Vision-Based Autonomous Path Following Using a Human Driver Control Model With Reliable Input-Feature Value Estimation

Kazuhide Okamoto, *Student Member, IEEE*, Laurent Itti, and Panagiotis Tsiotras, *Fellow, IEEE*

***Abstract*—Autonomous vehicles differ from many other autonomous systems in the sense that these vehicles have to share the environment with humans (e.g., pedestrians, cyclists, and other drivers in traffic). This requirement poses a challenging perception and planning problem. Our research focuses on the problem of a vehicle sharing the environment with a human passenger/driver and, specifically, of designing a controller that captures the natural tendencies of the human driver so as to guarantee that the resulting control action is comparable to that of the human. In this paper, we propose to use a human driver control model into the autonomous vehicle control framework, which has previously been shown to predict short-term driver actions, and we develop an approach that reliably and accurately estimates input-feature values from driver-point-of-view images. After the feature-input values have been estimated, the human driver control model computes the corresponding steering-wheel angle. We, thus, provide more structure to the overall processing pipeline, compared to the recent end-to-end approaches. The proposed approach is validated using numerical simulations with synthetic images. The results validate the importance of combining traditional structured (e.g., transfer function) models with parsimonious neural-network representations.**

***Index Terms*—Intelligent vehicles, autonomous vehicles, Bayes methods, vehicle driving.**

I. Introduction

AUTONOMOUS driving has been actively investigated in the literature to improve road safety [1]–[3]. The current control framework of autonomous vehicles assumes that a driving strategy is feasible and thus the vehicle can execute this strategy. However, if the behavior of a self-driving vehicle is significantly different from "normal" human driving, human passengers may feel unsafe or uncomfortable. This situation is especially relevant when a self-driving vehicle does not behave similarly to the way a human driver would respond in a similar situation. One way to solve this problem is to design a controller that captures the natural tendencies of human drivers and guarantees that the resulting control action is comparable to that of a human driver. To this end, we propose to use a human-driver-control model within the control loop. This model captures the part of steering command generation from certain feature points in the image. Several psychological studies have shown that these features are used extensively by human subjects while driving. Using this model we are therefore able to predict human driver behavior based on the observed scene and use this information to design controllers that take that behavior into consideration. Ideally, for a purely autonomous vehicle the controller will try to "mimic" the normal driving behavior of the human driver, thus leading to a more safe, predictable, and comfortable ride.

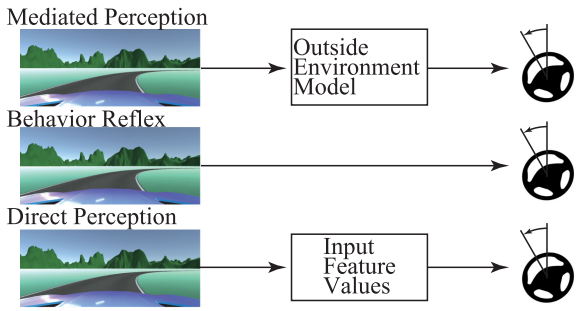


Fig. 1. Comparison of self-driving vehicle control approaches.

A. Related Research

According to [4], autonomous driving systems can be categorized into three groups: *mediated perception* (MP), *behavior reflex* (BR), and *direct perception* (DP), as illustrated in Fig. 1.

1) Mediated Perception: In the MP approach, the system first recognizes relevant objects using its sensors and constructs a world representation of the environment including other vehicles [5], traffic lights [6], and road segments [7]. Based on this world representation, steering and acceleration commands are then computed. As the authors of [4] pointed out, because only a small amount of the detected objects affect the resulting control actions, and since the precise reconstruction of the

Manuscript received March 12, 2018; revised August 31, 2018 and December 15, 2018; accepted January 16, 2019. Date of publication May 28, 2019; date of current version August 23, 2019. This work was supported by NSF Award CPS-1544814. The work of K. Okamoto was supported in part by Funai Foundation for Information Technology and in part by Ito Foundation USA-FUTI Scholarship. *(Corresponding author: Kazuhide Okamoto.)*

K. Okamoto is with the School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0150 USA (e-mail: kazuhide@gatech.edu).

L. Itti is with the Department of Computer Science, University of Southern California, Los Angeles, CA 90089-0781 USA (e-mail: itti@usc.edu).

P. Tsiotras is with the School of Aerospace Engineering and the Institute for Robotics and Intelligent Machines, Georgia Institute of Technology, Atlanta, GA 30332-0150 USA (e-mail: tsiotras@gatech.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIV.2019.2919476

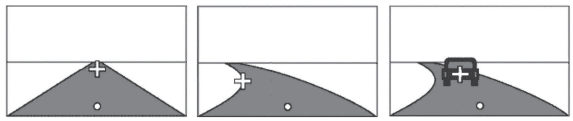
environment using sensor detection is still challenging [8], the MP approaches may lead to unnecessarily complex and costly systems. In addition, MP approaches do not guarantee, in general, that the resulting control actions will be similar to human driving, and, as a result, human passengers may feel unsafe and/or uncomfortable.

2) Behavior Reflex: The BR approach utilizes artificial neural networks trained to directly map sensor inputs to control actions. This end-to-end learning has attracted a lot of attention since the 80's [9], [10]. LeCun *et al.* [11] demonstrated the ability of an end-to-end approach to avoid obstacles while driving. Recently, Bojarski *et al.* [12] revisited this idea using a deep convolutional neural network (CNN). Because the ground truth of the control actions is typically provided from actual human driving, BR approaches try to imitate human driving (see e.g., [13]–[15]).

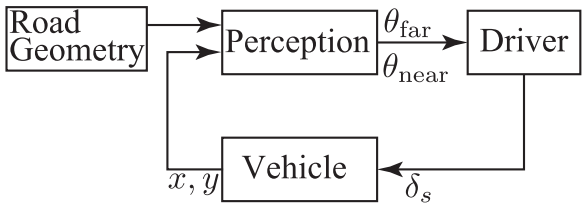
While BR approaches are elegant and attractive, the problem of mapping from a driver-view image to a steering-wheel angle is ill-posed owing to the variability in human-driver decision-making [4]. In addition, developing a controller in an end-to-end fashion lacks the interpretability of the intended rationale behind the applied control actions. This interpretability issue is a common problem with deep neural networks (DNNs) and is currently an active area of research [16], [17].

3) Direct Perception: The DP approach has recently been developed to overcome the drawbacks of the MP and BR approaches. In the DP approach, the system first maps the input images to a feature-value vector, or affordance, which then becomes the input to the controller [18], [19]. Chen *et al.* [4] proposed to train a CNN to map a driver-view image to several feature values of a traffic situation such as the yaw angle, the position, and the relative distance to other vehicles. Such estimates of feature values have been actively investigated. For instance, Hadsell *et al.* [20] developed a terrain classifier that estimates the traversability of the road surface from input images. Wulfmeier *et al.* [21] constructed the cost map of the surrounding environment from LIDAR point clouds.

Our proposed method also falls into the DP category, as we exploit input features for a visual driver control model, which then processes these features to compute the corresponding steering wheel angle. Human-driver control models have been actively researched since 1930's [22], with the aim of understanding human intentions while driving [23], predicting human-driver-control inputs [24], and reducing driving work-load [25]. See [26] for a recent overview. One such driver control model that has received attention recently is the two-point visual driver control model (TPVDCM) [27]–[29]. Several simulation and experimented studies have shown that the path-following behavior of the TPVDCM correlates with actual human driving commands. Because its inputs, namely, the view-ahead angles, are derived from observing actual human driving [30], TPVDCM keeps the interpretability of its output. The model is based on transfer functions, which have been well studied by control engineers. By incorporating such a control-theoretic and interpretable driver-control model within the control framework, we utilize the legacy of the human-driver control-modeling research, and, partially at least, remedy the interpretability issue of previous BR approaches.



