Estimating Gaze Direction of Vehicle Drivers using a Smartphone Camera

Meng-Che Chuang
University of Washington
Seattle, WA, USA
mengche@uw.edu

Raja Bala, Edgar A. Bernal, Peter Paul, Aaron Burry
Xerox Research Center Webster
Webster, NY, USA
{Raja.Bala, Edgar.Bernal, Peter.Paul, Aaron.Burry}@xerox.com

Abstract

Many automated driver monitoring technologies have been proposed to enhance vehicle and road safety. Most existing solutions involve the use of specialized embedded hardware, primarily in high-end automobiles. This paper explores driver assistance methods that can be implemented on mobile devices such as a consumer smartphone, thus offering a level of safety enhancement that is more widely accessible. Specifically, the paper focuses on estimating driver gaze direction as an indicator of driver attention. Input video frames from a smartphone camera facing the driver are first processed through a coarse head pose direction. Next, the locations and scales of face parts, namely mouth, eyes, and nose, define a feature descriptor that is supplied to an SVM gaze classifier which outputs one of 8 common driver gaze directions. A key novel aspect is an in-situ *approach for gathering training data that improves generalization performance across drivers, vehicles, smartphones, and capture geometry. Experimental results show that a high accuracy of gaze direction estimation is achieved for four scenarios with different drivers, vehicles, smartphones and camera locations.*

1. Introduction

The U.S. Department of Transportation's National Highway Traffic Safety Administration (NHTSA) reports that over 3000 fatalities from automobile accidents are caused by driver drowsiness or distraction [1]. In response, there is an emerging body of research on the use of in-vehicle cameras and sensors coupled with computer vision techniques to monitor driver behavior for enhanced safety [2-4]. Most technologies in the market today are found primarily in high-end automobiles (e.g., Mercedes Benz, BMW, etc.) and rely upon sophisticated image capture and processing afforded by specialized hardware that interacts with built-in vehicle telematics.

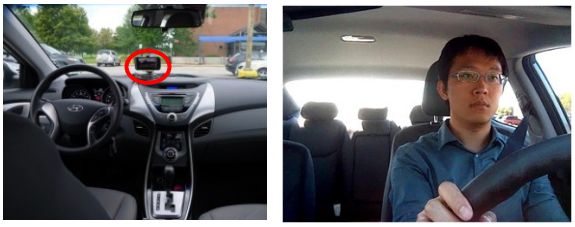
In this paper, we explore an approach for driver monitoring that can be executed on a mobile device such as a consumer smartphone. This approach has several advantages. First, consumer smartphones are widely used, and can offer certain simple and inexpensive forms of driver assistance accessible by the mainstream market. Secondly, the mobile solution is associated closely with the user rather than the vehicle, and thus can potentially incorporate customized driver-specific context into the safety monitoring process. Thirdly, mobile technologies are rapidly advancing in terms of hardware and imaging capabilities, and can be adapted and upgraded at a much faster cadence than embedded solutions offered by automobile manufacturers. All said, we acknowledge that the processing capabilities of a consumer smartphone are not likely to match that of a dedicated embedded solution and that the two frameworks could be synergistically combined to leverage the advantages of both.

Specifically, this paper focuses on estimating driver gaze direction from smartphone video captured of the driver. The novel contributions of the paper are two-fold. First, the 3D gaze space is quantized into 8 common gaze directions in feature space, and gaze estimation is formulated as an efficient classification problem. The second and more significant contribution is an offline training method that gathers training data for the gaze classifier *in-situ* for a given driver/vehicle/camera setup. This step significantly improves generalization performance of the classifier. Data collection is designed with safety and convenience in mind.

The rest of this paper is organized as follows. Section 2 briefly describes related work on smartphone-based driver monitoring and human gaze estimation. Section 3 introduces the proposed method of driver gaze estimation. Section 4 reports the experimental results, and concluding remarks are collected in Sec. 5.

