A Comparative Study of Data-Driven Human Driver Lateral Control Models

Kazuhide Okamoto[1] and Panagiotis Tsiotras[2]

Abstract— **To reduce human driving workload, many advanced driver assist systems (ADAS) have been developed using a single, often simple, driver model to predict human-driver interaction in the immediate future. However, each person drives differently, necessitating personalized driver models based on data obtained from actual driver actions. Yet, traditional control-theoretic and physics-based models have difficulty in accurately predicting driver actions. Being inspired from the recent achievements of machine-learning (ML) methods, this work compares several ML-based algorithms in predicting the lateral control actions of human drivers, evaluates each method using both simulated and real human-driving data sets, and discusses their performance.**

I. Introduction

To increase road safety advanced driver assist systems (ADAS) have been investigated to support human drivers. In order to properly assist drivers, ADAS need to accurately predict the immediate actions of the drivers, necessitating a thorough investigation of human driver control models. However, unlike the modeling of physical systems such as vehicles, reliable modeling of human driver behavior is difficult.

The majority of modern control design methodologies are model-based, i.e., in order to design a control system for a vehicle, one needs first a good vehicle model that, depending on the complexity of the task, ranges from a simple one (e.g., point-mass model) to something more complex (e.g., rolling rigid vehicle model) [1]. The identification of the model parameters requires system-identification techniques such as Kalman filters. Modeling and system identification of human driver actions is, on the other hand, lacking similar well-established identification techniques. Although several physics-based models, such as the two-point visual driver control model (TPVDCM) [2]–[4], have been proposed over the years, it is challenging to accurately predict the human driver control actions using traditional modeling methods. Recently, machine-learning (ML) algorithms have succeeded in many challenging pattern-recognition and inference tasks. In this paper, we employ ML techniques to compare the prediction accuracy of the lateral control actions of human drivers in the immediate future.

Using ML algorithms to identify the intentions and actions of human drivers has been investigated for decades. In the late '90s, Pentland and Liu employed a hidden Markov model (HMM) to predict driver intentions [5]. Since then, HMMs have been the most popular methods to identify the intentions of human drivers. Recently, researchers have actively employed several other algorithms. Aoude et al. [6], for instance, developed a support-vector-machine (SVM)-based and an HMM-based algorithms to identify the intentions of human drivers at intersections. In addition, Chandrasiri et al. [7] classified driving skills using k-nearest neighbors (kNN) and an SVM. The work in [8], [9] proposed ways to decrease the computational cost for vehicle motion prediction in traffic using machine learning techniques. These works are all based on classification algorithms.

This paper compares the performance of regression algorithms. Learning control inputs from human demonstrations, called learning from demonstration (LfD), has been actively researched in the robotics community [10]. In the case of vehicle control, Armand et al. [11] employed a Gaussian Process (GP) regression method to model human-driver braking control, while Miyajima et al. [12] employed Gaussian Mixture Models (GMMs) to model longitudinal control action of human drivers.

While, to our knowledge, this work is the first attempt to compare data-driven lateral driving control models, several researchers conducted comparative studies of LfD performance of other tasks. Lefèvre et al. [13], for instance, compared longitudinal control models and found that simple parametric models are enough for short-term horizons, but non-parametric models outperform parametric ones for tasks with long-term prediction horizons. Also, Wei et al. [14] compared the performance between receding horizon controller (RHC)-based model and artificial neural network (ANN)-based model for the driver behavior prediction tasks with data obtained from professional drivers.

Similarly to LfD, learned driver models can be employed to improve the performance of ADAS. Lefèvre et al. [15] developed an ADAS with a driver model, the parameters of which was tuned based on the history of the driver's control actions. Di Cairano et al. [16] developed an algorithm that predicts human longitudinal control actions based on a Markov chain, and then the algorithm computes the optimal control input from ADAS using a model predictive controller to achieve a more energy-efficient operation for hybrid electric vehicles.

II. Driver Steering Torque Prediction

A. Feature Vector

This work follows the modeling framework of the TPVDCM [2] (see Fig. 1). The system has five subsystems:

K. Okamoto is a Ph.D. Candidate at the School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA 30332-0150, USA. Email: kazuhide@gatech.edu

P. Tsiotras is a Professor at the School of Aerospace Engineering and the Institute for Robotics & Intelligent Machines, Georgia Institute of Technology, Atlanta, GA 30332-0150, USA. Email: tsiotras@gatech.edu

the road geometry, the perception subsystem, the driver, the steering column, and the vehicle dynamics subsystem. The perception subsystem converts road curvature ρ, side-slip angle β, yaw rate r, to two angles called $\theta_{\rm near}$ and $\theta_{\rm far}$ that the driver subsystem processes, in order to compute the steering torque $T_{\rm dr}$. Because the measurement of the angles $\theta_{\rm near}$ and $\theta_{\rm far}$ is difficult, this work combines the perception subsystem with the driver subsystem, which we call the "human driver" subsystem, as shown inside the dashed box in Fig. 1. The inputs to the human driver subsystem are ρ, β,r and the steering-wheel angle δ_s. The output is $T_{\rm dr}$.