(a) Descriptions of the view-ahead points depending on the driving situation; from [27].



(b) TPVDCM framework.

Fig. 2. Two-point visual driver control model.

B. Two-Point Visual Driver Control Model

This work uses the TPVDCM, originally developed in [27], to explicitly model human steering behavior. Although the structure of the model is quite simple, the behavior of the model aligns well with actual human driving behavior. Later, human sensorimotor activity was incorporated into the model [28]. Based on this model, an advanced driver assist system (ADAS) was developed in [25]. Recently, the authors in [29] used a model predictive controller in the anticipatory control channel of the sensorimotor TPVDCM to better model the high-level cognitive tasks of human drivers.

Figure 2(a) illustrates the basic idea and the typical geometric definition of the two view-ahead points used in the TPVDCM. The plus symbol represents the far view-ahead point, and the dot symbol represents the near view-ahead point. The TPVDCM framework is shown in Fig. 2(b). The "perception" subsystem computes the view-ahead angles θ_{far} and θ_{near} based on the relative road geometry with respect to the vehicle pose (see Figs. 4 and 5). The angle θ_{far} is the angle between the vehicle longitudinal axis and a far view-ahead point, which, while cornering, is placed at the road inner edge and at the road center at a certain distance ahead of the vehicle while driving straight. The angle θ_{near} is the angle between the vehicle longitudinal axis and a near view-ahead point, placed at a short distance ahead of the vehicle. The angle θ_{far} informs of the upcoming road trajectory and helps compensate the control action to cope with this trajectory change, while the angle θ_{near} maintains the current vehicle position close to the road centerline. After the values of θ_{far} and θ_{near} are obtained, the "driver" subsystem computes the steering-wheel angle δ_s, the input to the "vehicle" subsystem, which outputs the vehicle position (x, y).

C. Proposed Framework

In order to use the TPVDCM model to steer the vehicle, we need to estimate suitable input-feature values from driver-point-of-view images taken from a front-facing camera. Since the angles θ_{far} and θ_{near} are defined from the relative road geometry with respect to the current vehicle pose, it is not straightforward to compute θ_{far} and θ_{near} directly from the driver-point

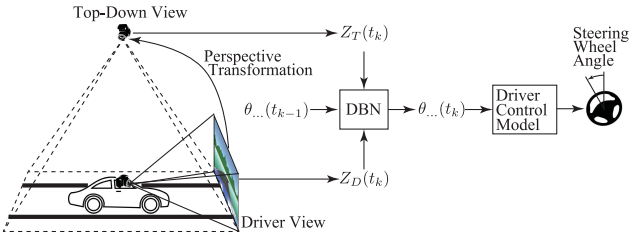


Fig. 3. Proposed approach.

-of-view image. Therefore, when using a TPVDCM, we generally assume some prior knowledge about the road geometry and the vehicle position [25]. This assumption has made human-driver-visual-control models impractical for many actual vehicle-control tasks.

Recently, CNNs [31] have been used extensively in generating steering commands from images while driving, using a so-called "end-to-end" approach [9], [10]. CNNs have succeeded in many challenging pattern recognition tasks [5], [32]. While CNNs are powerful tools and can be applied on a wide range of image-recognition problems, they need a large amount of training data, and it is difficult to know a priori how much or what kind of data is enough to train a CNN for autonomous driving tasks.

In order to reliably and accurately estimate the input-feature values from driver-point-of-view images, in this work we propose a new architecture, which is inspired from aircraft autopilot systems, that make their output more reliable by adding redundancy, i.e., by using multiple and separate ways to compute the control commands [33]. Specifically, in addition to a CNN, we propose to use a traditional image-processing approach and probabilistically combine the two estimates of θ_{far} and θ_{near} using a dynamic Bayesian network (DBN) [34]. Although several researchers have attempted to combine physics-based models and machine-learning approaches [19], [35], our approach is different from these works in the sense that while these works use CNNs to compute affordances and use physics-based models for control, here we use both CNNs and physics-based models to compute affordances. Our approach is more similar to the work of Peters and Itti [36], who proposed to combine the learning-based (top-down) and the saliency-model-based (bottom-up) predictions to estimate the eye positions of a human subject while playing a driving video game. The top-down prediction is based on a machine-learning method and requires training data to tune its parameters, while the bottom-up model uses a pre-defined saliency-map. In our proposed approach, the CNN corresponds to the "top-down" prediction, and the classical image-processing method corresponds to the "bottom-up" approach.

Figure 3 illustrates the proposed approach. We first use a perspective transformation and generate a top-down view of the road from the image taken using a front-looking camera. This approach has been utilized in automobile applications such as road-surface-traffic-sign recognition [37]. Based on the generated top-down view, the values of θ_{far} and θ_{near} are estimated We denote the estimate of these two angles using the top-down-view geometry as Z_T.

In order to add reliability to the estimation of the view-ahead angles, at the same time we process the driver-point-of-view image using a CNN. Let the output of the CNN be denoted as Z_D. We then probabilistically combine the estimates Z_T and Z_D using a DBN, allowing the use of time-dependent variables. A number of researchers [23], [38]–[41] have utilized DBNs and hidden Markov models (HMMs) to model/predict the behavior of human drivers.

D. Contribution

The contribution of this work lies in the incorporation of a human-driver control model into the image-based autonomous vehicle control framework. By doing so, we aim at achieving autonomous path-following behavior similar to human driving. While in our previous work [42] we tried to predict human driver control actions using data-driven human driver lateral control models, in this work we are primarily interested in controlling the vehicle. To this end, we develop a method to reliably and accurately estimate the input-feature view-ahead angle values. Since view-ahead angles are not defined in the driver-point-of-view images, we use a DBN to probabilistically combine and augment the estimate from driver-point-of-view images using top-down-view images. In the proposed framework, both these two approaches can be regarded as "virtual" sensors that output the measurements of certain variables. Combining multiple sensor information has been extensively investigated using Kalman filters [43], unscented filters [44], and particle filters [45], all of which are special cases of DBNs. To our knowledge, this work is the first attempt to formally incorporate the outputs of CNNs into a sensor-fusion framework.

II. Details of Proposed Approach

This section details the key components of the proposed approach. The key innovation is to incorporate the TPVDCM [27] in the control and perception processing pipeline. While recent TPVDCMs [28], [29] use the steering-wheel *torque* as the output, the original TPVDCM [27] outputs the steering wheel *angle*. In order to directly compare the behavior and performance of the proposed approach with an end-to-end approach in Section III, this work uses the original model. As illustrated in Fig. 3, in order to materialize the proposed methodology, we need to estimate: a) the view-ahead angles from driver-view images; and b) the parameters of the driver control model. The proposed approach combines the feature extraction step (from driver-view images to θ_{far} and θ_{near}) with the control input estimation step (from θ_{far} and θ_{near} to δ_s), allowing for a better understanding of the rationale behind the control actions of the vehicle.

A. View-Ahead Angle Estimation

We first introduce an algorithm to estimate θ_{near} and θ_{far} from the driver-point-of-view images. There are two distinct pipelines as shown in Fig. 3. One is based on the driver-point-of-view images, and the other one processes the top-down-view images.