2. Related work

Smartphone-based driver monitoring is a relatively nascent area offering rich potential for both research and application development. Eren et al. [5] proposed using the inertial sensors in a smartphone, namely the accelerometer, gyroscope, and magnetometer to measure position, speed, acceleration, and deflection angle, and relate the measured data to driver behavior. While this approach can monitor some aspects of driving behavior, it is unable to predict important events such as driver drowsiness or distraction that could lead to unsafe driving conditions. The iPhone app *iOnRoad* [6] requires the user to place the smartphone on the windshield with the rear camera facing the road. The app claims to monitor distance to nearby vehicles and lane



(a) (b)

Figure 1: (a) Camera dashboard mount; (b) driver view in used camera geometry.

departure, as well as vehicle speed via GPS. It does not, however, capture any imagery or footage of the driver and thus cannot monitor driver attention or fatigue. To our knowledge, one of the more comprehensive smartphone apps for driver monitoring is the *CarSafe App* proposed by You et al. [7]. The authors propose dual video capture from the driver-facing and road-facing cameras of a smartphone. Monitored road-facing events include lane departure and tailgating. Driver facing events include drowsiness and attention, the latter being monitored via coarse estimates of frontal, left, and right gaze directions.

There is a significant body of literature on human gaze estimation [8-13]. The approaches generally attempt to determine gaze either in terms of precise roll/pitch/yaw angle or in terms of coarse (e.g., left, front, right) directions. Some of the high-precision techniques are based on 3D geometric models of human pose and gaze.

This paper presents a method for driver gaze estimation that is amenable to execution on mobile devices with limited computational resources. Gaze space is quantized into 8 directions commonly encountered during driving. This approach thus falls in-between the two extremes of very fine and very coarse gaze estimation found in the literature. The gaze estimation is accomplished via a rather standard choice of features and SVM classifier. However, a key novelty is in the method of training the classifier to perform reliably across a wide range of conditions, as explained in Sec. 3.

3. Gaze estimation algorithm

Our approach requires that a smartphone be mounted on the windshield or dashboard of a vehicle with front-camera facing the driver. Since many state laws in the United States place restrictions on windshield mounts, our preference is for the dashboard mount, as shown in Fig. 1(a). Experiments have suggested that a mounting location near the center of the dashboard (i.e., below the rear view mirror) is convenient for user interaction and offers an acceptable view of the driver (see Fig 1(b)). In the proposed system, we specify 8 gaze directions commonly encountered during driving, as shown in Fig. 2. Gaze estimation thus boils down to gaze classification with respect to the 8 classes, which enables a computationally efficient implementation. Next

Figure 2: Eight common gaze directions used to train the classifier.

we describe the online classification method, with the training method being described in Sec. 3.2.

3.1. Gaze classification method

Figure 3 is a block diagram of the online steps involved in estimated gaze direction from input video frames. Given the need for quasi-real-time performance on a smartphone, computational efficiency was a prime consideration when designing the features and selecting the classification technique. The steps of the method are as follows:

i) **Coarse head pose detection.** Input video frames are first processed through frontal, left- and right-profile face detectors to determine coarse head pose direction. These face detectors comprise fast Adaboost cascade classifiers operating on Haar feature descriptors [14].

ii) **Gaze feature descriptors.** Next we process only those frames classified as containing frontal poses from the previous step. Facial regions are sent to modules that detect the location of left iris, right iris, mouth and nose (see Fig. 4). These are again Adaboost cascade classifiers trained specifically to detect the respective facial regions. A 14-dimension feature vector comprising normalized face part locations and sizes is defined as follows.

$$x = (x_{le}, y_{le}, s_{le}, x_{re}, y_{re}, s_{re}, \\ x_n, y_n, w_n, h_n, x_m, y_m, w_m, h_m) \quad (1)$$

where x and y are spatial locations, s denotes side length, and w and h denote width and height, respectively. Subscripts *le*, *re*, *n*, *m* denote respectively left eye, right eye, nose, and mouth. The position and size of each face part are normalized by the lengths of the axes of the "face coordinate system". The latter is defined by the square surrounding the detected face, with the origin located at the top-left corner of the square. The normalization equations are:

$$(x_p, y_p) = \left(\frac{x_p^{cen} - x_f^{lt}}{s_f}, \frac{y_p^{cen} - y_f^{lt}}{s_f} \right), \quad (2)$$