To incorporate the sequential changes of the inputs for the last $\ell\Delta t$ seconds, where $\ell \in \mathbb{N}^+$ and $\Delta t \in \mathbb{R}$ is the sampling interval, we define the following feature vector $z(t_k) \in \mathbb{R}^d$:

$$\begin{aligned} z(t_k) = [&\rho(t_k), \beta(t_k), r(t_k), \delta_s(t_k), \dots \\ &T_{\rm dr}(t_{k-1}), \rho(t_{k-1}), \beta(t_{k-1}), \dots, \delta_s(t_{k-\ell})]^\top, \end{aligned} \tag{1}$$

where $k \in \mathbb{N}^+$ is the index for the time steps. Note that we assume that the input variables at the current time step are available, eliminating the modeling error of the other parts of the system and enabling direct comparison of the estimation performance of $T_{\rm dr}$. The problem we wish to solve is to estimate the nonlinear function f such that

$$T_{\rm dr}(t_k) = f(z(t_k)). \tag{2}$$

We set the time discretization as $\Delta t = 1/\ell$ sec. Thus, the vector $z(t_k)$ includes the input variables over the previous one second and the steering torque output at the previous $\Delta t, 2\Delta t, \dots, \ell\Delta t(= 1)$ seconds.

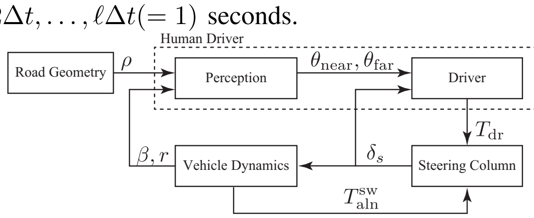


Fig. 1: The framework of the two-point visual control model.

B. Piecewise Linear Models

As a benchmark for evaluating the performance of ML methods, we use two simple models, which disregard much of the information in $z(t_k)$ and are expected to show a sound prediction performance if the steering command is smooth and Δt is sufficiently small. Otherwise, we cannot expect a good performance from this model, and the ML methods are expected to outperform these two naive methods.

1) Piecewise Constant Model (PCM): PCM assumes that the driver torque at the current time step is the same as the driver's torque at the previous time step, that is,

$$T_{\rm dr}^{\rm PCM}(t_k) = T_{\rm dr}(t_{k-1}). \tag{3}$$

2) Piecewise Linear Model (PLM): PLM employs the time derivative of $T_{\rm dr}$ and predicts the driver's torque at the current time step as follows:

$$T_{\rm dr}^{\rm PLM}(t_k) = T_{\rm dr}(t_{k-1}) + \left.\frac{{\rm d}T_{\rm dr}}{{\rm d}t}\right|_{t_{k-1}} \Delta t, \tag{4}$$

where we approximate the time derivative of $T_{\rm dr}$ at $t = t_{k-1}$ as $\left.\frac{{\rm d}T_{\rm dr}}{{\rm d}t}\right|_{t_{k-1}} \approx (T_{\rm dr}(t_{k-1}) - T_{\rm dr}(t_{k-2}))/\Delta t$.

C. Gaussian Process Regression (GP)

This subsection briefly explains the GP. For a more detailed discussion, we refer the reader to [17]. A GP is a non-parametric kernel-based method represented as a collection of random variables, any finite number of which have a joint Gaussian distribution.

The training of a GP assumes that the observation has an independent, identically distributed (i.i.d.) Gaussian noise ϵ

$$T_{\rm dr} = f(z) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma_n^2), \tag{5}$$

and assumes that N_D data points $\mathcal{D} = \{z(t_\kappa), T_{\rm dr}(t_\kappa)\}_{\kappa=1}^{N_D}$ have been observed. For simplicity, in this section we omit the argument t_κ, and we let z be $z(t_\kappa)$ and z' be $z(t_{\kappa'})$, where $\kappa, \kappa' \in \{1, \dots, N_D\}$. Note that κ is the index for the training data set.

