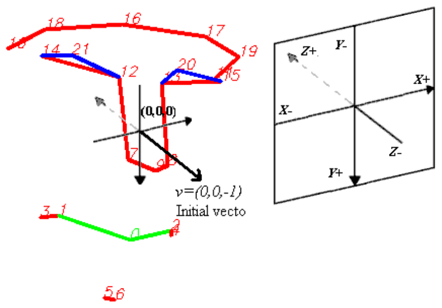
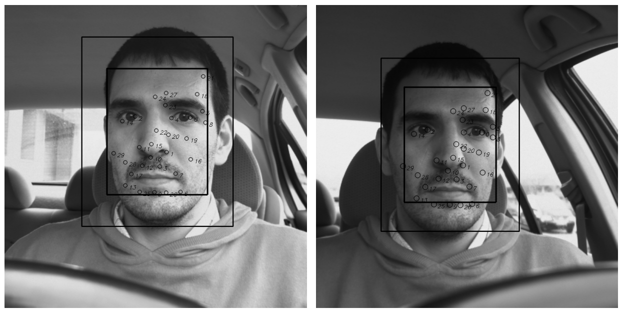


Figure 2. Face Model, coordinate system and initial model vector

considered to have a pointing vector $\vec{v}_{ini} = (x, y, z) = (0, 0, -1)$. If the user is not correctly positioned, the difference between the real pose vector at the initialization step and $\vec{v}_{ini}$ will appear as a constant offset error.



(a) Left image (b) Right image

Figure 3. Model Construction

The face model is defined by thirty points that are tracked over successive frames. To choose appropriate points, a predefined standard face pattern is scaled and placed over the detected inner box containing the face on the left camera image. These points may not correspond to any good feature on the user's face to be used for tracking, so a characteristic feature close to each pattern point is chosen for tracking, as shown in figure 3. The Harris algorithm [17] is used to locate points with good contrast and tracking characteristics. Stereo correspondence of these points over the other camera are used to calculate its 3D coordinates. The stereo camera system was calibrated using the Camera Calibration Toolbox for MATLAB.

The model is built with the 3D coordinates of the thirty feature points. The model origin is then moved to the closest point to the center of mass of the model, so that the initial 3D coordinates of the points are independent of the initial face position and distance to the camera, and the initial pose vector is set to $\vec{v}_{ini}$.

2.2. Model self-occlusion

The face model is subject to self-occlusion when the head turns over a certain angle, and some of the model points may not be visible, or appear too distorted to be correctly tracked. To detect such points, a hidden-point pattern is created after the model initialization. Each point is associated a limit rotation angle within the point is visible. When the face rotation angle is over the limit angle of a point, it is considered to be hidden and the point is not processed for tracking and pose estimation.

To create the hidden-point pattern, a circle is adjusted to the (x, z) coordinates of each model point, as shown in figure 4. The circle is adjusted to minimize the function

$$w_k = \sum(\sqrt{(x_o - x_i)^2 + (z_o - z_i)^2} - R^2), \quad i = 1..30 \tag{1}$$

where (x_i, z_i) are the x and z coordinates of each model point, and (x_o, z_o) and R are the center in the (x, z) plane and the radius of the circle.

Each point of the model is hidden when its angle with respect to v_{ini} excesses ± 60 degrees.

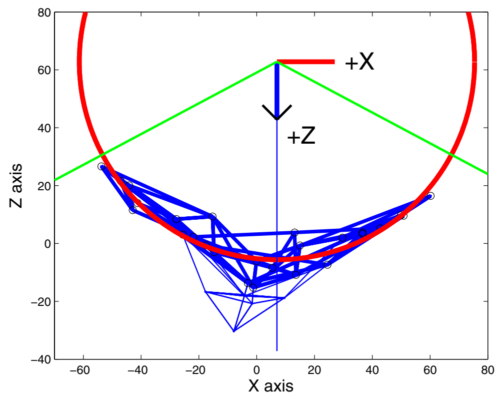


Figure 4. Circle fit to the face to get the limit angles

2.3. Tracking using SMAT

The Simultaneous Modeling and Tracking (SMAT) [18] is a recently developed technique for tracking objects in sequences. It is closely related to other techniques such as [22], but it does not require any previous training. We briefly outline its main characteristics here, and some modifications we have included over the original work proposed by Dowson *et al.*

SMAT works by building a library of exemplars obtained from previous frames in the sequence. The exemplars in the library, image patches in our case, are clustered based on their relative distance, and the medians of the clusters are used for fitting the model to the next frame. A new exemplar is included in one of the clusters depending on the distance to its medians, or a new cluster is created if the new exemplar is too far away from the existing ones. As a group of similar exemplars, each of these clusters will approximately represent different appearances of the same feature of the object. The resulting mixture model is fitted to the next frame. Two tracking examples with a simple 4 patch model are shown in figure 5.