1) Driver-View Pipeline: We directly process the driver-view image using a CNN and estimate the view angles. Although view-ahead angles are not defined in the driver-view image, by

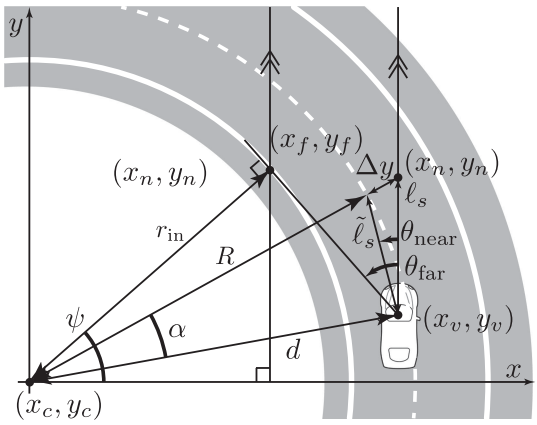


Fig. 4. Illustration of θ_{far} and θ_{near} when the road is curved.

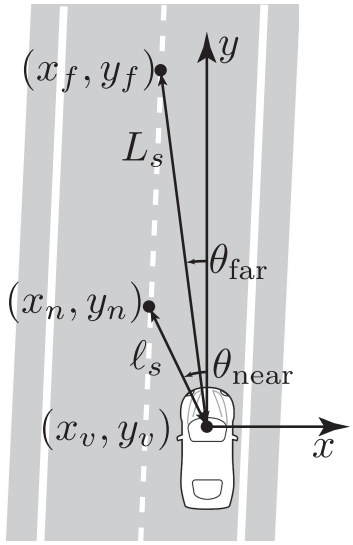


Fig. 5. Illustration of θ_{far} and θ_{near} when the road is straight.

giving the ground truth of the view-ahead angles, we train the CNN in a supervised-learning framework and estimate the values of the view-ahead angles Z_D. While deep learning-based approaches have exhibited superior performance than classical image-processing methods over the past few years [46], Z_D may overfit to the training data set. To remedy this problem and in order to accurately and reliably estimate the view-ahead-angle values, we utilize a DBN that probabilistically combines Z_T and Z_D. This additional processing step enables us to consider the time evolution of the view-ahead angles. We expect that this process will contribute to mitigating the overfitting of the CNN-based estimation and augment the classical image-processing-based estimation.

2) Top-Down-View Pipeline: We also process the reconstructed top-down view images. As the definition of the angles θ_{far} and θ_{near} are given in the relative road geometry with respect to the vehicle pose, this top-down view image pipeline is more straightforward than the driver-view pipeline. We use a classical image processing approach to find the values θ_{far} and θ_{near}, consisting of two steps. The first step deals with the road-geometry detection to estimate the centerline of the road. The second step recognizes (depending on the road geometry) whether the road is part of a circle or a straight line. Once the road centerline is identified, and in order to detect the road radius, we perform a least-squares regression to a circle or a straight line.

B. Visual Angle Identification

Once the road geometry is identified, we compute the values of θ_{near} and θ_{far}. In [25], the values of the view-ahead angles are not directly measured nor estimated, but derived from the road kinematics. This approach is subject to various errors such as modeling errors in vehicle dynamics. Therefore in this work, after we obtain the relative road geometry, we derive the values of θ_{near} and θ_{far} using a geometric approach as follows.

Referring to Fig. 4 for the computation of θ_{far}, we first assume that the coordinate of the road curve center is (x_c, y_c), the road inner-edge curve radius is r_{in}, and the distance from the vehicle center of mass to the road center is d. We may represent the coordinates of the far view-ahead point as

$$x_f = x_c + r_{\text{in}} \cos\psi = x_v - \sqrt{d^2 - r_{\text{in}}^2}\sin\theta_{\text{far}},$$

$$y_f = y_c + r_{\text{in}} \sin\psi = y_v + \sqrt{d^2 - r_{\text{in}}^2}\cos\theta_{\text{far}},$$

where (x_v, y_v) is the vehicle coordinate, and ψ is the angle of the far view ahead point with respect to the road center. Note that the y axis of the top-down view image is always aligned with the vehicle longitudinal axis. Thus, $\theta_{\text{far}} = \psi$. Therefore,

$$\theta_{\text{far}} = \tan^{-1}\left(\frac{\sqrt{d^2 - r_{\text{in}}^2}(x_c - x_v) + (y_c - y_v)r_{\text{in}}}{r_{\text{in}}(x_c - x_v) - (y_c - y_v)\sqrt{d^2 - r_{\text{in}}^2}}\right).$$

Figure 4 also illustrates the geometry to calculate θ_{near}. Here, $(x_n,\, y_n)$ is the coordinate of the near view ahead point. Since $x_n = x_v, y_n = y_v + \ell_s$, and letting $\boldsymbol{r}_{cv} = [x_v - x_c, y_v - y_c]^\top$ and $\boldsymbol{r}_{cn} = [x_n - x_c, y_n - y_c]^\top$, $\alpha = \cos^{-1}((\boldsymbol{r}_{cv} \cdot \boldsymbol{r}_{cn})/\|\boldsymbol{r}_{cv}\|\|\boldsymbol{r}_{cn}\|)$. Thus,

$$\tilde{\ell}_s = \sqrt{R^2 + \|\boldsymbol{r}_{cv}\|^2 - 2R\|\boldsymbol{r}_{cv}\|\cos\alpha}.$$

Finally, we compute θ_{near} as

$$\theta_{\text{near}} = \cos^{-1}\left((\ell_s^2 + \tilde{\ell}_s^2 - \Delta y^2)/(2\ell_s\tilde{\ell}_s)\right),$$

where $\Delta y = \|\boldsymbol{r}_{cn}\| - R$.

When the road geometry is a straight line as in Fig. 5, the near view point (x_n, y_n) and the far view point (x_f, y_f) are at a fixed distance ℓ_s and L_s ahead of the vehicle, respectively, and we simply compute θ_{near} and θ_{far} from

$$\theta_{\text{near}} = \tan^{-1}\left(\frac{y_n - y_v}{x_n - x_v}\right), \quad \theta_{\text{far}} = \tan^{-1}\left(\frac{y_f - y_v}{x_f - x_v}\right).$$

The view-ahead angles are related to the so-called vanishing points. Methods to detect the vanishing points may be applicable to the estimation of the view-ahead angles. We refer the reader to [47], [48].

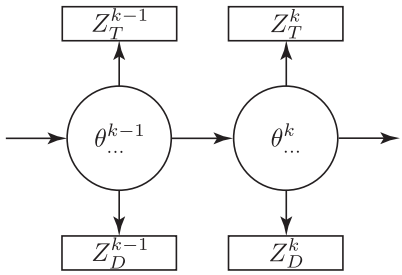


Fig. 6. The structure of the proposed DBN.

C. Driver-View-Image-Based Pipeline

Besides processing the top-down-view images, as outlined in Section II-B, we also use a CNN to directly process the driver-point-of-view images. The benefit of using a CNN is that we do not have to deal with the geometry of the driver-view images or their underlying kinematics. However, we need a large and diverse amount of training data to achieve good performance. Since our main research focus is not the development of a new CNN structure but rather augmenting the estimates using a separately and independently designed approach, we borrow the network proposed in [12] and train two networks separately, one for $\theta_{\rm near}$ and $\theta_{\rm far}$.

D. Dynamic Bayesian Network

Classical image-processing-based approaches do not need a lot of data to find good parameters. Nonetheless, as shown in other pattern recognition tasks [46] such approaches may perform worse than CNN-based approaches, which, however, have interpretability issues and require a large and diverse amount of data to train. In order to achieve both accurate and reliable predictions of $\theta_{\rm far}$ and $\theta_{\rm near}$, in this work we propose to utilize a DBN to probabilistically combine the two estimates.