A GP is specified by its mean and covariance: $f(z) \sim \mathcal{GP}(\mu(z), k(z, z'))$. Here, for simplicity, we describe the training and prediction of a zero-mean GP that is, $\mathcal{GP}(0, k(z, z'))$. For the kernel, we employ the exponential kernel covariance function, defined as

$$k(z, z'|\sigma_\ell, \sigma_f) = \sigma_f^2 \exp\left(-\frac{r(z, z')}{\sigma_\ell}\right), \tag{6}$$

where σ_ℓ is the characteristic length scale, σ_f is the signal standard deviation, and $r(z, z')$ is the Euclidean distance between z and z'. The prior covariance matrix of the observations with noise is $K_{T_{\rm dr}}(Z, Z) = K(Z, Z) + \sigma_n^2 I$, where $Z = [z(t_1), \dots, z(t_{N_D})]$, and $[K(Z, Z)]_{\kappa,\kappa'} = k(z, z')$. The hyper-parameters $\theta = (\sigma_\ell, \sigma_f, \sigma_n) \in \Theta$, are tuned during the process of training as $\theta^* = \underset{\theta\in\Theta}{\operatorname{argmax}} \Pr(Y|Z, \theta)$, where $\Pr(Y|Z, \theta) = \mathcal{N}(0, K_{T_{\rm dr}}(Z, Z))$. In order to compute θ^*, we take the logarithm of the conditional distribution, and this function is maximized with respect to θ.

When predicting $T_{\rm dr}$, because the joint distribution of the observed target values and function values at test locations under the above prior is

$$\begin{bmatrix} T_{\rm dr} \\ T_{\rm dr}^{\rm test} \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K_{T_{\rm dr}}(Z, Z) & K(Z, Z^{\rm test}) \\ K(Z^{\rm test}, Z) & K(Z^{\rm test}, Z^{\rm test}) \end{bmatrix}\right), \tag{7}$$

we obtain the following conditional probability distribution of the function values at the test locations

$$\Pr(T_{\rm dr}^{\rm test}|Z^{\rm test}, \mathcal{D}) \sim \mathcal{N}(\mu_{T_{\rm dr}^{\rm test}}, {\rm Cov}_{T_{\rm dr}^{\rm test}}), \tag{8}$$

where

$$\mu_{T_{\rm dr}^{\rm test}} = K(Z^{\rm test}, Z) K_{T_{\rm dr}}^{-1}(Z, Z) Y, \tag{9}$$

$$\begin{aligned} {\rm Cov}_{T_{\rm dr}^{\rm test}} &= K(Z^{\rm test}, Z^{\rm test}) - \\ &\quad K(Z^{\rm test}, Z) K_{T_{\rm dr}}(Z, Z)^{-1} K(Z, Z^{\rm test}), \end{aligned} \tag{10}$$

and $Y = [T_{\rm dr}(t_1), \dots, T_{\rm dr}(t_{N_D})]^\top$. Thus, given a new feature vector $z(t_k)$, the predicted driver torque by GP is

$$T_{\rm dr}^{\rm GP}(t_k) = K(z(t_k), Z) K_{T_{\rm dr}}(Z, Z)^{-1} Y. \tag{11}$$

D. Gaussian Mixture Regression (GMR)

GMR is a multivariate nonlinear function regression method based on GMMs. To simplify the notation, we again omit t_k in this subsection. We assume that the joint distribution $\Pr(z, T_{\rm dr})$ can be represented in the form of a GMM with N_G Gaussian functions:

$$\Pr(z, T_{\rm dr}) = \sum_{i=1}^{N_G} \pi_i \mathcal{N}(z, T_{\rm dr}; \mu_i, \Sigma_i), \tag{12}$$

where π_i is the initial probability for $(z, T_{\rm dr})$ to lie in the ith Gaussian. In addition, μ_i and Σ_i are the mean and the covariance matrix of the ith Gaussian, defined as

$$\mu_i = \begin{bmatrix} \mu_i^z \\ \mu_i^{T_{\mathrm{dr}}} \end{bmatrix}, \quad \Sigma_i = \begin{bmatrix} \Sigma_i^{zz} & \Sigma_i^{zT_{\mathrm{dr}}} \\ \Sigma_i^{T_{\mathrm{dr}}z} & \Sigma_i^{T_{\mathrm{dr}}T_{\mathrm{dr}}} \end{bmatrix}. \tag{13}$$

The number of Gaussians N_G be tuned based on the Akaike and Bayesian information criteria (AIC [18] and BIC [19]). We can then compute the conditional probability $\Pr(T_{\mathrm{dr}}|z,i)$ as

$$\Pr(T_{\mathrm{dr}}|z,i) = \mu_i^{T_{\mathrm{dr}}} + \Sigma_i^{T_{\mathrm{dr}}z}(\Sigma_i^{zz})^{-1}(z-\mu_i^{zz}). \tag{14}$$

Thus, the estimate of the driver's torque $T_{\mathrm{dr}} = \sum_{i=1}^{N_G} h_i(z)\Pr(T_{\mathrm{dr}}|z,i)$ through a GMR is

$$T_{\mathrm{dr}}^{\mathrm{GMR}} = \sum_{i=1}^{N_G} h_i(z)\left[\mu_i^{T_{\mathrm{dr}}} + \Sigma_i^{T_{\mathrm{dr}}z}(\Sigma_i^{zz})^{-1}(z-\mu_i^{zz})\right], \tag{15}$$

where $h_i(z) \in [0,1]$ is the probability of an observed input to belong to the ith Gaussian $h_i(z) = \mathcal{N}(z;\mu_i^z,\Sigma_i^{zz})$ and normalized such that $\sum_{i=1}^{N_G} h_i(z) = 1$.

