Autonomous Car Following: A Learning-Based Approach

Conference Paper · June 2015

DOI: 10.1109/IVS.2015.7225802

Stephanie Lefevre
Mercedes Benz Research and Development Nor…
31 PUBLICATIONS **242** CITATIONS
SEE PROFILE

Ashwin Carvalho
University of California, Berkeley
17 PUBLICATIONS **78** CITATIONS
SEE PROFILE

Autonomous Car Following: A Learning-Based Approach

Stéphanie Lefèvre, Ashwin Carvalho, Francesco Borrelli

Abstract— **We propose a learning-based method for the longitudinal control of an autonomous vehicle on the highway. We use a driver model to generate acceleration inputs which are used as a reference by a model predictive controller. The driver model is trained using real driving data, so that it can reproduce the driver's behavior. We show the system's ability to reproduce different driving styles from different drivers. By solving a constrained optimization problem, the model predictive controller ensures that the control inputs applied to the vehicle satisfy some safety criteria. This is demonstrated on a vehicle by artificially creating potentially dangerous situations with virtual obstacles.**

I. Introduction

There have been many contributions to the field of autonomous driving in the last decade, starting with robot-like cars which could navigate autonomously in artificial urban environments [1], then progressively integrating automation technologies into commercial cars to assist the driver in predefined scenarios (e.g. Adaptive Cruise Control, Autonomous Emergency Braking). However the contributions have focused on individual software components (e.g. perception, control), while the system architecture has not evolved. Current autonomous vehicles use the same architecture as the DARPA Urban Challenge vehicles [1], [2], [3], [4]. This architecture is shown in Fig. 1 and the different processing modules are described below:

- The "Perception + Localization" module combines data received from sensors and digital maps to estimate some relevant states representing the driving situation (e.g. ego vehicle, other vehicles, road geometry).
- The "Motion planner" selects the appropriate high-level behavior (e.g. car following, lane changing) and generates a trajectory to execute that behavior.
- The "Trajectory controller" computes the steering and acceleration commands with the objective to follow the reference trajectory as closely as possible. These commands are sent to the actuators.

This architecture has been successfully used in the field of terrestrial robotics for decades. However an autonomous car has an additional constraint compared to robots; it needs to ensure the comfort of its passengers. Therefore in the context of autonomous driving the motion planner in Fig. 1 has many responsibilities: it must generate trajectories which are safe, comfortable, and feasible. As a result motion planners tend to be very complex and computationally expensive [5]. Their design is made even harder by the interdependence of all the objectives: modifications to improve comfort might impact the feasibility or the safety, and vice-versa. One of the practical consequences is the need for a lot of tuning to obtain a good balance of safety, comfort, and feasibility.

In this paper we replace the trajectory generator by a driver model. The role of the driver model is to generate commands resembling those of a human driver, while the controller is in charge of safety. This redistribution of responsibilities leads to a flexible framework where the elements responsible for comfort and safety can be designed independently, and the challenge of generating a feasible trajectory is avoided. We implement and test this idea for a car-following application on a real vehicle. We introduce metrics to evaluate how closely the autonomous driving matches human driving, and show that the vehicle operates in a comfortable, safe manner.

The remaining of the paper is organized as follows. Section II reviews related work on architectures for autonomous driving. The proposed approach is described in Section III, and experimental results are presented in Section IV. Section V concludes the paper.

The authors are with the Department of Mechanical Engineering, University of California at Berkeley, USA, {slefevre, ashwinc, fborrelli}@berkeley.edu.

This material is based upon work partially supported by the National Science Foundation under Grant No. 1239323. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. The work was also supported by the Hyundai Center of Excellence at UC Berkeley.

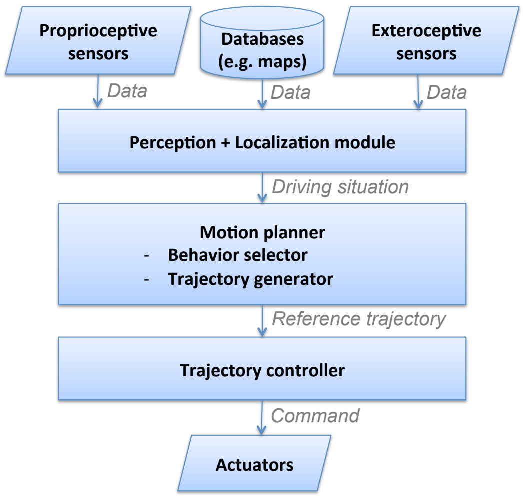

Fig. 1. Standard architecture of an automated ground vehicle (adapted from [3]).