As illustrated in Fig. 6, the proposed DBN has a latent variable $\theta_{\cdots} \in \{\theta_{\rm near}, \theta_{\rm far}\}$, along with the top-down-view-based observations Z_T, and the driver-view-based observations Z_D. Based on the structure of the network we probabilistically compute the latent variable at time step t_k given the observations, $Z_T^{1:k}$ and $Z_D^{1:k}$ [*] using Bayes rule and conditional independence conditions between the observations and $\theta_{\cdots}$. Specifically, in the proposed network structure, the following conditional independence conditions hold

$$\left\{Z_T^k, Z_D^k\right\} \perp\!\!\!\perp \left\{Z_T^{1:k-1}, Z_D^{1:k-1}\right\} | \theta_{\cdots}^k, \tag{1}$$

$$Z_T^k \perp\!\!\!\perp Z_D^k | \theta_{\cdots}^k, \tag{2}$$

$$\theta_{\cdots}^k \perp\!\!\!\perp \left\{Z_T^{k-1}, Z_D^{k-1}\right\} | \theta_{\cdots}^{k-1}. \tag{3}$$

These conditions and the Bayes rule lead to the following expression for the conditional probability for $\theta_{\rm near}$ and $\theta_{\rm far}$ given the estimates

$$P(\theta_{\cdots}^k | Z_T^{1:k}, Z_D^{1:k}) = P(Z_T^k | \theta_{\cdots}^k) P(Z_D^k | \theta_{\cdots}^k) \times \\ \times \int P(\theta_{\cdots}^k | \theta_{\cdots}^{k-1}) P(\theta_{\cdots}^{k-1} | Z_T^{1:k-1}, Z_D^{1:k-1}) \mathrm{d}\theta_{\cdots}^{k-1}. \tag{4}$$

[*] $Z_{\cdots}^{1:k} = [Z_{\cdots}^1, \ldots, Z_{\cdots}^k]$

Note that, in order to simplify the notation we used superscripts to represent the time step, e.g., $\theta_{\cdots}^k = \theta_{\cdots}(t_k)$.

E. Steering Angle Reproduction

In this section we introduce a method to identify the TPVDCM parameters. Parameter identification techniques for linear dynamical systems are well understood. For the case of the sensorimotor TPVDCM, the prediction error method [28] and nonlinear Kalman filters [49], [50] can estimate the model parameters. Here we use the original TPVDCM [27], expressed as a proportional-integral controller as follows

$$\delta_s(t) = k_f \theta_{\rm far}(t) + k_n \theta_{\rm near}(t) + k_I \int_{t_0}^{t} \theta_{\rm near}(s)\, \mathrm{d}s, \tag{5}$$

where k_f, k_n, and k_I are the control gains to be identified. In order to identify the parameters, we first approximate the integral term as $\int_{t_0}^{t} \theta_{\rm near}(s) \mathrm{d}s \simeq \sum_{s=t_0}^{t} \theta_{\rm near}(s) \Delta t$, and then use least-squares to find the parameters from a series of measurements.

Remark 1: The original model in [27] outputs the steering *angle*, which is the result of the interaction between the human driver and the steering-wheel dynamics. Thus, and in order to be more precise, this should have been denoted as a "driver & steering-wheel" system. However we call this as the "driver" model in order to simplify the terminology.

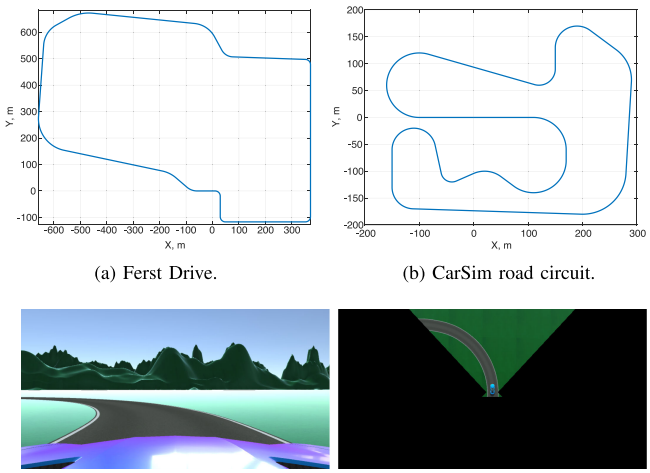
III. Numerical Simulations

In Section II, we introduced an algorithm to estimate $\theta_{\rm near}$ and $\theta_{\rm far}$ using driver-view and top-down-view images. In this section, we verify the proposed algorithm via numerical simulations. To this end, we developed a vehicle simulator using the Unity3D software [51]. In our simulation environment, instead of using perspective transformations, we place a camera directly above the vehicle to obtain the top-down view as we do not have access to the camera parameters. We implemented CNNs using TensorFlow [52] and used MATLAB Image Processing Toolbox to process the top-down view images.

In general, and due to the variation in human-driver decision making, the end-to-end approach from a driver-view image to a steering-wheel angle is ill-posed [4]. However, in our simulations we restricted the task to just road-center following. Thus, the mapping from a driver-view image to a steering-wheel angle is *not* ill-posed. Determining the path to follow (e.g., obstacle avoidance) is the task of a higher level path planner and is outside the scope of this work. Before discussing the results, we briefly explain each subsystem shown in Fig. 2 for this setup.

A. Road Geometry

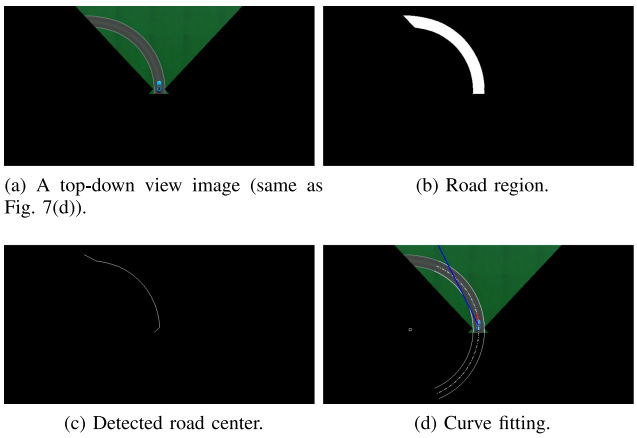
We use two road courses to check the performance of the proposed approach. One is the Ferst Drive at the Georgia Tech campus, and the other one is a CarSim pre-installed road circuit (Fig. 7). We use the Ferst Drive road course for training and parameter tuning and the CarSim pre-installed road course for testing. The length of the training and the test road courses is 3.23 km and 2.24 km, respectively.



(a) Ferst Drive. (b) CarSim road circuit.

(c) A driver view image. (d) A top-down view image.

Fig. 7. (a) and (b): Road circuits. In both scenarios, the vehicle starts from $(0, 0)$ and heads toward the $(-1, 0)$ direction. (c) and (d): A sample of the driver and top-down view images.



(a) A top-down view image (same as Fig. 7(d)). (b) Road region.

(c) Detected road center. (d) Curve fitting.

Fig. 8. The process of detecting the view-ahead angles from the top-down view image Fig. 7(d).

B. Perception

The perception subsystem outputs the view-ahead angles θ_{far} and θ_{near} based on the road geometry and the vehicle position. Next, we describe the top-down-view-based road geometry detection method and the DBN used in this work.

