Original Article
International Journal of Fuzzy Logic and Intelligent Systems
Vol. 13, No. 1, March 2013, pp. 50-58
http://dx.doi.org/10.5391/IJFIS.2013.13.1.50

ISSN(Print) 1598-2645
ISSN(Online) 2093-744X

Novel Backprojection Method for Monocular Head Pose Estimation

Kun Ju[1], Bok-Suk Shin[2], and Reinhard Klette[2]
[1]Vista Entertainment Solutions, Auckland, New Zealand
[2]Computer Science Department, The University of Auckland, Auckland, New Zealand

ljfis

Abstract

Estimating a driver's head pose is an important task in driver-assistance systems because it can provide information about where a driver is looking, thereby giving useful cues about the status of the driver (i.e., paying proper attention, fatigued, etc.). This study proposes a system for estimating the head pose using monocular images, which includes a novel use of backprojection. The system can use a single image to estimate a driver's head pose at a particular time stamp, or an image sequence to support the analysis of a driver's status. Using our proposed system, we compared two previous pose estimation approaches. We introduced an approach for providing ground-truth reference data using a mannequin model. Our experimental results demonstrate that the proposed system provides relatively accurate estimations of the yaw, tilt, and roll angle. The results also show that one of the pose estimation approaches (perspective-n-point, PnP) provided a consistently better estimate compared to the other (pose from orthography and scaling with iterations, POSIT) using our proposed system.

Keywords: Head pose estimation, Backprojection, Monocular vision

1. Introduction

Driver-assistance technologies are the subjects of widespread research and development in the car industry, especially computer-vision-based driver assistance [1]. Computer-vision-based driver monitoring is aimed mainly at detecting sleepiness and distraction using a camera mounted in front of the driver (e.g., on the dashboard). By studying and analyzing the frames recorded by the camera, the system can provide feedback or warn the driver if their state is abnormal, even in non-ideal lighting conditions [2]. Research is focused on detecting the signs of sleepiness, particularly by analyzing a driver's eyes and their gaze. For example, long blinks shows that a driver is experiencing fatigue. Gaze detection can also determine where a driver is looking. When the eyes focus in one direction for a long period, this indicates that driver may be experiencing distraction.

However, some situations must include head pose estimation, e.g., eye detection will not work if a driver is wearing (dark) sunglasses while driving, and eyes cannot be detected on occasions when the driver is wearing a hat. The direction of the head can convey important information, so we can use head poses to understand where driver is focused and to determine a driver's state while driving. For example, a sudden head movement while driving may indicate that the driver is looking at something special outside the car. Long-term bowing or frequent nodding can also indicate that the driver is tired or sleepy. In these cases, it is better

Received: Feb. 25 2013
Revised : Mar. 13, 2013
Accepted: Mar. 15, 2013

Correspondence to: Reinhard Klette
(r.klette@auckland.ac.nz)

to use head pose analysis to determine whether a driver is distracted or suffering from sleepiness. These issues are addressed in this paper. Our goal is to develop a system that can provide a robust estimation of a driver's head pose to analyze the status of the driver. This is achieved by *combining well-established methods with a novel backprojection technique*. We use two pose-estimation approaches, perspective-n-point (PnP) [3] and pose from orthography and scaling with iterations (POSIT) [4]. Both pose estimation approaches are based on 2D-3D correspondences, so we also use common KLT features [5] to detect and track *good feature points* inside the face region. The driver's head is represented usingan ellipsoid model, which provides a more intuitive view of the estimated head pose.

The remainder of this paper is divided into the following sections. Section 2 provides a brief description of previous head-pose estimation approaches. Section 3 describes the details of our proposed system. Section 4 presents all the experiments we conducted, with a discussion of the results. Section 5 provides our conclusions and possible future work.

2. Related Work

Previous head-pose estimation approaches can be grouped into *feature-based*, *appearance-based*, or *model-based approaches* [6–10].

Feature-based approaches generally utilize facial features, such as the eyes, nose, or mouth, in 3D space or in the image domain to calculate the actual head direction. They learn the correlations between the positions of facial features and the head pose using training data [11, 12]. Furthermore, in [6], the researchers detected the upper points of the eyebrows, the upper nasolabial-furrow corners, and the nasal root. They considered these features as stable facial feature points.