II. Related work

Most autonomous vehicles follow the architecture shown in Fig. 1, but some alternative architectures have been proposed to reduce the complexity of the trajectory generation step or avoid it altogether.

One option is to use a data-driven framework called learning-by-demonstration where a driver model is learned from data and then directly used as a controller. This strategy has been implemented in the past using a Piece-Wise AutoRegressive eXogenous model [6], Artificial Neural Networks [7], and Inverse Optimal Control [8]. The results demonstrated that it is possible to reproduce human driving behavior in an autonomous car, by learning from example data. However these approaches did not address the problem of safety.

Another approach, proposed recently by Wei et al. [5], consists in breaking down the “Motion planner” module of Fig. 1 into 3 submodules handling simplified problems. The first module generates a smooth reference trajectory taking into account the road geometry but assuming no traffic. The second module acknowledges the presence of other vehicles using a Prediction- and Cost-function Based algorithm. The idea is to generate sample trajectories from intelligently sampled sequences of states (such as the distance to the preceding vehicle) and to use a cost function to evaluate each potential trajectory in terms of comfort, safety, and efficiency. The best sample is sent as a reference to the final module, which consists of several controllers running in parallel and dedicated to specific tasks such as Adaptive Cruise Control and Lane Keeping.

III. Proposed approach

We propose a solution which combines two of the methods presented in the previous section: learning-by-demonstration, which is able to generate commands which feel natural to the passengers, and predictive control, which relies on model-based predictions to make decisions while enforcing state and input constraints. Our approach inherits the advantages of both. The proposed solution is illustrated in Fig. 2, and the differences with Fig. 1 are explained below:

- The *trajectory generator* is replaced by a *driver model*. The driver model uses learning-by-demonstration: it is trained using real driving data and can generate control inputs which imitate the driving style of the driver it learned from.
- The *trajectory controller* is replaced by a *Model Predictive Control (MPC)* based controller which takes as an input the reference control inputs generated by the driver model. The role of the controller is to satisfy safety constraints such as collision avoidance. The control input computed by the controller matches the reference input provided by the driver model unless a safety constraint violation is anticipated, in which case the controller modifies the command to keep the vehicle safe.

Since the comfort and safety objectives are handled by different modules, the architecture in Fig. 2 is more flexible than the one in Fig. 1. The driver model and the model predictive controller can be designed independently, and

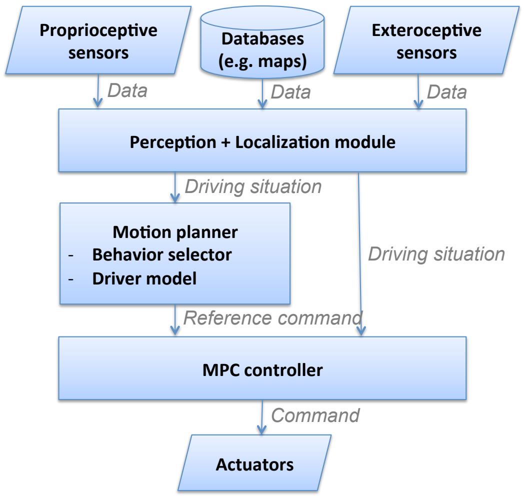

Fig. 2. Proposed architecture combining Learning-by-Demonstration and Model Predictive Control

one can be modified at any time without impacting the performance of the other. This feature is particularly useful if one wants to adjust the car’s driving style over time: the driver model can learn continuously, or be replaced, without having to readjust any other module.

The remainder of this paper will focus on longitudinal control during lane keeping for autonomous highway driving. The steering controller was purposefully disabled to allow us to evaluate the system’s ability to perform car following independently of the curvature of the road it was tested on. For more information about the driver model and controller which we generally use for lateral control, we refer the reader to our previous publications [9], [10].