E. Hidden Markov Model Gaussian Mixture Regression (HMM-GMR)

The HMM-GMR [20] combines a GMR with an HMM. The combination of these two methods enables the algorithm to consider both the spatial and sequential information of $z(t_k)$ by incorporating transitions between Gaussians from the recursive computation of the weights in Eq. (15) as

$$h_i(z(t_k)) = \frac{\left(\sum_{j=1}^{N_G} h_j(z(t_{k-1}))a_{ji}\right)\hat{h}_i(z(t_k))}{\sum_{l=1}^{N_G}\left[\left(\sum_{j=1}^{N_G} h_j(z(t_{k-1}))a_{jl}\right)\hat{h}_l(z(t_k))\right]}, \tag{16}$$

where $\hat{h}_i(z(t_k)) = \mathcal{N}(z(t_k);\mu_i^z,\Sigma_i^z)$, and a_{ji} is the transition probability from the jth Gaussian to the ith Gaussian. To train the transition probability a_{ji}, we employ an expectation maximization algorithm. The resulting prediction of HMM-GMR is

$$T_{\mathrm{dr}}^{\text{HMM-GMR}}(t_k) = \sum_{i=1}^{N_G} h_i(z)\left[\mu_i^{T_{\mathrm{dr}}} + \Sigma_i^{T_{\mathrm{dr}}z}(\Sigma_i^{zz})^{-1}(z-\mu_i^{zz})\right], \tag{17}$$

where z, as before, denotes $z(t_k)$.

F. Artificial Neural Network (ANN)

ANNs are nonlinear function regression methods. While currently many deep networks are actively proposed, this work employs a simple ANN that has one hidden layer with N_h nodes to prevent overfitting. At each hidden unit $i \in \{1,\ldots,N_h\}$, the input vector $z(t_k)$ is multiplied by a weight vector $W_h^i \in \mathbb{R}^d$ such that $h^i = W_h^{i\top}z(t_k) + b_h^i$, where $b_h^i \in \mathbb{R}$ is a bias-term. The signal h^i is then fed into a nonlinear transfer function $\phi:\ \mathbb{R} \to [-1,1]$. In this work, we employ a hyperbolic tangent function: $\phi(h^i) = 2/(1+\exp(-2h^i)) - 1$. The output of the hidden nodes $\boldsymbol{\phi}(h) = [\phi(h^1),\ldots,\phi(h^{N_h})]^\top \in \mathbb{R}^{N_h}$, is fed into the output layer, and its output

$$T_{\mathrm{dr}}^{\mathrm{ANN}}(t_k) = W_o^\top \boldsymbol{\phi}(h) + b_o, \tag{18}$$

where $W_o \in \mathbb{R}^{N_h}$ and $b_o \in \mathbb{R}$, is the predicted T_{dr} by this ANN. The training of the ANN deals with finding the best parameters W_h, W_o, b_h, and b_o, such that the output in Eq. (18) best fits the training and validation data set.

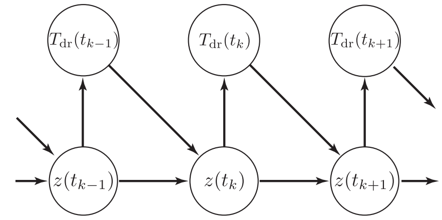


Fig. 2: The graphical model of driver behavior prediction.

III. Experiments

In this section we describe the experiments conducted to compare the performance of these algorithms. In the first experiment (Exp I), we employ a driver control model to generate a synthetic human-driver control data set, apply the algorithms in Section II, and evaluate their performance. The purpose of Exp I was to evaluate the performance without having the effect of inconsistency inherent in human driving skills, which always exists with real data, especially from non-professional human drivers. Unlike mathematical human driver control models, human drivers do not always respond in the same manner to the same stimuli. By employing a human driver control model, we can eliminate this uncertainty of human driving skills and isolate and evaluate the regression performance of each algorithm. In the second experiment (Exp II), we employ a real human-driving data set that is collected using a driving simulator. By conducting this experiment, we compare the performance when driving-behavior variation exists in the data set so that we can evaluate the robustness to driving-behavior variations.