Appearance-based approaches focus on the entire head image instead of specific facial features before learning the pose by training. These are also known as *global head-pose estimation approaches*. To locate the head, they only need to locate the face, without detecting any facial features or creating a face model. After the face is located, template matching is performed to find the best matching pose and generate the pose estimation. The most popular template-matching approaches use Gabor wavelets, principal components analysis (PCA), support vector machines (SVM), or neural networks [13–15].

Model-based approaches always need to create a 3D model to represent the human head. Pose recognition is then achieved by matching the 2D points in the head region in the image plane with their corresponding 3D points on a 3D model. The pose (rotation and translation) of the 3D model is then calculated and it has to be consistent with the pose of the human head. Various 3D models have been proposed for representing the human head. Frequently used 3D models are the active appearance model (AAM), a cylindrical head model (CHM), the ellipsoidal head model (EHM), or a simple planar model [16–18].

3. Proposed System

The proposed head-pose estimation system is able to provide the roll, yaw and tilt angle of the driver's head from a monocular image sequence. Then, according to those angles and detected changes of them, we will be able to understand the status of the driver.

3.1 System Overview

The flowchart in Figure 1 demonstrates basic procedures of the proposed head-pose estimation system. A monocular image is the input of the system for one run. Initially, face detection is applied to find the face region in the input image. The detected region is considered to be the *region of interest*.

Then, the system detects the *good features* within this region of interest. The next step is to create a 3D model to represent the human head in order to provide an intuitive visualization when demonstrating the driver's status, and also to have a surface for feature backprojection.

In the proposed system, because POSIT and PnP are utilized for pose estimation, both require not only the feature points in 2D coordinates on the image plane but also corresponding 3D coordinates which are located on the surface of the 3D model. Therefore, the projection points of the 2D feature points on the 3D model have to be computed beforehand.

Moreover, by camera calibration we calculate the focal length of the used camera before computing 3D projection points and implementing the POSIT pose-estimation function, and also the camera intrinsic matrix and distortion coefficients needed for implementing the PnP pose-estimation function.

Finally, the pose-estimation algorithms are used to calculate or update the affine transform of the driver's head. Based on the rotation matrix, the system computes roll, yaw, and tilt angles of the driver's head. Then, the created 3D model is rotated based on those three estimated rotation angles (e.g., for visualizing the head pose). As a consequence, the viewing direction of the driver can be detected.

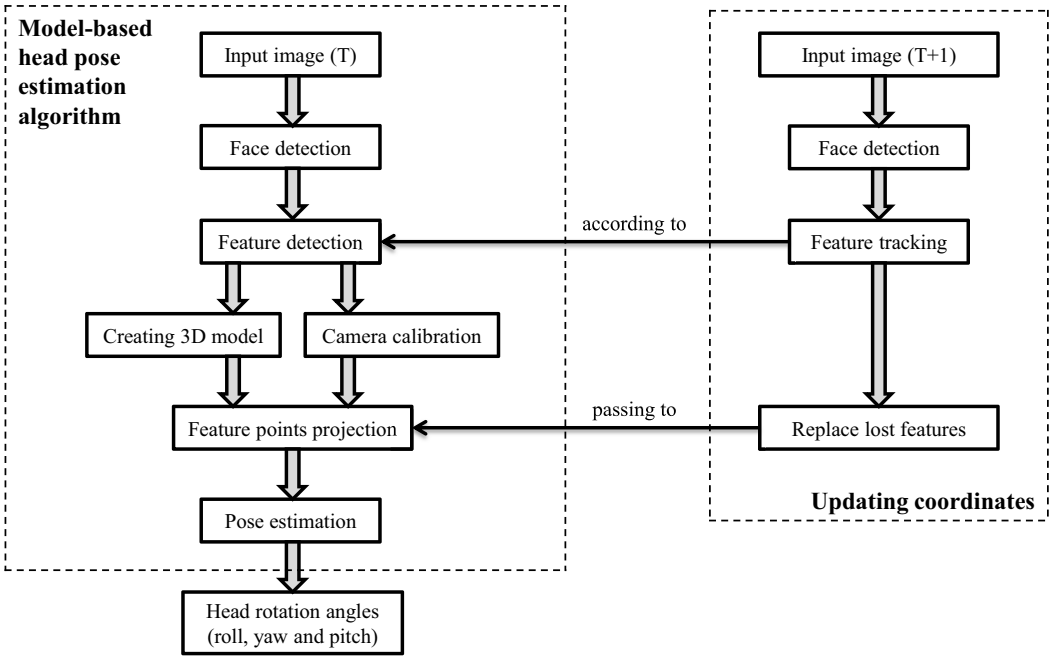


Figure 1. Flowchart of the proposed-model-based head pose estimation algorithm.