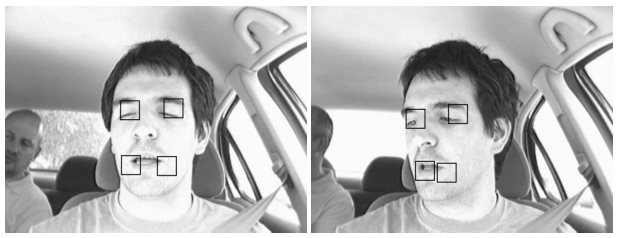

(a) Left image (b) Right image

Figure 5. SMAT based tracking

In the original paper, Dowson *et al.* [18] included a model of structure for the distribution of the exemplars on the images, that is also built on-line. However, our implementation only builds an appearance model, and the point distribution of the shape is constrained by the 3D model. It has been shown [23] that location and tracking errors are mainly due to appearance, and that a generic shape model for faces is easier to construct. Then, it is possible to reduce uncertainty and complexity by only learning the appearance model. This model also permits to use robust methods, and discard points and exemplars that do not fit well, improving the overall robustness of the tracking. Further details are given in the section below.

The formulation of the SMAT algorithm is independent of the definition of distance and the minimization method used. We have used Zero mean Normalized Cross-Correlation (ZNCC) and Sum of Squared Differences (SSD) as distances, and Gauss-Newton and the Nelder-Mead simplex method [24] for the minimization process.

2.4. Pose estimation

After the position of the tracking points has been updated for both the left and right frames independently, the 3D face pose is to be estimated from the 2D projection of each point. However, the matching process may not succeed for all points, and can result in errors or drifting for some of them. These errors negatively influence the accuracy of the estimated pose. Thus, a robust optimization method is required to estimate the best matching 3D face pose, that would detect as outliers the points that have been incorrectly tracked, so they can be safely discarded. We also consider that points that have been correctly tracked may have some random noise. The RANSAC algorithm is used to eliminate the outliers. 3D pose is obtained using DeMenthon's four point iterative pose estimation algorithm (POSIT) [20]. The POSIT algorithm calculates the pose of a 3D rigid object from its projection on a single image. It estimates the pose by first approximating the perspective projection as an scaled orthographic projection, and then iterating refining the estimation until the distance between the projected points and the ones obtained with the estimated pose falls below a threshold.

The pose is given as a translation $\vec{T}$ and a rotation $\vec{R}$ matrices, which indicate the position of the central point of the model with respect to the camera coordinate system, and its rotation from the initial model given.

In each RANSAC iteration, seven points are randomly selected from the model, and used to calculate the pose ($\vec{R}$ and $\vec{T}$ matrix) using the POSIT algorithm. With this $\vec{R}$ and $\vec{T}$, all 3D original points of the model are projected over the image plane, and the Euclidean distance from the tracking point to the corresponding projected point is calculated. If this distance is less than a threshold, this point is considered to be correct, and marked as an inlier. The RANSAC algorithm runs for enough iterations to guarantee a 99% of success with 50% of outliers.

This process is performed over the left and right frames independently, and the final pose estimation is calculated from the pose estimations as a weighted sum, according to the expressions:

$$\vec{R}_{model} = \frac{\vec{R}_{right} \cdot I_l}{I_l + I_r} + \frac{\vec{R}^r_{left} \cdot I_r}{I_l + I_r}, \quad if\ I_r, I_l > I_{min} \tag{2}$$

$$\vec{T}_{model} = \frac{\vec{T}_{right} \cdot I_l}{I_l + I_r} + \frac{\vec{T}^r_{left} \cdot I_r}{I_l + I_r}, \quad if\ I_r, I_l > I_{min} \tag{3}$$

where I_l and I_r are the number of inliers from the left and right pose estimations, as determined with RANSAC. $\vec{R}_{model}$ and $\vec{T}_{model}$ are the resulting pose estimation. $\vec{R}_{right}$ and $\vec{T}_{right}$ are the pose estimation from the right image, and $\vec{R}^r_{left}$ and $\vec{T}^r_{left}$ are pose estimation from the left camera, translated to the right camera using the corresponding stereo equations and camera calibration parameters. In case the number of inliers of any of the cameras is less than the I_{min} threshold, set to half the total number of points, that estimation is discarded and the estimation of the other camera is used. If inliers for both images are below the threshold, the frame is rejected and the estimation from the previous frame is used.