1) Top-Down-View-Based Road Geometry Detection: In order to identify the road geometry, we detect the road centerline from a top-down view image. To this end, we conduct the following steps: a) road region detection; b) morphological operation; and c) density-based clustering. Figure 8 illustrates how we process the image shown in Fig. 7(d). In the first step, we convert the original RGB image to hue, saturation, and value (HSV) coordinates. Then, we create a binary road mask image $\mathcal{RM}$, the pixel of which is 1 if the pixel is in the road region and 0 otherwise. Since the road regions have lower saturation values than the other regions in the top-down-view images, we simply set a threshold ϵ_S and assume that the pixel (i, j) is in the road region, i.e., if $Sat(i, j) < \epsilon_S$, $\mathcal{RM}(i, j) = 1$, otherwise, $\mathcal{RM}(i, j) = 0$, where $Sat(i, j) \in [0, 1]$ is the saturation value of the original top-down view at (i, j) (Fig. 8(a)).

In the second step, we dilate and erode $\mathcal{RM}$ to eliminate noise. Dilation eliminates small false-detected road regions, eroding regenerates large-enough true road regions. We perform this operation using the disk-shaped morphological structuring element with radius 10 pixels. In addition, we perform another morphological operation to convert the detected road region to a line (see Fig. 8(b)).

As a last step, we use DBSCAN [53] to further eliminate any falsely-detected road regions. The cluster which is closest to the vehicle's coordinate represents the observed road-center-line-points (x_o, y_o). The detected points are then processed to find the best-fit circle or straight line (Fig. 8(c)).

2) Discrete Dynamic Bayesian Network: We use a discrete DBN with discretization interval 0.1 deg, because it is simple and does not need a Gaussian noise assumption. The resulting DBN is an HMM with multiple measurements. Since the latent variables (θ_{far} and θ_{near}) are discrete and the ground-truth is available in our simulation environment, the training of the DBN consists of just counting the state transitions and measurement emissions and add them to matrices, which represent prior distributions. In order to reduce overfitting to the training data, we also applied a Gaussian filter with size 3×3 to the emission matrix of the CNN observations.

C. Driver

Based on the values of θ_{far} and θ_{near}, the driver subsystem computes the corresponding steering wheel angle δ_s. In order to control the vehicle, we employ Eq. (5). The output of the model is the steering-wheel angle $\delta_s = \delta / g_s$, where δ is the front-wheel angle and g_s is the gear ratio. We set the distance to the far and near view ahead points as $L_s = 15$ m and $\ell_s = 5$ m, respectively. The TPVDCM drives around the road shown in Fig. 7 at 10 m/s (36 kph). The speed setup was chosen based on the curvature of the corners of the road geometry.

Remark 2: Since we use simulation data, we did not have to estimate the values of L_s and ℓ_s. In the case of human-driving data, due to the definition of the view-ahead points [27], [30], we can estimate the values of L_s and ℓ_s by observing the gaze position of the human driver [36].

D. Vehicle

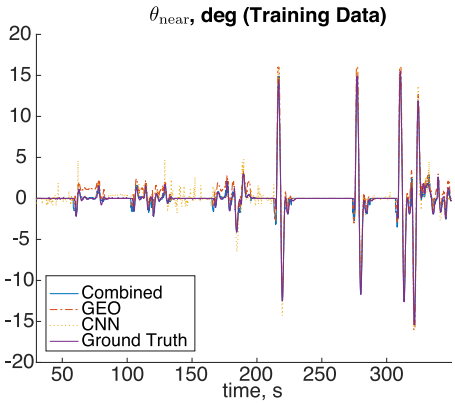
The vehicle subsystem updates the vehicle position based on the steering wheel angle δ_s from the driver subsystem. The vehicle model is the linearized single-track model given below.

$$\dot{\beta} = -\frac{C_r + C_f}{mV_x}\beta + \left(-1 + \frac{\ell_R C_r - \ell_F C_f}{mV_x^2}\right) r + \frac{C_f}{mV_x}\delta,$$

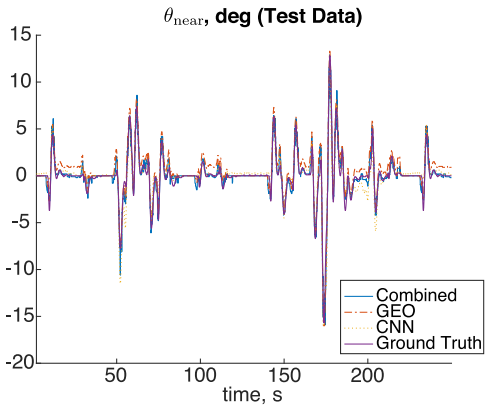
$$\dot{r} = \frac{\ell_R C_r - \ell_F C_f}{I_z}\beta + \left(-\frac{\ell_R^2 C_r + \ell_F^2 C_f}{I_z V_x}\right) r + \frac{\ell_F C_f}{I_z}\delta,$$

$$\dot{\phi} = r,$$

where β is the vehicle side-slip angle, r is the vehicle yaw rate, ϕ is the vehicle yaw angle, $V_x (= V \cos\beta)$ is the longitudinal component of the velocity of the vehicle center of mass in the vehicle frame, m is the vehicle mass, and I_z is the vehicle moment of inertia with respect to the mass center of the vehicle. Furthermore, ℓ_F and ℓ_R are the distances of the front and rear axles

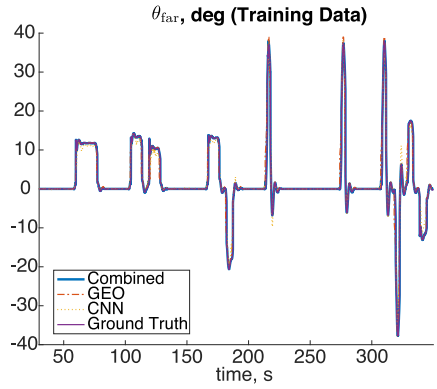


(a) Training data set.

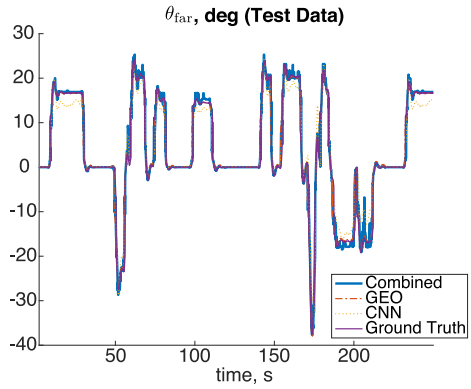


(b) Test data set.

Fig. 9. The results of θ_{near} estimations.



(a) Training data set.



(b) Test data set.

Fig. 10. The results of θ_{far} estimations.

TABLE I
RMSE OF θ_{near} (LEFT) AND θ_{far} (RIGHT), DEG

	Combined		GEO		CNN	
Training Data	**0.39**	**0.18**	0.75	2.13	0.81	1.54
Test Data	**0.58**	**2.06**	0.85	2.16	0.96	2.92

TABLE II
RMSE OF θ_{near}, DEG

	Training:8 m/s Test: 10m/s			Training:12 m/s Test: 10m/s		
	Comb.	GEO	CNN	Comb.	GEO	CNN
Train.	**0.39**	0.74	0.69	**0.38**	0.72	0.98
Test	**0.59**	0.85	0.96	**0.65**	0.85	0.96

TABLE III
RMSE OF θ_{far}, DEG

	Training:8 m/s Test: 10m/s			Training:12 m/s Test: 10m/s		
	Comb.	GEO	CNN	Comb.	GEO	CNN
Train.	**0.42**	2.12	2.06	**0.42**	1.99	1.90
Test	**2.11**	2.16	2.92	**2.14**	2.16	2.92

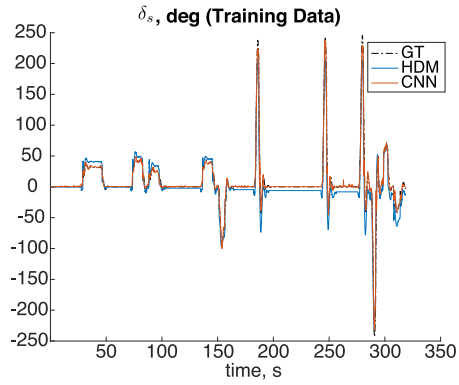
from the center of mass, respectively. In addition, C_f and C_r are the cornering stiffnesses of the front and rear tires, which we assume constant. Finally, the vehicle position is updated based on $\dot{x} = V\cos(\beta + \psi)$ and $\dot{y} = V\sin(\beta + \psi)$.