Moreover, to analyze the status of the driver, the system has to work on a sequence of images to study changes in roll, yaw and tilt angles. Thus, a procedure for updating the coordinates is also included. It consists of detecting the new face region in the new input image and tracking previously detected good features. Then, as some of the good feature points are lost during tracking, the number of lost features is replaced by newly selected ones. Next the updated good features will be passed on to the process of feature point backprojection. At the end, the new pose is estimated according to the updated feature points.

After applying all these processes on a sequence of images, we analyze the results for providing information about the status of the driver. The remaining parts of this section describe the procedures more in detail.

3.2 Face Detection

The camera is assumed to be mounted on the dashboard in front of the driver. Face detection is the first step for locating the head. For initialization, we assume a frontal view on a face where yaw, tilt, and roll angles are small. In general, a driver is looking forward, thus we can expect that this initialization state occurs in a reasonable time.

The most descriptive head feature points are in the face region. Compared to this, using a profile face view or other angular face views for initialization works against robust and "meaningful" feature detection, and creates an uncertainty for the initialization of head pose angles. The Viola-Jones face detection approach is used for detecting the face region by simply using the pre-trained frontal-view cascade classifier from the OpenCV library due to its robustness and high performance for the considered initialization step.

3.3 Feature Detection

Tracking the movement of individual feature points in the face region provided for us better results (for understanding head rotation) than tracking the whole face region at once. We apply the KLT feature detector for selecting the features in the face region. Detected features are evaluated for their quality (their trackability), thus defining selected good features within the detected face region. These features are stored in descending order according to their *goodness* and then fed to the tracker later on.

3.4 Creating a 3D Model

There are many different 3D models that can be utilized for representing a driver's head. Popular 3D models are the cylin-

drical head model, the ellipsoidal head model, a detailed active appearance model, or a planar model. For the detailed active appearance model, there are many degrees of freedom that need to be considered when initializing the model. Furthermore, its tracking performance is not regarded as being robust and precise [16]. The cylindrical or ellipsoidal head model appear to be more preferable due to simplicity (fewer degrees of freedom, reasonable fits to a human head).

According to experimental results in [16], the 3D cylindrical head model can provide robust performance on yaw and roll, but is unstable for tilt. Therefore, for the proposed system, we decide to render an *ellipsoid of revolution* [19] as the model to represent the driver's head.

3.5 Camera Calibration

As mentioned before, camera calibration is needed due to the requirement of both pose estimation algorithms, as well as the approach which is used to backproject 2D points onto the 3D ellipsoidal model surface. By calibrating the camera, we only need to obtain the *camera intrinsic matrix* and the *lens distortion coefficients*.

3.6 Backprojection of 2D Points

In order to get rotation angles to render the ellipsoid model at the correct pose of the driver's head, we compare the performance of PnP and POSIT pose-estimation methods. We have to provide not only coordinates of 2D image points but also their corresponding 3D coordinates on the surface of the ellipsoid. The feature detection stage provided the 2D feature points. In this step, we calculate their corresponding 3D points which are projected back onto the surface of the ellipsoid.

The concept here is to calculate the intersection of rays with an ellipsoid [20]. A ray is defined by

$$L(s) = \mathbf{p} + s\mathbf{d} \tag{1}$$

where L defines the locations of points on the ray in dependency of a non-negative real s; $\mathbf{p}$ is the origin of the ray, and $\mathbf{d}$ its direction. We assume that the camera's focal point is the origin of projection rays. See Figure 2. Direction $\mathbf{d}$ is represented as (u, v, f), where u and v are the coordinates of a 2D point in the image plane, and f is the effective focal length of the used camera. As we keep the number of selected good feature points constant, we also have the same constant number of rays from origin towards the ellipsoid.