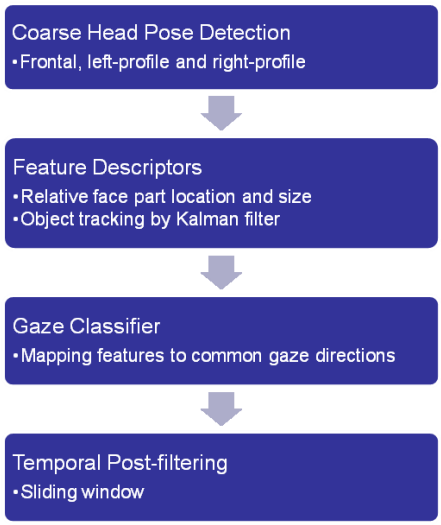


Figure 3: Block diagram of the online stage.

$$s_p = s_p^{ori} / s_f \ , \tag{3}$$

$$(w_p, h_p) = \left(w_p^{ori} / s_f \, , h_p^{ori} / s_f \right), \tag{4}$$

where index $p \in \{le, re, n, m\}$ denotes the face parts, and variables with subscript f are facial coordinates and sizes. The value of each component of the feature vector is thus normalized to the range $[0,1]$. This makes the feature descriptors invariant to facial translation and scaling, as well as to pixel resolution. Additionally, this intrinsic data normalization process ensures that decisions are not dominated by any particular feature. Lastly, the locations of the feature points are temporally smoothed using a Kalman filter tracker.

iii) **Gaze classifier**. The extracted feature vector is input to a classifier that outputs an estimated gaze direction out of 8 possible directions. In the proposed system, a multi-class linear support vector machine (SVM) classifier is employed with a one-versus-one scheme, wherein a binary decision function is learned between every pair of classes.

iv) **Temporal post-filtering**. The final step ensures smoothness in gaze estimates over time, and enforces the assumption that transitions between different gazes occur smoothly relative to the acquisition frame rate. A sliding window history of class labels from the previous 5 frames is maintained, and a majority voting scheme is enforced to determine the final prediction of gaze in the current frame.

3.2. Offline Classifier Training

A crucial and novel aspect of the proposed system is the method by which the gaze classifier is trained. When deploying the application "in the wild", we encountered wide variations across different drivers, vehicles,

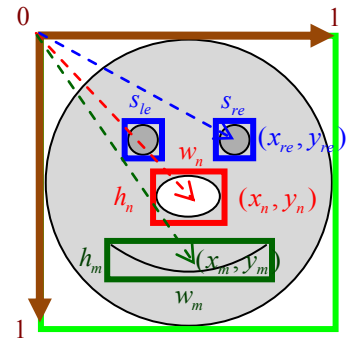


Figure 4: Illustration of relative face part features in normalized face coordinates.

smartphones, and camera placements. Not surprisingly, we noted that the aforementioned classification method did not generalize well when trained on one group of driver/vehicle/camera combinations and tested on another. To address this problem, we propose an *in-situ* approach for gathering training data that is carried out just once for a given combination of driver, vehicle, and camera setup. We must keep in mind, however, that gathering labeled data during the act of driving is unsafe, as it necessitates that the driver gaze in 8 directions in a controlled fashion while the vehicle is in motion. To ensure driver safety during training, we propose a two-stage procedure as follows.

In the first stage, while the vehicle is stationary, the mobile app prompts the driver to gaze in the 8 directions for 4-second intervals; video is recorded and training data automatically collected for each gaze class. This procedure improves generalization across driver/vehicle/camera; however, experiments revealed that a classifier built from this data still does not generalize satisfactorily from a stationary to a moving vehicle. Presumably this is due to effects such as camera jitter, driver movement, varying illumination, variability in driver pose while actually driving, etc. We observed, however, that classification performance can be significantly improved if we augment the training data from the stationary vehicle with data from a moving vehicle for the "road-gazing" class alone (gaze no. 3 in Fig. 2, which is the dominant class).