A. Problem formulation

The longitudinal control problem during lane keeping is formulated below. We define:

- $\xi_t = [d_t, v_t]$ the state of the ego vehicle at time t, with $d_t \in \mathbb{R}^+$ the longitudinal position of the ego vehicle in a road-aligned coordinate system, and $v_t \in \mathbb{R}^+$ the longitudinal velocity of the ego vehicle,
- $\xi_t^p = [d_t^p, v_t^p]$ the state of the preceding vehicle at time t, with $d_t^p \in \mathbb{R}^+$ the longitudinal position of the preceding vehicle in a road-aligned coordinate system, and $v_t^p \in \mathbb{R}^+$ the longitudinal velocity of the preceding vehicle,
- $z_t = [d_t^r, v_t^r, v_t]$ the features representing the current driving situation at time t, to be used by the driver model. $d_t^r = (d_t^p - d_t)$ is the relative distance to the preceding vehicle, $v_t^r = (v_t^p - v_t)$ is the relative velocity to the preceding vehicle.

At each time step t the driver model must generate an acceleration sequence $\mathbf{a_t^{\text{ref}}} = [a_t^{\text{ref}}, ..., a_{t+N_d-1}^{\text{ref}}]$, where $N_d = T/\Delta t_d$ is the number of time steps in the prediction horizon T for the driver model’s sampling time Δt_d. This acceleration sequence is used by the MPC-based controller as a reference. The controller solves a constrained optimization problem over the prediction horizon T, and generates a safe acceleration sequence. This acceleration sequence is denoted $\mathbf{a_t} = [a_t, ..., a_{t+N_c-1}]$, where $N_c = T/\Delta t_c$ is the number of time steps in the prediction horizon T for the controller's sampling time Δt_c. At time step t, only the first acceleration in the planned sequence is applied.

The next two sections provide details on the driver model and the controller.

B. Driver modeling

The role of the driver model is to generate an acceleration sequence $\mathbf{a_t^{\mathrm{ref}}}$ which will serve as a reference for the controller, using the history of driving situations $z_{1:t} = [z_1, ..., z_t]$. Our goal is to generate sequences which are as close as possible to what the driver would have done in the same situation. Many longitudinal driver models have been proposed in the past, notably by the microscopic traffic simulation community [11]. In this work we use a combination of Hidden Markov Model and Gaussian Mixture Regression (HMM+GMR) [12], a non-parametric regression method which was shown to outperform standard parametric approaches for acceleration prediction in the context of driving [11]. In what follows we describe the training phase, during which the driver model is learned, and then the method used to compute the future accelerations.

1) Training phase: During a training phase, real data is used to build a fully connected HMM representation of human control strategies during car following. Since our goal is to model the dependencies between the driving situation and the driver's acceleration, we define the following random variables for the HMM:

- $m_t \in \{1, ..., M\}$ is the hidden mode at time t, with M the number of possible hidden modes.
- $o_t = [z_t, a_t] \in \mathbb{R}^4$ is the vector of observations at time t, composed of the driving situation and the acceleration applied by the driver.

In the HMM, the joint distribution between the hidden modes and the observations is written as follows:

$$\begin{aligned} P(m_{0:t}, z_{1:t}, a_{1:t}) = \quad & P(m_0) \cdot \prod_{k=1}^{t} [P(m_k|m_{k-1}) \\ & \cdot P(z_k, a_k|m_k)] \end{aligned}$$

A multivariate Gaussian distribution is assumed for $P(z_k, a_k|m_k)$. The parameters of the HMM are: the number of hidden modes M, the initial distribution $P(m_0)$, the means and covariance matrices of the multivariate Gaussian distributions, and the transition probabilities α_{ij} between the i^{th} and j^{th} hidden modes. These parameters are learned from the training data using the Expectation-Maximization algorithm and the Bayesian Information Criterion as in [12].