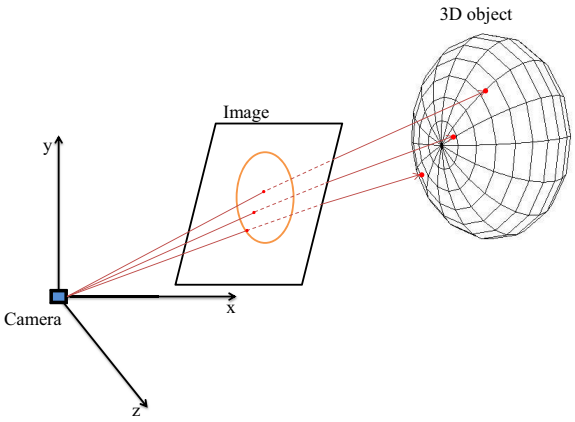


Figure 2. Ray tracing starts at the camera's focal point, passes through 2D points in the image plane and corresponding 3D points on the ellipsoid's surface.

Next, a general ellipsoid is represented as below, where $C = (C_x, C_y, C_z)$ is the center point, r the radius (e.g., scaled to be equals 1), with axes defined by k, l, and m:

$$k(x - C_x)^2 + l(y - C_y)^2 + m(z - C_z)^2 - r^2 = 0 \tag{2}$$

However in our system, we utilize an ellipsoid of revolution with a pair of equal semi-axes (i.e., $k = m$) but a distinct third semi-axis. Thus, the equation of the ellipsoid can be modified to

$$k\left[(x - C_x)^2 + (z - C_z)^2\right] + l(y - C_y)^2 - r^2 = 0 \tag{3}$$

By substituting Eq. (1) into Eq. (3), the intersection is defined by

$$\begin{aligned} k \quad \cdot \quad & \left[(p_x + sd_x - C_x)^2 + (p_z + sd_z - C_z)^2\right] \\ & + l(p_y + sd_y - C_y)^2 - r^2 = 0 \end{aligned} \tag{4}$$

simplified as a quadratic equation in the form

$$as^2 + bs + c = 0 \tag{5}$$

with

$$a = k(d_x^2 + d_z^2) + ld_y^2 \tag{6}$$

$$\begin{aligned} b = \; & 2k\left[(p_x - C_x)d_x + (p_z - C_z)d_z)\right] \\ & + 2l(p_y - C_y)d_y \end{aligned} \tag{7}$$

$$\begin{aligned} c = \; & k\left[(p_x - C_x)^2 + (p_z - C_z)^2\right] \\ & + l(p_y - C_y)^2 \end{aligned} \tag{8}$$

Table 1. OpenCV pose estimation methods

solvePnP	*cvPOSIT*
3D object points	3D object points
2D image points	2D image points
Camera matrix	Focal length
Distortion coefficients	Criteria

PnP, perspective-n-point; POSIT, pose from orthography and scaling with iterations.

and solutions

$$s_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{9}$$

By substituting a, b, and c into Eq. (9), two s values are defined. In our case, we only need the closest intersection point (i.e., the smaller s-value) and substitute it back into Eq. (1) for calculating the coordinate of the projection point on the ellipsoidal surface.

Now we have 2D feature points and corresponding 3D coordinates on the ellipsoid surface; we are ready to estimate the pose of the driver's head.

3.7 PnP/POSIT Pose Estimation

For this stage, we implemented two different pose estimation methods from the OpenCV library, *sovlePnP* and *cvPOSIT*, for comparison. Referring to the OpenCV library, we need to provide some required data for both methods.

In fact, all the previous steps are the preparation stages for triggering these pose estimation methods. Once the program is initialized, the requested 2D points can be obtained from the feature detection step as well as the corresponding 3D points on the surface of the ellipsoid. These data are required by both pose-estimation methods as indicated by Table 1. Data *camera matrix*, *distortion coefficients* and *focal length* are obtained by camera calibration. Criteria are used to decide how many iterations need to be done for generating a sufficiently accurate pose. A result when using *solvePnP* is shown on the left of Figure 3, and the right image shows a result generated by *cvPOSIT*.

The previously described steps are all for initializing head pose estimation. In order to analyze the status of the driver, we also analyze subsequent frames for providing a sequence of head-pose estimations.