The second stage of unsupervised training data collection requires the driver to drive normally while the mobile app gathers video footage, usually for a short duration (e.g., 3-4 minutes). Face-part features are extracted from each frame as outlined in Section 3.1. We hypothesize that the various gaze directions form clusters in feature space, and make a critical assumption that the "road-gazing" class will be the dominant cluster. An unsupervised clustering technique is applied in order to identify the dominant cluster. Specifically, we employ the well-known Expectation Maximization (EM) algorithm [15] to learn a Gaussian mixture model from the available samples. The EM algorithm assigns a K-dimensional membership vector

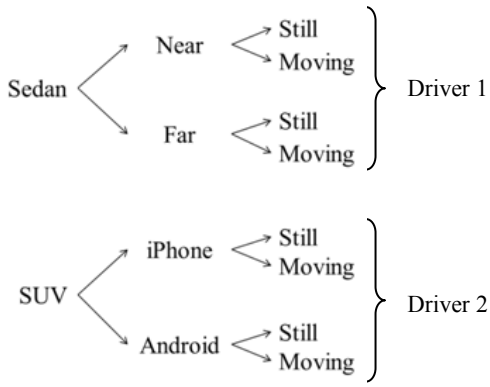


Figure 5: Video dataset from different scenarios.

$m_i \in [0,1]^K$ to each input feature vector $x_i, i = 1,...,n$, where K is the number of clusters (8 in our case), and each element $(m_i)_j$ of m_i represents the likelihood that the i-th input sample belongs to the j-th cluster. Note that the elements of m_i sum to one, i.e., $\sum_{j=1}^{K}(m_i)_j = 1, i = 1,...,n$. It follows that the set of samples that belong to the dominant class (assumed to be road-facing) is given by:

$$X_i^{j_{road}} = \{x_i \mid \arg\max_{j=1,...,K}(m_i)_j = j_{road}\}, \quad (5)$$

where j_{road} denotes the road class. Once the "road-gazing" class is identified, the input samples that are deemed most likely to belong to it based on the output of the EM algorithm (e.g., samples i for which $(m_i)_{j_{road}} \geq (m_i)_j, \forall j$) are assigned to the road-gazing class.

Training data for the 8 gaze classes from the first stage in the stationary vehicle is then combined with training data for the dominant (assumed road-facing) class from the second stage in the moving vehicle to form the overall training set. Finally, a linear SVM classifier is trained on data combined from the two stages.

It is possible that in some instances (e.g., a driver coming out of a parking lot), the dominant data collected in the moving vehicle during the second stage will not correspond to the road gazing class. As a safeguard measure, the application could use additional logic that compares data from the moving vehicle to the clusters gathered in the stationary vehicle, to ensure that the dominant cluster indeed corresponds to road-gazing data. Furthermore route information from GPS could also be incorporated to ensure the training phase encompasses normal road driving.

4. Experiments

The aforementioned gaze classifier was implemented in C++ using a combination of OpenCV and custom libraries. The offline training data collection was performed by smartphone apps running on Android and iOS platforms. Online execution is currently implemented on a Windows 7 laptop machine and is planned to be ported to a mobile platform shortly.

Table I: Coarse-Grained Head Pose Detection Performance

Scenario	Accuracy (%)
Sedan-iPhone-Near	94.3
Sedan-iPhone-Far	99.6
SUV-iPhone	98.9
SUV-Android	98.4

Smartphone video datasets were collected in 4 different experimental scenarios to capture variations in vehicle, driver, smartphone, and mounting geometry, as shown in Fig. 5. Two vehicles were chosen with markedly different interior geometries, a sedan and sport-utility-vehicle (SUV). In the sedan, two mounting locations were chosen, the first directly beneath the rear-view-mirror, and the second midway between the first location and the center of the dashboard. An iPhone5 was used to record video with the front camera. In the SUV, video was recorded with both an iPhone5 and a Samsung Galaxy S4 Android phone. The mounting location was directly beneath the rear-view mirror. Video was captured in both the stationary and moving vehicle cases for each scenario for all 8 gaze classes. This combination of factors results in 8 different data sets.

During the training phase, the gathered data set is randomly split into a training set (80%) and a testing set (20%). A 10-fold cross validation procedure is conducted within the training set to select the model parameters. The classifier is trained accordingly, and then evaluated with the testing set.