2.5. Tracking Failure detection and Recovery

Points identified as outliers by the RANSAC algorithm are moved to a corrected position, so they can be tracked on the following frames. The new position of the points is calculated by re-projecting the 3D model on both camera planes with the final estimated pose, $\vec{R}_{model}$ and $\vec{T}_{model}$.

The SMAT model is also inspected when outliers are found, as it has been updated with all the image patches, regardless of their validity. Incorrect patches could contaminate the model and induce further tracking errors. Thus, patches that correspond to outlier points on the last frame are also considered outliers, and removed from the SMAT model.

3. Test results

The algorithm has been tested with videos recorded in the laboratory and in a moving car during daytime. The videos start with the user facing front to the cameras. The sequences recorded in the car show normal driving gestures and head movements. The videos were recorded using two synchronized FireWire cameras, at a framerate of 20 fps and a resolution of 800 x 800 pixels. The total length of the sequences is over an hour.

For each video, the model construction process is carried out over the first frame. The system chooses up to 30 characteristic tracking points to built the model. After the corrections are done and erroneous points are automatically eliminated, the model is correctly created, as the point occlusion pattern, based on a cylinder-like face.

After the model is created, the tracking and pose estimation process starts. As shown in figure 6, the pose is correctly estimated over face rotations. The more the face is rotated, the more points are hidden, and thus the accuracy of the pose estimation falls. This reduced accuracy appears for angles that result in more than 50% of the points being hidden (over 30 degrees). When approximately 75% of the model's points are hidden, the RANSAC algorithm does not have enough points to get the correct set of inliers and outliers, and thus the pose estimation fails. This situation is displayed in the last two frames on figure 7. The images cover different head rotations, and show the estimated pose vector.

To compare the results with the real face rotation angle, four points have been manually placed on the both images for each frame. From these points and using the stereo equations, the real face pose can be obtained with good precision. This value has been used to test the accuracy of the proposed algorithm, as shown in figure 6. Table 1 shows the estimation error. The high value of the error for the yaw angle is caused by the fast increasing error when the angle is over 30°, as mentioned above.

Angle	MAS error (°)	Std error (°)
Pitch	4.92	5.50
Yaw	8.14	7.98
Roll	2.66	2.94

Table 1. Pose Estimation Mean Absolute Error

The algorithm has been coded in C/C++, and it is able to run in real-time in a 2.4GHz Core2 Duo processor. Tracking with SMAT is the most time consuming process. Although its processing time varies slightly depending on the number of iterations required, it was below the real-time threshold in all our tests. Table 2 shows the mean and maximum processing times for tracking and pose estimation, for the given system with 30 points.

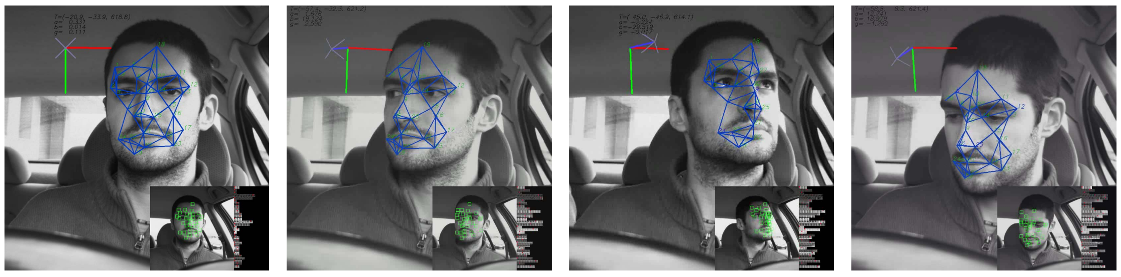
Task	Mean time	Max time
SMAT	18 ms	21 ms
RANSAC+POSIT	13 ms	15 ms

Table 2. Processing times

4. Conclusions

This papers presents a face tracking and pose estimation algorithm using stereo vision that runs in real-time. The algorithm is able to automatically construct a 3D model of the face, just requiring the driver to look straight ahead for a few seconds. Tracking of feature points is carried out independently on left and right images using SMAT, and incorrectly tracked points are rejected using RANSAC. 3D pose is recovered from the set of points using POSIT.