By using initial 3D model points and updated 2D points, both tested pose-estimation algorithms provide updated estimations. However, two problems need to be solved. First, when a driver's head is rotating, tracked feature point get lost. As a result, there are insufficient feature points to be fed into the pose estimation algorithms. As a consequence, they will not be able to provide robust estimations for a driver's head. Second, there is the goal of automatically re-initializing the system. We need to decide when and how to re-initialize, such that time-efficiency and system performance is guaranteed.

In the following sections, we describe how to update 2D coordinates for generating new estimations of the driver's head pose, as well as how to solve those two problems mentioned above.

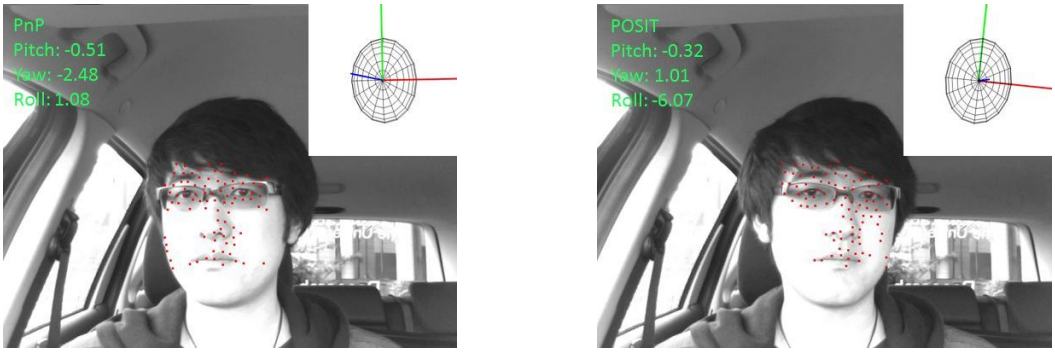
3.8 Updating Feature Points

Updating of 2D coordinates consists of two steps: tracking of feature points and replacing lost feature points by newly detected ones.

As illustrated in Figure 1, in order to increase the performance of tracking, face detection is applied again for the new input image. However, at this time, face detection is based on frontal and profile face detection, as well as on the previously detected face region. The reason is that the face in the new image may not be the frontal view. If not, the face will be detected by using profile-detection procedure or according to the previously detected face region due to the small displacement between two consecutive frames. Once the face region is again detected or estimated, the KLT tracker will track previously detected features within this region.

However, by experience, many features will be lost due to the preset constraints. Not only due to head motion, there are various other reasons which can lead to tracking errors, such as illumination changes, feature location occlusions, and so on. Thus, in order to prevent bad pose estimation due to a lack of feature points, lost feature points are replaced by using newly detected feature points. Once the replacing process is finished, the feature list is updated. Next, the new 2D and 3D correspondences will be obtained, then new pose estimation will be calculated.

According to our design, the system has to be able to do re-initialization automatically. For example, based on the detected head pose it could be decided when to trigger the function for replacing lost feature points. However, a simple approach of replacing lost features in every n-th frame provided even a better solution compared to more complicated approaches, such as, replacing lost features whenever the driver's face returns back to the frontal view, presetting few specific yaw angles as the re-initialization points (such as when ever the yaw angle is



(a) (b)

Figure 3. (a) Result of head pose estimation using *solvePnP*. (b) Result of head pose estimation using *cvPOSIT*. PnP, perspective-n-point; POSIT, pose from orthography and scaling with iterations.

about 0 °, 15 °, 35 °, etc.), and so on.

We take every fifth frame as a re-initialization frame to replace lost features. More in detail, contributing processes perform the following:

1. Estimate the head pose; if a frontal-view face then first (re-)initialization.
2. Calculate three head-rotation angles for the first (re-) initialization frame.
3. Trigger the function to replace lost features.
4. Estimate the head pose in the following frames, until the next re-initialization frame, according to newly updated features.
5. Calculate the head rotation angles for the current frame. The result will be referred to as that for the most recent re-initialization frame.
6. Adding newly calculated head rotation angles which refer to the most recent re-initialization frame for providing the head rotation angles which refer to the frontal view face.

4. Experimental Results

We conducted extensive experiments using our proposed system with image sequences recorded in a test vehicle. However, these real-world data lacked ground-truth data for comparison. Thus, we also developed an approach for generating ground-truth data. We compared the estimated results (using the PnP and POSIT pose-estimation approaches) with the head-pose ground-truth data, which were obtained from manually marked feature points or automatically tracked feature points in the input images.