4.1. Coarse-grained detection accuracy

Table I shows the confusion matrices along with the overall accuracy of the coarse head pose detection (first stage in Fig. 3) for all four experimental scenarios. The scenario labels "Near" and "Far" refer to the two mounting locations with respect to driver position. Results indicate that high accuracy is achieved by the proposed coarse pose detection system.

4.2. Fine-grained classification accuracy

The frontal face instances detected by the coarse-grained stage are further classified as one of the 8 possible driver gaze classes defined in Fig. 2. The overall classification accuracy of different techniques across the 4 experimental scenarios is reported in Fig. 6. The training techniques are:

- "*Self*": a classifier trained on data extracted from video of a vehicle in motion, applied to test data gathered under the same conditions. This case serves as an upper bound in classifier performance.
- "*Static to Moving*": a classifier trained on data extracted from video in a static vehicle, applied to test data from the same vehicle/driver/camera in motion.

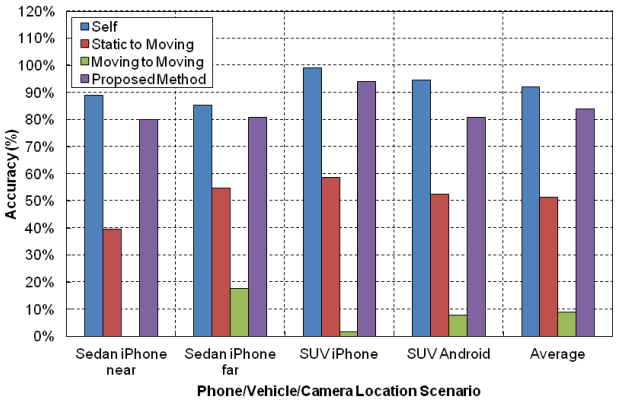


Figure 6: Fine-grained gaze classification accuracy vs. training techniques. Each color represents one method for offline training.

- "*Moving to Moving*": a classifier trained on data from one moving vehicle with a given driver/camera combination, applied to test data from a different moving vehicle and driver/camera combination.
- "*Proposed Method*": the proposed approach with a classifier trained in-vehicle applied to independent test data gathered from the same moving vehicle.

It is observed that classifier generalization from static to moving conditions, and from one moving vehicle/driver camera to another are both poor. This is a common observation in a wide range of machine learning tasks and applications where the training data and testing data are drawn from different distributions. In this particular case, the performance gap can be attributed to the fact that the features being used are fairly sensitive to pose. On the other hand, the proposed method with on-site training comes fairly close to achieving the upper bound in accuracy (i.e., "Self" bars). Note that advances in feature design and classification technique will improve performance in all cases, provided the computations are mobile-amenable.

To gain deeper insight into the performance of our proposed *in-situ* classifier, the confusion matrix for experimental scenario *SUV-iPhone* is shown in Fig. 7. As expected, most of the confusion lies between classes corresponding to similar gaze directions (e.g. road vs. dash-board). Figure 8 shows the classifier output plotted as a function of time (or frame number) with the proposed system. The output labels are the same as those in Fig 2. The results indicate that the proposed system outputs the correct label for a significant portion of time. Figure 9 shows video snapshots in the scenario *Sedan-iPhone-Near*. The final results of face detection, facial part localization and gaze direction class are labeled on the video frame. Clearly the proposed system succeeds at categorizing driver gazes into semantically meaningful directions.

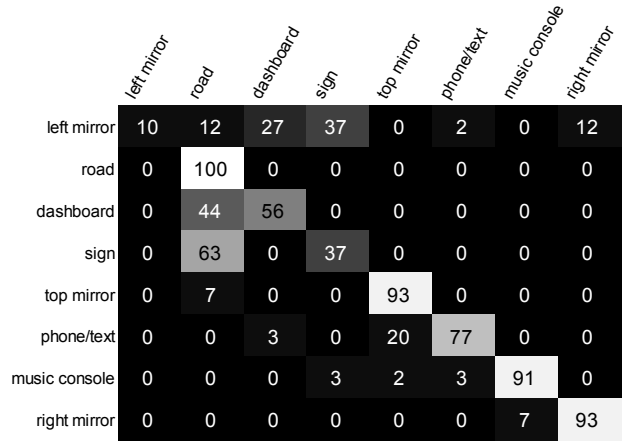


Figure 7: Confusion matrix (in percentage) for the proposed method. Rows are classifier output and columns are ground truth.