2) Computation of the current acceleration: The reference acceleration at time t is computed from the consecutive values of the driving situation using Gaussian Mixture Regression, i.e. a_t^{ref} is computed as the conditional expectation of a_t given the sequence $z_{1:t}$:

$$\begin{aligned} a_t^{\mathrm{ref}} &= E[a_t|z_1, ..., z_t] \quad (1) \\ &= \sum_{i=1}^{M} \beta_{i,t} \cdot \left[\mu_i^a + \Sigma_i^{az}(\Sigma_i^z)^{-1}(z_t - \mu_i^z)\right] \quad (2) \end{aligned}$$

where

$$\mu_i = \begin{bmatrix} \mu_i^z \\ \mu_i^a \end{bmatrix} \text{ and } \Sigma_i = \begin{bmatrix} \Sigma_i^z & \Sigma_i^{za} \\ \Sigma_i^{az} & \Sigma_i^a \end{bmatrix}$$

and $\beta_{i,t}$ is the mixing coefficient for mode i at time t, computed as the probability of being in mode $m_t = i$ and observing the sequence $z_{1:t}$:

$$\beta_{i,t} = \frac{(\sum_{j=1}^{M} \beta_{j,t-1} \cdot \alpha_{ji}) \cdot \mathcal{N}(z_t|\mu_i^z, \Sigma_i^z)}{\sum_{l=1}^{M} \left[(\sum_{j=1}^{M} \beta_{j,t-1} \cdot \alpha_{jl}) \cdot \mathcal{N}(z_t|\mu_l^z, \Sigma_l^z)\right]} \quad (3)$$

3) Computation of the future accelerations: The driver model must provide the controller with a reference acceleration sequence $\mathbf{a_t^{\mathrm{ref}}} = [a_t^{\mathrm{ref}}, ..., a_{t+N_d-1}^{\mathrm{ref}}]$ computed using the history of driving situations $z_{1:t}$. This sequence is computed by iteratively applying the driver model defined in (1)–(2) and a linear time-invariant model to propagate the driving situation. The propagation model uses a kinematic point mass model for the ego vehicle and assumes constant velocity for the preceding vehicle:

$$\begin{aligned} d_{t+1}^r = & \quad d_t^r + v_t^r \cdot \Delta t_d - \frac{1}{2} a_t \cdot \Delta t_d^2 \quad (4) \\ v_{t+1}^r = & \quad v_t^r - a_t \cdot \Delta t_d \quad (5) \\ v_{t+1} = & \quad v_t + a_t \cdot \Delta t_d \quad (6) \end{aligned}$$

where Δt_d is the discretization time of the driver model. Compactly, the propagation model above is written in state-space form as:

$$z_{t+1} = Az_t + Ba_t. \quad (7)$$

To summarize, the sequence $\mathbf{a_t^{\mathrm{ref}}}$ is obtained by looping on the following two equations until the prediction horizon T is reached:

$$\begin{cases} z_{t+1} &= Az_t + Ba_t^{\mathrm{ref}} \\ a_{t+1}^{\mathrm{ref}} &= E[a_{t+1}|z_1, ..., z_{t+1}] \end{cases} \quad (8)$$

C. Controller

We formulate the goal of tracking the desired acceleration sequence $\mathbf{a_t^{\mathrm{ref}}}$ provided by the driver model while enforcing a set of safety constraints as a MPC problem. MPC has been shown to be an effective strategy for the real-time control of autonomous vehicles due to its ability to systematically handle constraints, nonlinearities and uncertainty (see e.g. [9], [10], and references therein). In MPC, at each sampling time, a sequence of control inputs is computed by solving a constrained finite-time optimal control problem. Only the first element of this sequence is applied to the system. The process is repeated at the next sampling time using the new measurements. The vehicle model and safety constraints used to formulate the online optimization problem are described below.

1) Vehicle model: A kinematic point-mass model is used for the control design. The state update equations are,

$$d_{t+1} = \quad d_t + v_t \cdot \Delta t_c + \frac{1}{2} a_t \cdot \Delta t_c^2 \tag{9}$$

$$v_{t+1} = \quad v_t + a_t \cdot \Delta t_d \tag{10}$$

where the controller's discretization time Δt_c is possibly different from Δt_d. Compactly, the linear time-invariant vehicle model is written in state-space form as,

$$\xi_{t+1} = C\xi_t + Da_t. \tag{11}$$

2) Safety constraints: The main role of the MPC-based control strategy is to prevent the ego vehicle from colliding with the preceding vehicle. This safety requirement is enforced by constraining the ego vehicle's longitudinal coordinate as,

$$d_t + d_{\text{safe}} \leq d_t^p, \tag{12}$$

where the longitudinal coordinate d_t^p of the preceding vehicle is computed using a constant velocity model, and the safety distance d_{safe} is a parameter.