E. Results of View-Ahead Angle Estimation

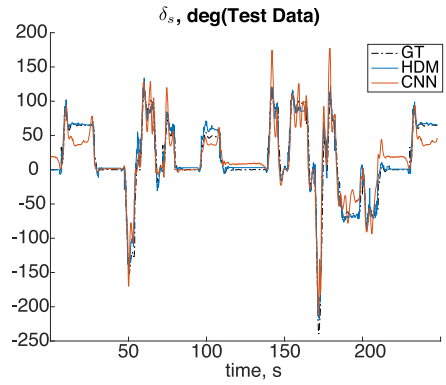
We collected 4,500 images for the training data set and 4,000 images for the test data set. Figures 9 and 10 show the results of the θ_{near} and θ_{far} estimates. The labels indicate the following: 1) Combined: proposed method; 2) GEO: the geometry-based, top-down-view approach, Z_T; 3) CNN: the CNN-based, driver-view approach, Z_D; 4) Ground Truth: the ground truth of the signal. Table I summarizes the root-mean-squared error (RMSE[1]) of each estimation algorithm. As shown in Table I, by probabilistically fusing the observations, the proposed approach exhibits less RMSE than both GEO and CNN.

We also inspected the sensitivity of these approaches to the vehicle speed. Tables II and III summarize the results. The left subtable lists the RMSE when the vehicle speed is set to 8 m/s (28.8 kph) for the training data set and 10 m/s (36 kph) for the test data set. The right subtable lists the RMSE for the case when the vehicle speed is set to 12 m/s (43.2 kph) for the training data set and 10 m/s for the test data set. The proposed approach

[1]$\text{RMSE}(\hat{\theta}_{\cdots}, \theta_{\cdots}) = \sqrt{1/N_\theta \sum_{k=1}^{N_\theta} (\hat{\theta}_{\cdots}(k) - \theta_{\cdots}(k))^2}$, where $\hat{\theta}_{\cdots}$ is the estimation, $\theta_{\cdots}$ is the ground truth, and N_θ is the number of observations.



(a) Training data set.



(b) Test data set.

Fig. 11. The result of steering angle reproduction.

TABLE IV
RMSE OF STEERING WHEEL ANGLE, DEG

	HDM	CNN
Training Data	12.52	**7.43**
Test Data	**11.14**	20.91

still shows less RMSE than the GEO and CNN thanks to the probabilistic augmentation of the estimation from the DBN.

F. Results of Steering-Angle Reproduction

This subsection discusses the results of steering angle reproduction. We compared the performance with an end-to-end learning-based approach, the network structure of which is borrowed from [12].

Figure 11 illustrates the results for training and test data sets.[2] In this figure 'GT' represents the ground truth and 'HDM' represents the output of the human driver model (with $k_f = 3.6$, $k_n = 4.7$, and $k_I = 0.8$) given the estimated values of $\theta_{\rm near}$ and $\theta_{\rm far}$ in Section III-E. Furthermore, 'CNN' is the output of the CNN, the structure of which is the same as [12] and is trained to estimate the steering-wheel angle using the same data in Section III-E.

As shown in Table IV, with the training data set, the CNN-based end-to-end approach exhibited less RMSE than the proposed approach, while with the test data set, the CNN-based approach exhibited greater RMSE. In contrast, the proposed approach exhibited sound performance with both the training and test data sets. This tendency is also observed when we check the sensitivity to speed listed in Table V. These results imply that by incorporating a model of the internal driver control processing pipeline, we achieve a more reliable system. By combining the outputs of the CNN and of the image-processing module using a DBN, we have successfully demonstrated an autonomous vehicle controller that is reliable and keeps the interpretability of the outputs.

[2]Videos are available for the Ferst Drive: https://youtu.be/f2ed44ThDeo and the CarSim course: https://youtu.be/mkJQsViB4yw

TABLE V
RMSE OF STEERING WHEEL ANGLE, DEG

	Training:8 m/s Test: 10m/s		Training:12 m/s Test: 10m/s	
	HDM	CNN	HDM	CNN
Training Data	12.43	**9.67**	11.53	**6.33**
Test Data	**10.89**	20.91	**11.68**	20.91

Remark 3: It may also be possible to generate driver control actions using a recurrent neural network (RNNs), such as a long-short-term memory (LSTM) network to estimate the view-ahead angles.[3] However, RNNs still have an interpretability issue, and thus, this work did not further explore the use of recurrent neural networks.

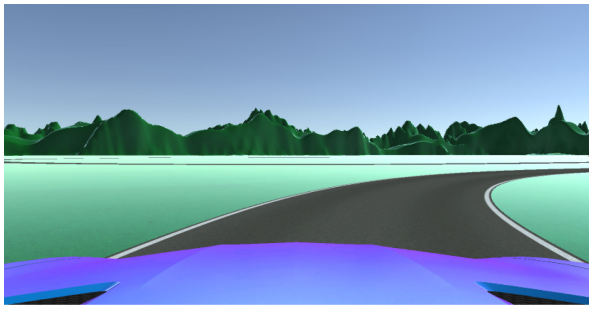
G. Additional Simulation With Adversarial Noise

In this section, we clarify the benefit of adding redundancy for estimating the view-ahead angles via DBN with multiple estimates. Since we do not have access to real-world images with ground truth of the view-ahead angle values, we use synthetic image data with adversarial noise [54], which is designed to make machine-learning algorithms misclassify. Image inputs with well-designed adversarial noise do not affect how humans recognize them, but affect the outputs of the machine learning algorithms. In some sense, this setting is more severe than the one using the real-world images, the noise of which is just random. We use the Fast Gradient Sign Method (FGSM) [55] with a different network structure than the one we used in the experiments above, assuming adversaries do not know the network structure [56]. In order to implement input driver-view images with adversarial noise, we used TensorFlow [52] and Cleverhans [57]. Figure 12 illustrates an example of adversarial noise to make the CNN misclassify $\theta_{\rm near}$. After the noise is added, the road is still recognizable to a human observer. Table VI shows the RMSE of the estimates with the CarSim road circuit data. By adding adversarial noise, the performance of the CNN got significantly worse (from 0.96 to 2.13). By compensating the CNN error with the GEO estimation using DBN, the performance degradation of the combined estimation is mitigated (1.22 compared to 2.13).

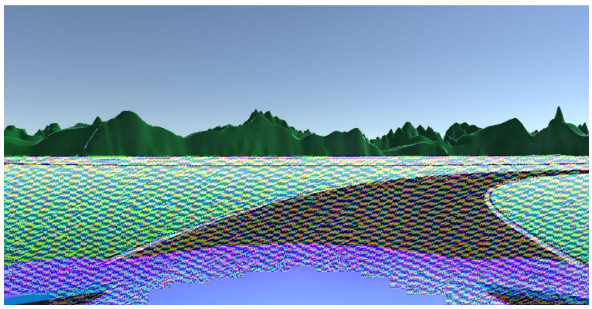
H. Future Real-World Implementation and Extensions

In this section we have validated the proposed algorithm using synthetic images. Before we are ready to apply the proposed approach to actual autonomous vehicles, it is imperative to also

[3]We thank an anonymous reviewer for this observation.