4.3. "Safe vs. Unsafe" classification

A variant of the proposed method can be used to provide coarse-scale monitoring of safe vs. unsafe driver behavior. We define two broad, albeit meaningful, categories as follows: Safe = {"road", "left mirror", "right mirror", "top mirror"}, and Unsafe = {"dashboard", "phone/text", "music console"}. Table II reports the accuracy achieved by the fine-grained classifiers for the same four scenarios when analyzed at this broad level of classification. It is shown that classification accuracy is very high at this broad granularity, and may suffice for many applications.

4.4. Computational performance

On a Windows laptop PC powered by an Intel Core i5 2.5GHz processor with 4 GB of RAM, without optimization in the implementation, the gaze estimation time was 50-60 milliseconds per frame, which amounts to 15-20 frames per second. This includes the steps of coarse face direction detection, object tracking, feature extraction, gaze classification and temporal filtering.

Although the online gaze estimation system is yet to be ported to mobile platform, an expected computational performance is argued as follows. We select Samsung Galaxy S4 as a benchmark of modern smartphone, for its largest market share in the U.S. at the time of this work. Powered by a 4-core 1.6-GHz CPU, its computational speed is approximately 64% of the Windows PC used in simulations. One can infer that the processing rate of at least 9.6 frames per second (fps) is achieved by a modern smartphone. This is acceptable for real-time processing, since temporal subsampling can be used to keep up with acquisition frame rate. The computational cost can be further reduced by applying algorithmic optimization. As shown in Fig. 2, most steps involve only linear operations in a 14-dimensional space: Kalman filtering, linear SVM and

Table II: Accuracy of "Safe vs. Unsafe" classification

Scenario	Accuracy (%)
Sedan-iPhone-Near	86.4
Sedan-iPhone-Far	91.2
SUV-iPhone	97.4
SUV-Android	96.0

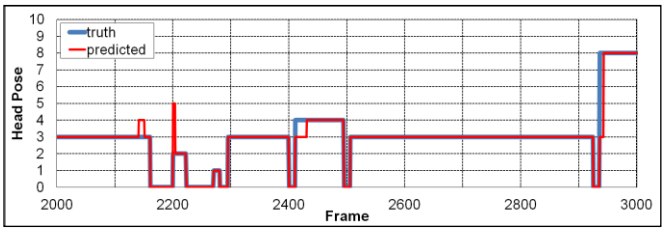


Figure 8: Time plot of gaze direction estimates by the proposed method (red) comparing with the ground truth (blue).

temporal averaging. The heaviest step is object detection, which can be optimized to achieve 15-fps processing rate by a 600-MHz ARM processor [16]. This shows the capability of real-time processing on a modern smartphone even for detecting face and multiple facial parts.

5. Conclusion and future directions

We propose a driver gaze direction estimation technique from smartphone video using computer vision methods. Categorizing the gaze space into 8 common driving gaze directions enables fast extraction of relevant gaze information. Fast feature descriptors and a linear SVM classifier are chosen to contain computational cost. A novel *in-situ* data collection procedure significantly improves generalization performance of the classifier across different driver/vehicle/camera settings without compromising driver safety or convenience.

Immediate future efforts include: i) porting the real-time gaze classification onto the mobile platform; ii) additional experimental validation across a wider gamut of drivers, vehicles and mobile devices. Longer term directions include: i) investigating more efficient and effective features and machine learning approaches for gaze classification; ii) integrating the gaze monitoring module into a larger driver monitoring system that incorporates road-facing video capture along with input from other smartphone sensors (e.g., GPS, accelerometer, etc.) to place gaze direction in context with the state of the vehicle and its environment; iii) incorporating warning mechanisms, e.g. visual, audio, or haptic feedback, that can positively affect driving behavior.