The algorithm has been tested in video sequences recorded in the laboratory and in a moving vehicle, and works reliably for face rotations under 40° degrees. Rotations greater than this value result in a great number of points being occluded, and the pose can not be estimated. To solve this problem, we are working on on-line extended model creation. This would augment the model when the face is rotated left or right, and the global algorithm accuracy drops below a threshold due point occlusion. The same method used to create the initial frontal model will be used to extend the model



(a) Frame 25 (b) Frame 110 (c) Frame 225 (d) Frame 310

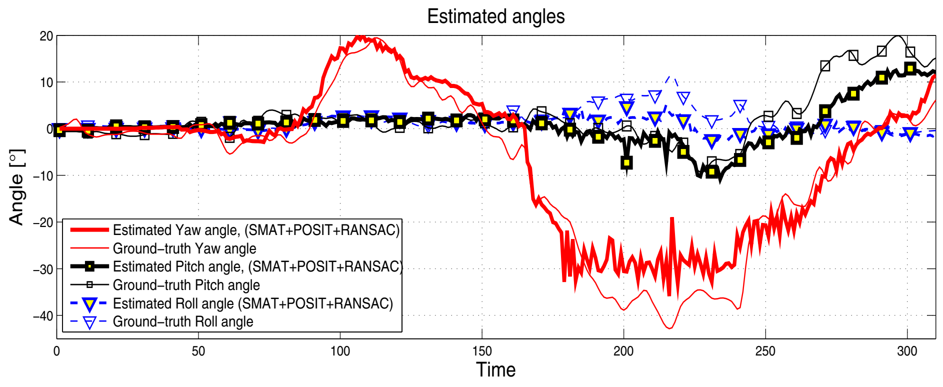
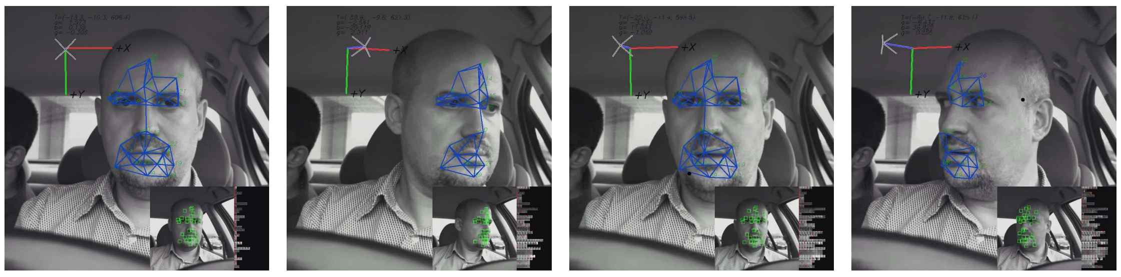


Figure 6. Tracking and Pose estimation of the face of a driver A under day light driving conditions.



(a) Frame 10 (b) Frame 80 (c) Frame 130 (d) Frame 225

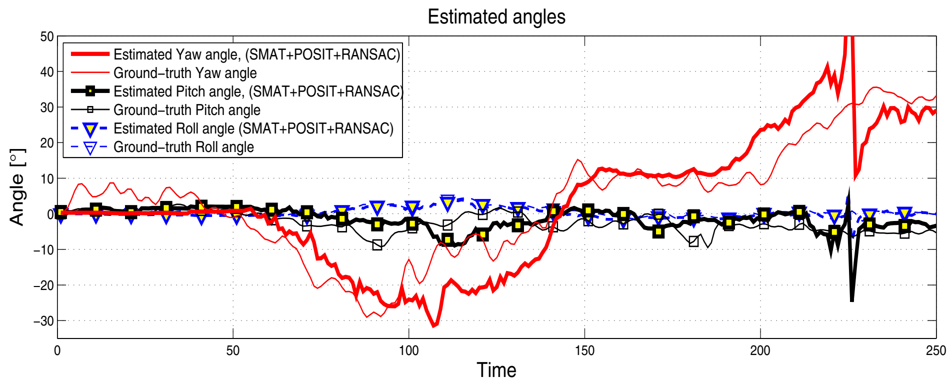


Figure 7. Tracking and Pose estimation of the face of a driver B under day light driving conditions.

along the sides of the face, taking care that no possible features in the user's hair are chosen. This augmented model would include all the initial model points and the newly added. A Kalman filter would also improve system stability especially when the face rotation angle is close to its limits. With these additions, this algorithm is to be used as the base of a distraction detection system for drivers.