To achieve this, we used a mannequin model as our test object and recorded three sequences: one for changing the tilt angle only, one for changing the yaw angle only, and one for changing the roll angle only. Sample input images and their manually marked feature points are shown in Figure 4.

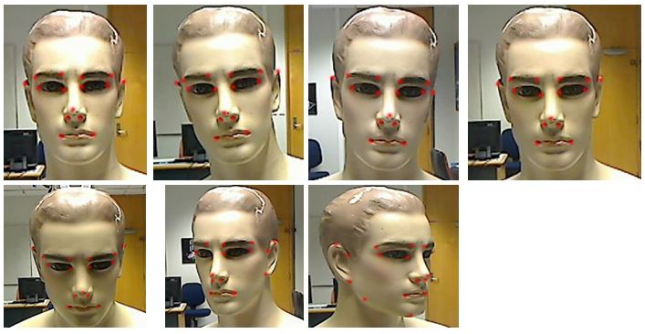


Figure 4. Recorded images of a mannequin model with manually marked feature points.

4.1 Using Mannequin Model Sequences

Based on experiments using mannequin model sequences, we were able to determine the accuracy of our results when using manually marked feature points for head pose estimation or the results when running our system to automatically detect feature points, and we compared the results with the ground-truth head pose data. We calculated the standard deviations of the errors between the pose results based on manually marked feature points or the KLT-detected feature points compared with the measured mannequin head pose.

The first four rows in Table 2 show the standard deviations of the errors for the estimated angles, which were obtained using both pose estimation approaches with manually marked feature points or KLT-detected features, compared with the ground-truth data. In general, PnP provided better estimates than POSIT, while the manually marked feature points yielded

Table 2. Standard deviations of the errors

	Yaw	Tilt	Roll
PnP (manual vs. ground truth)	2.22°	2.02°	1.43°
POSIT (manual vs. ground truth)	2.40°	2.21°	1.64°
PnP (KLT vs. ground truth)	2.73°	2.42°	1.81°
POSIT (KLT vs. ground truth)	2.94°	2.70°	1.94°
PnP (KLT vs. manual)	4.11°	3.17°	2.62°
POSIT (KLT vs. manual)	4.24°	3.24°	3.15°

PnP, perspective-n-point; POSIT, pose from orthography and scaling with iterations.

Table 3. PnP results

Parameter	Standard deviation of errors				Avg. std.
Tilt-PnP	3.17°	2.62°	2.83°	3.34°	2.99°
Yaw-PnP	4.11°	3.95°	3.64°	4.4	4.04°
Roll-PnP	2.62°	1.25°	1.98°	1.65°	1.88°
Tilt-POSIT	3.24°	2.53°	2.85°	3.5	3.04°
Yaw-POSIT	4.30°	3.28°	3.46°	3.96°	3.75°
Roll-POSIT	3.15°	2.02°	2.92°	1.8	2.49°

PnP, perspective-n-point; POSIT, pose from orthography and scaling with iterations.

better results than automatically (here: KLT) detected feature points.

The results obtained using manually mark points were very close to the (measured) head-pose ground-truth data. Thus, they can be regarded as *substitute reference values* for testing the accuracy of the results when applying the proposed system to the test vehicle sequences. Both pose-estimation approaches were relatively sensitive to changes in the types of feature points. Furthermore, we found that the estimated roll angles had the most accurate values. This was because the change in the angle was not significant when the head was slanted so the majority of the feature points remained visible.

This required fewer replacements of lost feature points, and to more accurate and robust estimates. The estimated yaw angles had the largest standard deviation error. This was because the changes in yaw angle could be very large and a large yaw angle change required more replacements.

As mentioned earlier, both pose estimation approaches were sensitive to changes in the feature points. Therefore, there were bigger errors when estimating the yaw angle. The accuracy of the estimated tilt angle was between the yaw and roll angle estimates.

The last two rows in Table 2 show the standard deviation error for the KLT approach relative to the manually marked points approach. These values can be used as reference for checking the accuracy of the results when applying the pose estimation approaches to test vehicle sequences.