(a) Original driver-view image.



(b) Driver-view image with adversarial noise.

Fig. 12. Example driver-view images.

TABLE VI
RMSE OF θ_{near} UNDER ADVERSARIAL NOISE, DEG

	Combined	GEO	CNN
Without Noise	0.58	0.85	0.96
With Noise	1.22	0.85	2.13

validate it using data obtained from human drivers, as part of of our future work. The numerical results of this section show, however, that the idea of probabilistically combining the feature estimates is promising. Furthermore, as demonstrated previously in [36], this framework can accommodate other sophisticated methods that can deal with increased levels of noise and diverse illumination conditions. Finally, it is straightforward to increase the number of observations using the proposed approach. For instance, one can add another CNN that has a different structure or is trained using a different data set.

IV. SUMMARY

This paper proposed a new framework to control an autonomous vehicle by incorporating a human driver control model with the aim of achieving path-following behavior similar to that of a human driver. The driver model uses as input-features the values of two angles obtained from the driver-point-of-view images. To estimate these angles we used a DBN that probabilistically combined the outputs of driver-view image -based and top-down-view image-based approaches. The proposed method explicitly and separately solved the feature-extraction and the control-output-estimation problems, allowing to interpret the rationale behind the generated control actions. Using numerical simulations and high-fidelity synthetic image data, we have shown that the proposed approach leads to accurate and stable performance for a variety of driving scenarios. Future work will investigate the on-line control performance of the proposed method using more realistic test environments.

REFERENCES

[1] S. Lefèvre, D. Vasquez, and C. Laugier, "A survey on motion prediction and risk assessment for intellgent vehicles," *Robomech J.*, vol. 1, no. 1, pp. 1–14, 2014.

[2] A. Carvalho, S. Lefévre, G. Schildbach, J. Kong, and F. Borrelli, "Automated driving: The role of forecasts and uncertainty—A control perspective," *Eur. J. Control*, vol. 24, pp. 14–32, 2015.

[3] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Trans. Intell. Veh.*, vol. 1, no. 1, pp. 33–55, Mar. 2016.

[4] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *Proc. IEEE Int. Conf. Comput. Vis.*, Santiago, Chile, Dec. 11–18, 2015, pp. 2722–2730.

[5] B. Huval *et al.*, "An empirical evaluation of deep learning on highway driving," 2015, arXiv:1504.01716.

[6] M. Weber, P. Wolf, and J. M. Zöllner, "DeepTLR: A single deep convolutional network for detection and classification of traffic lights," in *Proc. IEEE Intell. Veh. Symp.*, Gothenburg, Sweden, Jun. 19–22, 2016, pp. 342–348.

[7] M. Teichmann, M. Weber, M. Zoellner, R. Cipolla, and R. Urtasun, "MultiNet: Real-time joint semantic reasoning for autonomous driving," in *IEEE Intelligent Vehicles Symp.*, Changshu, China, Jun. 26–28, 2018, pp. 1013–1020, arXiv:1612.07695.

[8] K. Ashraf, B. Wu, F. N. Iandola, M. W. Moskewicz, and K. Keutzer, "Shallow networks for high-accuracy road object-detection," 2016, arXiv:1606.01561.

[9] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, Denver, CO, USA, Nov. 27–30, 1989, pp. 305–313.

[10] D. A. Pomerleau, "Efficient training of artificial neural networks for autonomous navigation," *Neural Comput.*, vol. 3, no. 1, pp. 88–97, 1991.

[11] Y. LeCun, U. Muller, J. Ben, E. Cosatto, and B. Flepp, "Off-road obstacle avoidance through end-to-end learning," in *Proc. Adv. Neural Inf. Process. Syst.*, Vancouver, BC, Canada, Dec. 5–8, 2005, pp. 1–8.

[12] M. Bojarski *et al.*, "End to end learning for self-driving cars," 2016, arXiv:1604.07316.

[13] E. Rehder, J. Quehl, and C. Stiller, "Driving like a human: Imitation learning for path planning using convolutional neural networks," in *Proc. IEEE Int. Conf. Robot. Autom. Workshops*, Marina Bay, Singapore, May 29–Jun. 3, 2017, pp. 1–5.

[14] J. Ho and S. Ermon, "Generative adversarial imitation learning," in *Proc. Adv. Neural Inf. Process. Syst.*, Barcelona, Spain, Dec. 5–10, 2016, pp. 4565–4573.

[15] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Boston, MA, USA, Jun. 8–12, 2015, pp. 3431–3440.

[16] G. Montavon, W. Samek, and K.-R. Müller, "Methods for interpreting and understanding deep neural networks," *Digit. Signal Process.*, vol. 73, pp. 1–15, 2018.

[17] J. Kim and J. Canny, "Interpretable learning for self-driving cars by visualizing causal attention," in *IEEE Int. Conf. Comput. Vision*, Venice, Italy, Oct. 22–29, 2017, pp. 2961–2969, arXiv:1703.10631.

[18] L. Caltagirone, M. Bellone, L. Svensson, and M. Wahde, "Simultaneous perception and path generation using fully convolutional neural networks," 2017, arXiv:1703.08987.

[19] P. Drews, G. Williams, B. Goldfain, E. A. Theodorou, and J. M. Rehg, "Aggressive deep driving: Model predictive control with a CNN cost model," in *Conf. on Robot Learning*, Mountain View, CA, USA, Nov. 13–15, 2017, pp. 133–142, arXiv:1707.05303.

[20] R. Hadsell *et al.*, "Learning long-range vision for autonomous off-road driving," *J. Field Robot.*, vol. 26, no. 2, pp. 120–144, 2009.

[21] M. Wulfmeier, D. Z. Wang, and I. Posner, "Watch this: Scalable cost-function learning for path planning in urban environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, Daejeon, South Korea, Oct. 9–14, 2016, pp. 2089–2095.

[22] J. J. Gibson and L. E. Crooks, "A theoretical field-analysis of automobile-driving," *Amer. J. Psychol.*, vol. 51, no. 3, pp. 453–471, 1938.

[23] A. Pentland and A. Liu, "Modeling and prediction of human behavior," *Neural Comput.*, vol. 11, no. 1, pp. 229–242, 1999.