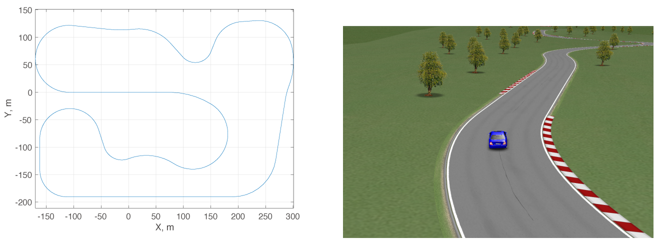
A. Performance Measure

As the performance measure, we employ the root mean squared error (RMSE)

$$\mathrm{RMSE}(T_{\mathrm{dr}}^{\cdots}) = \sqrt{\frac{1}{N_{DT}}\sum_{k=1}^{N_{DT}}(T_{\mathrm{dr}}(t_k) - T_{\mathrm{dr}}^{\cdots}(t_k))^2}, \tag{19}$$

where T_{dr} is the true value of the driver torque, $T_{\mathrm{dr}}^{\cdots}$ is the predicted value by each algorithm in Section II, and N_{DT} is the number of test data points.

We compare the performance only over short-term prediction horizons. Long-term predictions employ modeled dynamics and perform an iterative computation of the future state and the driver's steering torque as depicted in the graphical model in Fig. 2. The function f, which the algorithms try to identify, corresponds to the perpendicular arrows from z to T_{dr}. The horizontal arrows between z correspond to the vehicle dynamics, steering column, and road geometry models. Also, the slant arrows from T_{dr} to z shows that T_{dr} at the previous time step is incorporated into an entry of z at the next time step (see Eq. (1)). To predict the future driving torque at t_{k+1}, we need to first estimate $T_{\mathrm{dr}}(t_k)$ from $z(t_k)$ using the methods outlined in Section II. We then propagate the information in $z(t_k)$ based on a vehicle dynamics model and road geometry, and obtain the estimated value of $z(t_{k+1})$. Having obtained the estimated value of $z(t_{k+1})$, we can estimate $T_{\mathrm{dr}}(t_{k+1})$.



(a) (b)

Fig. 3: (a) The road geometry. (b) A screenshot from CarSim.

B. Synthetic Data

In Exp I, we employ data generated using CarSim® with a hybrid sensorimotor TPVDCM [4]. The model we employ accounts for the anticipatory control of human drivers with a model predictive controller. We try to reproduce the actions of this “human control” using the methods in Section II.

1) Data Set: The road circuit we use is one of the pre-installed CarSim® road circuits (see Fig. 3 (a)). The length of one lap is around 2 km. Each data set consists of one clockwise lap and one counter-clockwise lap. The vehicle speed is fixed at 50 km/h, which is high for the radii of the corners of the road circuit, so that we can investigate both the comfort and non-comfort zones of the driver.

2) Data Preprocessing: Before applying the regression algorithms in Section II, we normalize all inputs to achieve mean zero and standard deviation one. Then, to reduce noise, we apply a first-order lowpass filter with a cutting frequency at $\omega_s = 2.5$ rad/s and employ a local weighted linear least squares to a second degree polynomial. In addition, we resample the data set with $\Delta t = 0.2$ sec, setting $\ell = 5$ and $d = 29$ in Eq. (1).

To evaluate the performance of the methods, we divided the data set into four sub-data sets A,B,C, and D, and performed a four-fold cross validation. For instance, in sub-data set A, we employ the first 3/4 of data as the training data set and the last quarter as the test data set.

3) Results: Figure 4 shows the RMSE of each method. In Exp I, we used the GMR and HMM-GMR with $N_G = 21$ and the ANN with $N_h = 2$. The PWC exhibits the largest RMSE in all the data sets, while PWL demonstrates slightly smaller values. The RMSE of the GP is almost 1/3 of that of the PWC and PWL. The performance of GMR is competitive with that of the GP. Since it takes into account both the spatial and sequential information of z, the HMM-GMR, not surprisingly, outperforms the GMR. Finally, the performance of the ANN is comparable with that of the HMM-GMR.

C. Human Driving Data

In Exp II, in order to verify the performance of each method against real human-driving data, we employed actual human-driver data set collected using a driving simulator and compared the performance of all the methods. The human driver subject has eight-year driving experience. The data was pre-processed in the same way as for the simulated human driving data set described in Section III-B.2.

1) Georgia Tech Driving Simulator (GTDS): We collected data using the simulator shown in Fig. 5. With the GTDS we