4.2 Using Test Vehicle Sequences

We applied manually marked feature points to four test vehicle sequences (including scenes of a non-occluded driver's face, eyeglasses, and dark sunglasses) to provide a *substitute reference value*. Table 3 shows the standard deviation errors for the *substitute reference values* relative to the PnP and POSIT pose estimation results, as well as the average standard deviation of the errors.

Table 3 shows that the average errors were very similar to the errors shown in the last two rows in Table 2. This demonstrated that the use of the mannequin model sequences was appropriate. Our system can provide relatively accurate estimates of a driver's head pose and PnP provided better results than POSIT using our proposed system.

5. Conclusions and Future Work

We developed a model-based monocular head pose estimation system that utilizes a novel form of backprojection and a KLT feature tracker for selecting, tracking, and replacing feature points in a detected face region, as well as calculating their corresponding 3D coordinates on the surface of an ellipsoid model to estimate the pose of a driver's head and to analyze the status of the driver's head.

There is no publicly available source code for other head-pose estimation methods, so our comparisons were limited to re-implementation. However, we compared two different pose estimation algorithms: PnP and POSIT. Using the results for manually marked feature points as *substitute reference values*, we showed that PnP yielded better results than POSIT with our system. Furthermore, based on the experiments we conducted to obtain ground-truth data, we showed that both of these pose estimation approaches were sensitive to changes (i.e., updates) in the feature points.

It would be interesting to test the performance of other feature points (e.g., SIFT and Harris corner detector) to improve the accuracy of our estimation results. Moreover, the rapid advances in mobile devices, such as smartphones, means that a mobile device application would be a very useful extension of our system. We could obtain many advantages from mobile device applications. For example, the majority of smartphones have a built-in camera so we would simply mount it on the driver's side of the windscreen instead of using a separate camera.

Conflict of Interest

No potential conflict of interest relevant to this article was reported.

References

[1] R. Klette, J. Ahn, R. Haeusler, S. Hermann, J. Huang, W. Khan, S. Manoharan, S. Morales, J. Morris, R. Nicolescu, F. Ren, K. Schauwecker, and X. Yang, "Advance in vision-based driver assistance," in *Proceedings of 2011 International Conference on Electric Technology and Civil Engineering*, Lushan, 2011, pp. 987-990. http://dx.doi.org/10.1109/ICETCE.2011.5775884

[2] M. Rezaei and R. Klette, "Adaptive Haar-like classification for eye monitoring in non-ideal lighting conditions," in *Proceedings of the 27th Conference on Image and Vision Computing New Zealand*, Dunedin, 2012, pp. 521-526. http://dx.doi.org/10.1145/2425836.2425934

[3] Y. H. Wu and Z. Y. Hu, "PnP problem revisited," *Journal of Mathematical Imaging and Vision*, vol. 24, no. 1, pp. 131-141, Jan. 2006. http://dx.doi.org/10.1007/s10851-005-3617-z

[4] D. F. Dementhon and L. S. Davis, "Model-based object pose in 25 lines of code," in *Proceedings of Second European Conference on Computer Vision*, Ligure, 1992, pp. 335-343. http://dx.doi.org/10.1007/3-540-55426-2_38

[5] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, Vancouver, 1981, pp. 674-679.

[6] M. Dahmane and J. Meunier, "Oriented-filters based head pose estimation," in *Proceedings of Fourth Canadian Conference on Computer and Robot Vision*, Montreal, 2007, pp. 418-425.

[7] I. Marras, N. Nikolaidis, and I. Pitas, "3D head pose estimation in monocular video sequences by sequential camera self-calibration," in *Proceedings of 2009 IEEE International Workshop on Multimedia Signal Processing*, Rio De Janeiro, 2009, pp. 1-6. http://dx.doi.org/10.1109/MMSP.2009.5293570

[8] X. Wang, X. Huang, J. Gao, and R. Yang, "Illumination and person-insensitive head pose estimation using distance metric learning," in *Proceedings of the 10th European Conference on Computer Vision: Part II*, Marseille, 2008, pp. 624-637. http://dx.doi.org/10.1007/978-3-540-88688-4_46

[9] M. T. Wenzel and W. H. Schiffmann, "Head pose estimation of partially occluded faces," in *Proceedings of the 2nd Canadian Conference on Computer and Robot Vision*, Victoria, 2005, pp. 353-360. http://dx.doi.org/10.1109/CRV.2005.45

[10] Y. D. Zhu and K. Fujimura, "Head pose estimation for driver monitoring," in *Proceedings of 2004 IEEE Intelligent Vehicles Symposium*, Parma, 2004, pp. 501-506. http://dx.doi.org/10.1109/IVS.2004.1336434

[11] P. Fitzpatrick, *Head Pose Estimation without Manual Initialization*, Cambridge, MA: AI Lab, MIT, 2000.