[24] H. Wei, W. Ross, S. Varisco, P. Krief, and S. Ferrari, "Modeling of human driver behavior via receding horizon and artificial neural network controllers," in *Proc. IEEE Annu. Conf. Decis. Control*, Florence, Italy, Dec. 10–13, 2013, pp. 6778–6785.
[25] S. Zafeiropoulos and P. Tsiotras, "Design of a lane-tracking driver steering assist system and its interaction with a two-point visual driver model," in *Proc .Amer. Control Conf.*, Portland, OR, USA, Jun. 4–6, 2014, pp. 3911–3917.
[26] P. C. Cacciabue, *Modeling Driver Behavior in Automotive Environments: Critical Issues in Driver Interactions With Intelligent Transport Systems*. London, U.K.: Springer-Verlag, 2007.
[27] D. D. Salvucci and R. Gray, "A two-point visual control model of steering," *Perception*, vol. 33, no. 10, pp. 1233–1248, 2004.
[28] C. Sentouh, P. Chevrel, F. Mars, and F. Claveau, "A sensorimotor driver model for steering control," in *Proc. Int. Conf. Syst., Man Cybern.*, San Antonio, TX, USA, Oct. 11–14, 2009, pp. 2462–2467.
[29] K. Okamoto and P. Tsiotras, "A new hybrid sensorimotor driver model with model predictive control," in *Proc. Int. Conf. Syst., Man, Cybern.*, Budapest, Hungary, Oct. 9–12, 2016, pp. 1866–1871.
[30] M. F. Land and D. N. Lee, "Where we look when we steer," *Nature*, vol. 369, no. 6483, pp. 742–744, 1994.
[31] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org
[32] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 1, pp. 142–158, Jan. 2016.
[33] Y. C. Yeh, "Triple-triple redundant 777 primary flight computer," in *Proc. IEEE Aerosp. Appl. Conf.*, Aspen, CO, USA, Feb. 3–10, 1996, vol. 1, pp. 293–307.
[34] K. Murphy, "Dynamic Bayesian networks: Representation, inference and learning," Ph.D. dissertation, Comput. Sci., University of California, Berkeley, Berkeley, CA, USA, 2002.
[35] C. Hubschneider *et al.*, "Integrating end-to-end learned steering into probabilistic autonomous driving," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, Yokohama, Japan, Oct. 16–19, 2017, pp. 2109–2115.
[36] R. J. Peters and L. Itti, "Beyond bottom-up: Incorporating task-dependent influences into a computational model of spatial attention," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Minneapolis, MN, Jun. 17–22, 2007, pp. 1–8.
[37] A. Kheyrollahi and T. P. Breckon, "Automatic real-time road marking recognition using a feature driven approach," *Mach. Vis. Appl.*, vol. 23, no. 1, pp. 123–133, 2012.
[38] N. Oliver and A. P. Pentland, "Driver behavior recognition and prediction in a SmartCar," *Proc. SPIE*, 2000, vol. 4023, pp. 280–290.
[39] Q. Ji, P. Lan, and C. Looney, "A probabilistic framework for modeling and real-time monitoring human fatigue," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 36, no. 5, pp. 862–875, Sep. 2006.
[40] N. Kuge, T. Yamamura, O. Shimoyama, and A. Liu, "A driver behavior recognition method based on a driver model framework," SAE Tech. Paper, Warrendale, PA, USA, 2000.
[41] T. Kumagai, Y. Sakaguchi, M. Okuwa, and M. Akamatsu, "Prediction of driving behavior through probabilistic inference," in *Proc. Int. Conf. Eng. Appl. Neural Netw.*, Malaga, Spain, Sep. 8–10, 2003, pp. 117–123.
[42] K. Okamoto and P. Tsiotras, "A comparative study of data-driven human driver lateral control models," in *Proc. Annu. Amer. Control Conf.*, Milwaukee, WI, USA, Jun. 27–29, 2018, pp. 3988–3993.
[43] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. Cambridge, MA, USA: MIT Press, 2005.
[44] E. A. Wan and R. V. D. Merwe, "The unscented Kalman filter for nonlinear estimation," in *Proc. IEEE Adaptive Syst. Signal Process., Commun., Control Symp.*, Lake Louise, AB, Canada, Oct. 1–4, 2000, pp. 153–158.
[45] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 174–188, Feb. 2002.
[46] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, Lake Tahoe, NV, USA, Dec. 3–8, 2012, pp. 1097–1105.
[47] C.-K. Chang, J. Zhao, and L. Itti, "Deep VP: Deep learning for vanishing point detection on 1 million street view images," in *Proc. Int. Conf. Robot. Autom.*, Brisbane, QLD, Australia, May 21–25, 2018, pp. 4496–4503.
[48] F. Kluger, H. Ackermann, M. Y. Yang, and B. Rosenhahn, "Deep learning for vanishing point detection using an inverse gnomonic projection," in *Proc. German Conf. Pattern Recognit.*, Basel, Switzerland, Sep. 13–15, 2017, pp. 17–28.
[49] C. You, J. Lu, and P. Tsiotras, "Driver parameter estimation using joint E-/UKF and dual E-/UKF under nonlinear state inequality constraints," in *Proc. Int. Conf. Syst., Man, Cybern.*, Budapest, Hungary, Oct. 9–12, 2016, pp. 1215–1220.
[50] C. You, J. Lu, and P. Tsiotras, "Nonlinear driver parameter estimation and driver steering behavior analysis for ADAS using field test data," *IEEE Trans. Human Mach. Syst.*, vol. 47, no. 5, pp. 686–699, Oct. 2017.
[51] Unity Technologies, "Unity3D." 2017 [Online]. Available: unity3d.com
[52] M. Abadi *et al.*, "TensorFlow: A system for large-scale machine learning," in *Proc. USENIX Symp. Oper. Syst. Des. Implementation*, Savannah, GA, USA, Nov. 2–4, 2016, pp. 265–283.
[53] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. Int. Conf. Knowl. Discovery Data Mining*, Portland, OR, USA, Aug. 2–4, 1996, pp. 226–231.
[54] C. Szegedy *et al.*, "Intriguing properties of neural networks," 2013, arXiv:1312.6199.
[55] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Int. Conf. Learning Representations*, San Diego, CA, USA, May 7–9, 2015, arXiv:1412.6572.
[56] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proc. ACM Asia Conf. Comput. Commun. Security.*, Abu Dhabi, UAE, Apr. 2–6, 2017, pp. 506–519.
[57] N. Papernot *et al.*, "Technical report on the cleverhans v2.1.0 adversarial examples library," 2018, arXiv:1610.00768.

Kazuhide Okamoto (S'18) received the bachelor's and master's degrees in aeronautics and astronautics from the University of Tokyo, Tokyo, Japan. He is currently working toward the doctoral degree in aerospace engineering at the Georgia Institute of Technology, Atlanta, GA, USA. He is researching stochastic optimal control and its application to human–vehicle systems. He was also a summer research intern with the Mitsubishi Electric Research Laboratories, Boston, MA, USA, and with Honda Research Institute USA, Mountain View, California.

Laurent Itti received the M.S. degree in image processing from the Ecole Nationale Superieure des Telecommunications, Paris, France, in 1994, and the Ph.D. degree in computation and neural systems from Caltech, Pasadena, CA, USA, in 2000. He was an Assistant Professor, an Associate Professor, and is currently a Full Professor of computer science, psychology, and neuroscience with the University of Southern California, Los Angeles, CA, USA. His research interests include biologically inspired computational vision, in particular in the domains of visual attention, scene understanding, control of eye movements, and surprise. This basic research has technological applications to, among others, video compression, target detection, and robotics. He has coauthored more than 150 publications in peer-reviewed journals, books and conferences, three patents, and several open-source neuromorphic vision software toolkits.

Panagiotis Tsiotras (F'19) received the Ph.D. degree in aeronautics and astronautics from Purdue University, West Lafayette, IN, USA, in 1993 and the Mathematics and Mechanical Engineering degree. He is a Dean Professor with the School of Aerospace Engineering, Georgia Institute of Technology (Georgia Tech), Atlanta, GA, USA, and the Director of the Dynamics and Controls Systems Laboratory (DCSL) in the same school as well as an Associate Director of the Institute for Robotics and Intelligent Machines, Georgia Tech. He has held visiting research appointments at MIT, JPL, INRIA Rocquencourt, and Mines ParisTech. His research interests include optimal control of nonlinear systems and ground, aerial and space vehicle autonomy. He has served in the Editorial Boards of the Transactions on Automatic Control, the IEEE CONTROL SYSTEMS MAGAZINE, the *AIAA Journal of Guidance, Control and Dynamics* and the *Journal Dynamics and Control*. He is the recipient of the NSF CAREER award and the Outstanding Aerospace Engineer award from Purdue. He is a Fellow of AIAA, and a member of the Phi Kappa Phi, Tau Beta Pi, and Sigma Gamma Tau Honor Societies.