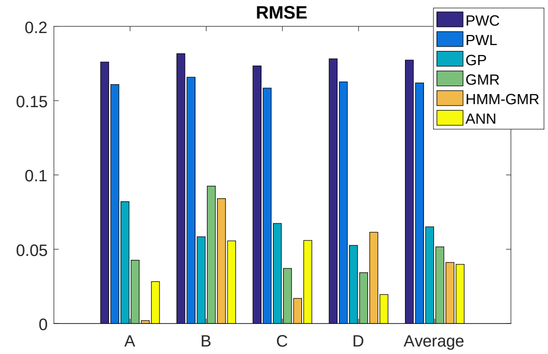
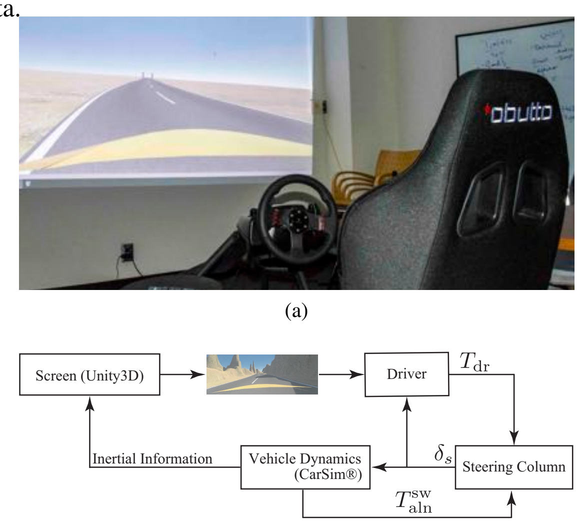


Fig. 4: The RMSE of each method for Exp I.

duplicated exactly the same road circuit employed for Exp I and eliminated the difference between simulation and experiment road geometry, thus enabling direct comparison of prediction performance against synthetic and actual driving data.



(a)

(b)

Fig. 5: (a) GTDS. (b) Structure of the simulator.

Figure 5(b) illustrates the interaction between a human driver and the components of the simulator. Each component is connected via the Robot Operating System (ROS) [21], while CarSim® computes the vehicle dynamics. The high-fidelity vehicle model of CarSim® enables the reproduction of realistic vehicle behaviors in a simulation environment. In Exp II, similarly to Exp I, the vehicle speed is fixed at 50 km/h. Thus, the control output from the human driver is only T_{dr}, which is then fed into the steering wheel. The output of the steering wheel is δ_s. The simulator employs a high-end gaming steering wheel with force-feedback functionality, and the driver can feel the alignment torque from the steering wheel. This force-feedback makes the simulation environment more realistic. To visualize the information computed by CarSim®, we use Unity3D, a game-development platform. The simulated view from the driver’s seat was projected on a 8 ft x 6 ft screen. Based on the view of the screen, the human driver steers the vehicle (see Fig. 5(a)). Figure 6 depicts the obtained driving torque data.

2) Results: Figure 7 depicts the RMSE for each method. We employed the GMR and HMM-GMR with $N_G = 23$

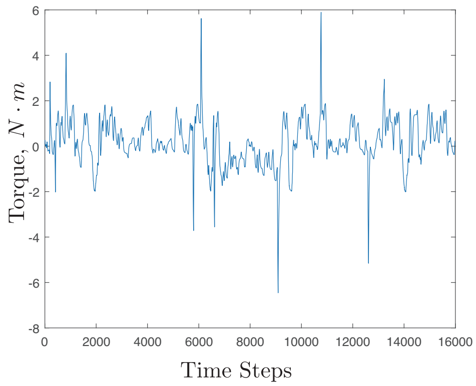


Fig. 6: Driving torque.

in this experiment and the ANN with $N_h = 2$. The RMSE of PWC and PWL show almost the same performance as in Exp I. In contrast, GP, which exhibited superior performance than PWC and PWL in Exp I, shows almost equivalent performance to PWC and PWL in this experiment. In addition, the performance of GMR is worse than that of PWC and PWL. The HMM-GMR error is even larger than the GMR error with sub-data set B. These results are contrary to our expectation, because these methods, i.e, GP, GMR, and HMM-GMR, incorporate the information of state variables (i.e.,β, r, δ_s, and ρ), they were expected to outperform both PWC and PWL. By contrast, and similarly to the case in Exp I, the ANN exhibits a superior performance than the other methods. In order to investigate this counter-intuitive result, we conducted another experiment (Exp III).

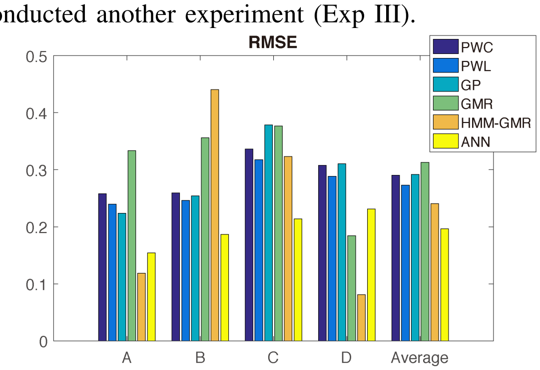


Fig. 7: The RMSE of each method for Exp II.