[12] T. Vatahska, M. Bennewitz, and S. Behnke, "Feature-based head pose estimation from images," in *Proceedings of 2007 7th IEEE-RAS International Conference on Humanoid Robots*, Pittsburgh, 2007, pp. 330-335. http://dx.doi.org/10.1109/ICHR.2007.4813889

[13] L. G. Dong, L. M. Tao, G. Y. Xu, and P. Oliver, "A study of two image representations for head pose estimation," in *Proceedings of Fifth International Conference on Image and Graphics*, Xian, 2009, pp. 963-968. http://dx.doi.org/10.1109/ICIG.2009.141

[14] N. Gourier, J. Maisonnasse, D. Hall, and J. Crowley, "Head pose estimation on low resolution images," in *Proceedings of the 1st international evaluation conference on Classification of events, activities and relationships*, Southampton, 2007, pp. 270-280. http://dx.doi.org/10.1007/978-3-540-69568-4_24

[15] 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 the 2007 IEEE Intelligent Transportation Systems Conference*, Seattle, 2007, pp. 709-714. http://dx.doi.org/10.1109/ITSC.2007.4357803

[16] S. Choi and D. Kim, "Robust head tracking using 3D ellipsoidal head model in particle filter," *Pattern Recognition*, vol. 41, no. 9, pp. 2901-2915, Sep. 2008. http://dx.doi.org/10.1016/j.patcog.2008.02.002

[17] P. Martins and J. Batista, "Monocular Head Pose Estimation," in *Proceedings of 5th international conference on Image Analysis and Recognition*, Povoa de Varzim, 2008, pp. 357-368. http://dx.doi.org/10.1007/978-3-540-69812-8_35

[18] E. Murphy-Chutorian and M. M. Trivedi, "Head pose estimation and augmented reality tracking: an integrated system and evaluation for monitoring driver awareness," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 2, pp. 300-311, Jun. 2010. http://dx.doi.org/10.1109/TITS.2010.2044241

[19] S. Basu, I. Essa, and A. Pentland, "Motion regularization for model-based head tracking," in *Proceedings of the 13th International Conference on Pattern Recognition*, Vienna, 1996, pp. 611-616. http://dx.doi.org/10.1109/ICPR.1996.547019

[20] P. J. Schneider and D. H. Eberly, "Geometric tools for computer graphics," Amsterdam: Morgan Kaufmann Publishers, 2003.

Kun Ju is a Software Services Consultant at Vista Entertainment Solutions Ltd, New Zealand. Before he was a Master student at the Computer Science Department, The University of Auckland, and he was a member of the *.enpeda..* (Environment Perception and Driver Assistance) research group. He was doing research related to optical flow (evaluation of different algorithms) and head pose estimation for driver assistance systems.

Bok-Suk Shin is a PhD graduate of Pusan National University, Korea. Since February 2011 she is a Post-Doc researcher at the Computer Science Department, The University of Auckland, in the *.enpeda..* (Environment Perception and Driver Assistance) research group. She has 10 years of experience in research, project design, and teaching in data analysis, pattern recognition, computer vision, 3D visualization, and 3D game development. So far, her publications have been dominantly on track recognition for small species, with a current shift towards 3D computer vision.

Reinhard Klette is a Fellow of the Royal Society of New Zealand and a professor at The University of Auckland. He is currently the Editor-in-Chief of the Journal of Control Engineering and Technology, on the editorial boards of the International Journal of Computer Vision and the International Journal of Fuzzy Logic and Intelligent Systems, and was an associate editor of IEEE PAMI in 2001-2008. He (co-)authored more than 250 publications in peer-reviewed journals or conferences, and books on computer vision, image processing, geometric algorithms, and panoramic imaging. He presented more than 20 keynotes at international conferences.