References

[1] "Blueprint for ending distracted driving", NHTSA Report, 2012. Available: http://www.distraction.gov.

[2] M. Bertozzi, A. Broggi, A. Fascioli, "Vision-based intelligent vehicles: State of the art and perspectives," *Robot. Auton. Syst.*, vol. 32, no. 1, pp. 1–16, Jul. 2000.

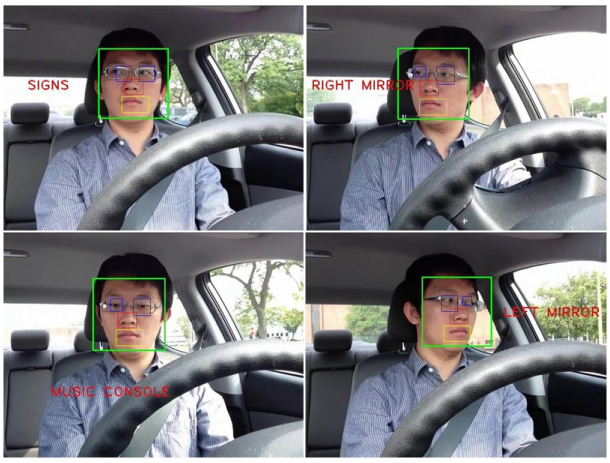


Figure 9: Video snapshots for driver gaze direction estimation in scenario *Sedan-iPhone-Near*. Red labels are gaze estimates.

[3] M. Bertozzi, A. Broggi, M. Cellario, A. Fascioli, P. Lombardi, M. Porta, "Artificial vision in road vehicles," *Proc. IEEE*, vol. 90, no. 7, pp. 1258–1271, Jul. 2002.

[4] M. M. Trivedi, T. Gandhi, J. McCall, "Looking-in and looking-out of a vehicle: computer-vision based enhanced vehicle safety," *IEEE T-ITS*, vol. 8, no. 1, 2007.

[5] S. Makinist, E. Akin, A. Yilmaz, "Estimating driver behavior by a smartphone", *IEEE IV*, 2012.

[6] iOnRoad mobile app. http://www.ionroad.com

[7] C. You, N. D. Lane, F. Chen et al., "CarSafe App: Alerting Drowsy and Distracted Drivers using Dual Cameras on Smartphones", *ACM MobiSys*, 2013.

[8] Q. Ji, X. Yang, "Real-Time Eye, Gaze, and Face Pose Tracking for Monitoring Driver Vigilance", *Real-Time Imaging 8*, 357–377, 2002.

[9] E. Murphy-Chutorian, M. Trivedi, "Head Pose Estimation in Computer Vision: A Survey", *IEEE T-PAMI*, vol. 31, no. 4, pp. 607–626, 2009.

[10] E. Murphy-Chutorian, A. Doshi, M. M. Trivedi, "Head Pose Estimation for Driver Assistance Systems: A Robust Algorithm and Experimental Evaluation", *IEEE ITSC*, 2007.

[11] D. Witzner Hansen, Q. Ji, "In the Eye of the Beholder: A Survey of Models for Eyes and Gaze", *IEEE T-PAMI*, vol. 32, no. 3, pp. 478–500, 2010.

[12] M. Reale, T. Hung, L. Yin, "Pointing with the eyes: gaze estimation using a static/active camera system and 3d iris disk model", *IEEE ICME*, *2010*.

[13] L. Fletcher, A. Zelinsky, "Driver inattention detection based on eye gaze-road event correlation", *Int. J. Robotics Research*, vol. 28, no. 6, pp. 774–801, 2009.

[14] P. Viola, M. Jones, "Rapid object detection using a boosted cascade of simple features," *IEEE CVPR*, 2001.

[15] A. Dempster, A., N. Laird, D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Royal Stat. Soc.*, vol. 39, no. 1, pp. 1-38, 1977.

[16] J. Ren, N. Kehtarnavaz, L. Estevez, "Real-time optimization of Viola-Jones face detection for mobile platforms," *IEEE CAS Workshop: SoC - Design, Applications, Integration, and Software*, 2008.