3) Performance Evaluation using a Data Generated with a TPVDCM: In order to investigate the inconsistency between the results in synthetic data (Exp I) and real-driving data (Exp II) of GMR and HMM-GMR, we conducted Exp III, in which we evaluated the performance based on the output of a sensorimotor TPVDCM, the parameters of which were identified based on the real human driving data employed in Exp II. We smoothed the data to eliminate nonlinearities and applied a dual EKF [22] to identify the parameters of the sensorimotor TPVDCM, thus enabling the evaluation of the prediction performance against the actual driving data without driving skill inconsistency. Figure 8 depicts the steering torque data used in Exp III.

Figure 9 depicts the regression performance of all algorithms. We observe a significant performance improvement of GMR and HMM-GMR. In addition, both GMR and

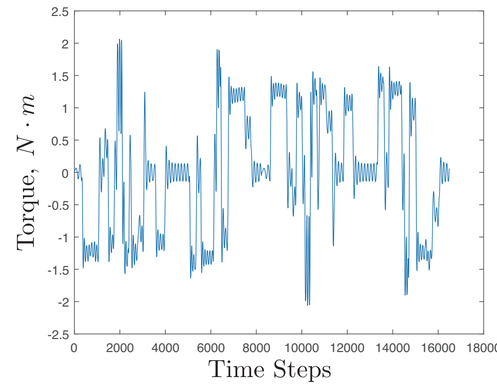


Fig. 8: The output of trained driver control model.

HMM-GMR exhibited superior performance than the ANN.

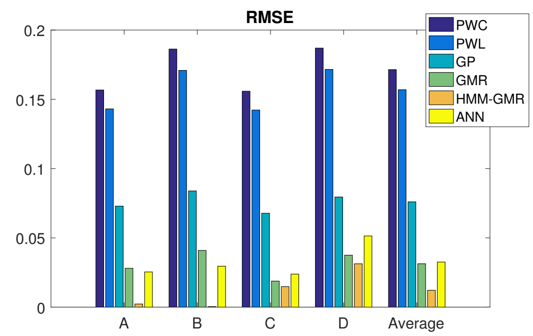


Fig. 9: The RMSE of each method for Exp III.

IV. Discussion

This section discusses the results of Section III. The PWC and PWL are the simplest models and do not use the input variables, i.e, β, r, δ_s, and ρ, but still exhibited sound performance in Exp I and II. We would like to emphasize, however, that, since they ignore the information in z other than T_{dr}, they are expected to show inferior performance to other methods in long-term prediction tasks.

Among the tested ML algorithms, GP showed the largest RMSE. In addition, the large computational cost due to the inverse operation in Eq. (11), the need to choose the kernel function to employ, and the number of hyper-parameters to tune, make GP a less than ideal choice for predicting the driver's torque. By contrast, the HMM-GMR exhibited almost an equivalent performance to the ANN in Exp I. This result is similar to the case of longitudinal control of human drivers [13]. Also, the HMM-GMR and ANN are competitive in terms of the implementation-complexity. In order to train GMR and HMM-GMR, we need to specify the number of Gaussians, while, in order to train an ANN, we need to choose the number of hidden layers and nodes. Thus, from the performance and implementation complexity perspective, with the data set from Exp I, we regard the HMM-GMR and ANN as competitive.

In Exp II, however, while HMM-GMR exhibited worse performance than PWC and PWL, especially in the sub-data set B, the ANN always showed superior performance than those. We hypothesize that this difference is due to the robustness of each method to the control variations of human drivers. Since human-driver control models are mathematical abstractions, their response is always the same if the input is the same. By contrast, human driver behavior does not necessarily follow this abstraction, especially when the situation is outside the comfort zone of the driver. This hypothesis is supported by experiment, Exp III, in which a sensorimotor TPVDCM was identified using the given data and was used to regenerate "real" human driving data, which has no control variations. In Exp III, we observed that GMR and HMM-GMR exhibited a significant performance improvement, which implies that GMR and HMM-GMR can exhibit a good performance if the human subject is a skilled driver who exhibits little control variations. The result also implies that, since ANN showed consistent performance from Exp I to III, in addition to skilled human-driver behavior [14], ANN can accurately predict *novice* human-driver behavior.

We conjecture that this stable performance of the ANN is due to its simple structure, which may prevent overfitting to the training data set. In terms of performance stability, PWC, PWL, GP, and ANN exhibited constant performance. ANN exhibited less RMSE than the other methods because PWC and PWL neglect many components of the feature vector, and the i.i.d. Gaussian assumption of GP is strict for our data set. In terms of RMSE values, GMR, HMM-GMR, were outperformed by ANN in Exp II. One possible reason for this is that the EM algorithm used to train GMR started from a bad initial point and got trapped at a bad local optimum.

From the experiments in Section III, we may conclude that an ANN with one hidden layer and two nodes is a good choice to model the lateral control action of human drivers. It should be noted, however, that we do not claim that human driver control actions can be modeled with a simple ANN. Driving consists of multiple difficult tasks such as scene understanding and decision making. In this work, we focused our attention only on the specific task of path following. What we claim is that, for path following tasks, a simple ANN exhibits an acceptable performance.