Acknowledgment

This work has been funded by grant TRA2005-08529-C02-01 (MOVICOM Project) from the Spanish Ministry of Science and Technology (MCyT), and S-0505/DPI/000176 (Robocity2030 Project) from the Science Department of the Comunidad de Madrid. J. Nuevo is also working under a researcher training scholarship from the Education Department of the Comunidad de Madrid and the European Social Fund.

References

[1] M. Yang, D. Kriegman, and N. Ahuja, "Detecting faces in images: a survey," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 24, no. 1, pp. 34–58, 2002.

[2] R. Stiefelhagen, J. Yang, and Waibel, "Tracking eyes and monitoring eye gaze," in *Proceedings of PUI'97*, 1997.

[3] E. Murphy-Chutorian, A. Doshi, and M. M. Trivedi, "Head pose estimation for driver assistance systems: A robust algorithm and experimental evaluation," in *Proceedings of Intelligent Transportation Systems Conference 2007*, 2007, pp. 709–714.

[4] L. M. Bergasa, J. Nuevo, M. A. Sotelo, R. Barea, and E. López, "Real-time system for monitoring driver vigilance," *IEEE Trans. Intell. Transport. Syst.*, vol. 7, no. 1, pp. 1524–1538, Mar. 2006.

[5] R. Senaratne, D. Hardy, B. Vanderaa, and S. Halgamuge, "Driver Fatigue Detection by Fusing Multiple Cues," *Lecture Notes In Computer Science*, vol. 4492, p. 801, 2007.

[6] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active appearance models," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 23, pp. 681–685, Jan. 2001.

[7] T. Cootes, G. Wheeler, K. Walker, and C. Taylor, "View-based active appearance models," *Image and Vision Computing*, vol. 20, no. 9-10, pp. 657–664, 2002.

[8] J. Xiao, S. Baker, I. Matthews, and T. Kanade, "Real-time combined 2D+ 3D active appearance models," in *IEEE Conference on Computer Vision and Pattern Recognition 2004*, vol. 2, 2004, pp. 535–542.

[9] V. Blanz and T. Vetter, "Face recognition based on fitting a 3d morphable model," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 25, no. 9, pp. 1063–1074, 2003.

[10] J. Pilet, V. Lepetit, and P. Fua, "Real-time non-rigid surface detection," in *IEEE Conference on Computer Vision and Pattern Recognition 2007*, San Diego, CA, June 2005.

[11] M. Ozuysal, P. Fua, and V. Lepetit, "Fast keypoint recognition in ten lines of code," in *IEEE Conference on Computer Vision and Pattern Recognition 2005*, Minneapolis, MI, June 2007.

[12] A. Kapoor and R. Picard, "Real-time, fully automatic upper facial feature tracking," *Automatic Face and Gesture Recognition, 2002. Proceedings. Fifth IEEE International Conference on*, pp. 8–13, 2002.

[13] Z. Zhu and Q. Ji, "Real Time 3D Face Pose Tracking From an Uncalibrated Camera," *CVPR Workshop 2004, Conference on*, pp. 73–80, 2004.

[14] J. Huang, X. Shao, and H. Wechsler, "Face pose discrimination using support vector machines (SVM)," in *Proceedings of ICPR*, vol. 1, 1998.

[15] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *IEEE Conference on Computer Vision and Pattern Recognition 2001*, 2001, pp. 511–519.

[16] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[17] C. Harris and M. Stephens, "A combined corner and edge detector," *Alvey Vision Conference*, vol. 15, p. 50, 1988.

[18] R. Dowson, N.D.H.; Bowden, "Simultaneous modeling and tracking (SMAT) of feature sets," in *IEEE Conference on Computer Vision and Pattern Recognition 2005*, vol. 2, 2005, pp. 99–105.

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

[20] D. F. Dementhon and L. S. Davis, "Model-based object pose in 25 lines of code," *Int. J. Comput. Vision*, vol. 15, no. 1-2, pp. 123–141, 1995.

[21] P. Viola and M. Jones, "Robust real-time object detection," *Int. J. Comput. Vision*, pp. 137–154, 2002.

[22] D. Cristinacce and T. Cootes, "Feature Detection and Tracking with Constrained Local Models," in *17th British Machine Vision Conference*, 2006, pp. 929–938.

[23] R. Gross, I. Matthews, and S. Baker, "Generic vs. person specific active appearance models," *Image and Vision Computing*, vol. 23, no. 11, pp. 1080–1093, Nov. 2005.

[24] J. Nelder and R. Mead, "A simplex method for function minimization," *Computer Journal*, vol. 7, no. 4, pp. 308–313, 1965.