V. Summary

This paper addressed the problem of predicting lateral control actions of human drivers using ML methods. While we briefly proposed a method to perform long-term predictions, we compared the short-term prediction performance using two experiments. The first experiment employed a synthetic data set, while the second experiment used a real human-driving data set collected using a high fidelity driving simulator. From these experiments, we observed that the ANN exhibits the smallest RMSE and the highest robustness to the driving control variations. Future work will investigate personalized ADAS for vehicle lateral control by taking advantage of the predictive power of these driver control models.

VI. Acknowledgement

This work was supported by NSF award CPS-1544814 and by Ford Motor Company. K.Okamoto was also supported by Funai Foundation for Information Technology and Ito Foundation USA-FUTI Scholarship.

References

[1] R. N. Jazar, *Vehicle Dynamics: Theory and Application*. Springer Science & Business Media, 2013.

[2] D. D. Salvucci and R. Gray, "A two-point visual control model of steering," *Perception-London*, vol. 33, no. 10, pp. 1233–1248, 2004.

[3] C. Sentouh, P. Chevrel, F. Mars, and F. Claveau, "A sensorimotor driver model for steering control," in *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics*, San Antonio, TX, Oct. 11 – 14, 2009, pp. 2462–2467.

[4] K. Okamoto and P. Tsiotras, "A new hybrid sensorimotor driver model with model predictive control," in *IEEE Conference on Systems, Man, and Cybernetics*, Budapest, Hungary, Oct. 9 – 12, 2016, pp. 1866–1871.

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

[6] G. S. Aoude, V. R. Desaraju, L. H. Stephens, and J. P. How, "Driver behavior classification at intersections and validation on large naturalistic data set," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 2, pp. 724–736, 2012.

[7] N. P. Chandrasiri, K. Nawa, and A. Ishii, "Driving skill classification in curve driving scenes using machine learning," *Journal of Modern Transportation*, vol. 24, no. 3, pp. 196–206, 2016.

[8] K. Okamoto, K. Berntorp, and S. Di Cairano, "Similarity-based vehicle-motion prediction," in *American Control Conference*, Seattle, WA, May 24–26, 2017, pp. 303–308.

[9] ——, "Driver intention-based vehicle threat assessment using random forests and particle filtering," in *The 20th World Congress of the International Federation of Automatic Control*, Toulouse, France, July 9 – 14, 2017.

[10] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469–483, 2009.

[11] A. Armand, D. Filliat, and J. Ibanez-Guzman, "Modelling stop intersection approaches using Gaussian processes," in *International IEEE Conference on Intelligent Transportation Systems*, The Hague, Netherlands, Oct. 6 – 9, 2013, pp. 1650–1655.

[12] C. Miyajima, Y. Nishiwaki, K. Ozawa, T. Wakita, K. Itou, K. Takeda, and F. Itakura, "Driver modeling based on driving behavior and its evaluation in driver identification," *Proceedings of the IEEE*, vol. 95, no. 2, pp. 427–437, 2007.

[13] S. Lefèvre, C. Sun, R. Bajcsy, and C. Laugier, "Comparison of parametric and non-parametric approaches for vehicle speed prediction," in *American Control Conference*, Portland, OR, June 4–6, 2014, pp. 3494–3499.

[14] 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 *52nd IEEE Conference on Decision and Control*, Firenze, Italy, Dec. 10 – 13, 2013, pp. 6778–6785.

[15] S. Lefèvre, A. Carvalho, and F. Borrelli, "Autonomous car following: A learning-based approach," in *Intelligent Vehicles Symposium*, Seoul, South Korea, June 28 – July 1, 2015, pp. 920–926.

[16] S. Di Cairano, D. Bernardini, A. Bemporad, and I. V. Kolmanovsky, "Stochastic MPC with learning for driver-predictive vehicle control and its application to HEV energy management," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 3, pp. 1018–1031, 2014.

[17] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[18] H. Akaike, "Information theory and an extension of the maximum likelihood principle," in *Selected Papers of Hirotugu Akaike*. Springer, 1998, pp. 199–213.

[19] G. Schwarz *et al.*, "Estimating the dimension of a model," *The Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1978.

[20] S. Calinon, F. D'halluin, E. Sauser, D. Caldwell, and A. Billard, "A probabilistic approach based on dynamical systems to learn and reproduce gestures by imitation," *IEEE Robotics and Automation Magazine*, vol. 17, no. 2, pp. 44–54, 2010.

[21] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2, Kobe, Japan, May 12 – 17, 2009.

[22] C. You, J. Lu, and P. Tsiotras, "Nonlinear driver parameter estimation and driver steering behavior analysis for adas using field test data," *IEEE Transactions on Human-Machine Systems*, vol. 47, no. 5, pp. 686–699, 2017.