In addition, physical limitations on the actuators impose bounds on the control input and slew rate,

$$a_{\min} \leq a_t \leq a_{\max} \tag{13}$$

$$\dot{a}_{\min} \leq \dot{a}_t \approx \frac{a_t - a_{t-1}}{\Delta t_c} \leq \dot{a}_{\max}. \tag{14}$$

3) MPC formulation: The optimal control input sequence is computed as the solution of the following optimization problem,

$$\min_{\mathbf{a}_t, \epsilon} \sum_{k=0}^{N_c-1} \left(\|a_{k|t} - a_{k|t}^{\text{ref}}\|_R^2 + \|a_{k|t} - a_{k-1|t}\|_P^2 \right) + S\epsilon \tag{15a}$$

$$\text{s.t. } \xi_{k+1|t} = C\xi_{k|t} + Da_{k|t} \tag{15b}$$

$$d_{k|t} + d_{\text{safe}} \leq d_{k|t}^p + \epsilon \tag{15c}$$

$$\mathbf{a}_t \in \mathcal{U} \tag{15d}$$

$$\xi_{0|t} = \xi_t \tag{15e}$$

$$a_{-1|t} = a_{t-1} \tag{15f}$$

where the notation $x_{k|t}$ denotes the variable x at time $(t+k)$ predicted at time t, $\|x\|_Q^2$ denotes the quadratic function $x^T Q x$, and the input sequence $\mathbf{a}_t = [a_{0|t}, \ldots, a_{N_c-1|t}]$ are the optimization variables. The state constraints (12) are imposed as soft constraints in (15c), with a high penalty S on the constraint violation ϵ. (15d) is the compact representation of the input constraints (13) and (14). (15e) and (15f) are the initial conditions. R and P are matrices of appropriate dimension penalizing the deviation from the reference inputs and the input change rate, respectively.

Parameter	Value	Parameter	Value
Δt_c	0.1s	T	2s
Δt_d	0.2s	$a_{\min}$	-3m/s^2
R	1	$a_{\max}$	3m/s^2
P	0.001	$\dot{a}_{\min}$	-10m/s^3
S	5000	$\dot{a}_{\max}$	10m/s^3

TABLE I
CONTROL DESIGN PARAMETERS

IV. EXPERIMENTAL RESULTS

The proposed autonomous car following system was evaluated in two steps. First we focused on the performance of our driver modeling approach and evaluated its ability to learn different driving styles from different drivers. Then we focused on the performance of the model predictive controller and verified experimentally that the safety constraints are enforced at all times. In this section we describe our experimental platform and present the results obtained for the two evaluations.

A. Experimental platform

Our experimental vehicle is a Hyundai Azera equipped with a forward-looking DELPHI ESR radar which provides the distance and velocity of the preceding vehicle. The computations are performed on a dSPACE MicroAutoBox II embedded computer which consists of an IBM PowerPC processor running at 900MHz. The online optimization problem is solved using the general purpose nonlinear solver NPSOL. The acceleration inputs computed by the MPC are tracked by low-level controllers in the vehicle's Adaptive Cruise Control (ACC) system. The sensors, the embedded computer, and the actuators communicate through a CAN bus.

The parameters used in the control design are listed in Table I. Note that the values of Δt_d and Δt_c are different. In order to obtain a suitable reference for the controller, the reference inputs provided by the driver model at time steps of Δt_d are linearly interpolated at time steps of Δt_c.

B. Driver modeling performance

The driver modeling approach presented in Sec. III-B was evaluated based on the ability of the learned models to reproduce different driving styles from different drivers. To this end we collected highway driving data from 5 human drivers: Driver A, B, C, D, E. Each driver drove for 60 minutes in both low speed (congested) traffic and high speed (free-flow) traffic. The datasets will be referred to as Dataset A, B, C, D, E.

1) Models compared: For each driver we learned two driver models: a *personalized* model and an *average* model.

- *Personalized* driver models are learned from data collected from a specific driver (e.g. *personalized* model of Driver A is learned from Dataset A).
- *Average* driver models are learned from data collected from the other drivers (e.g. *average* model of Driver A is learned from combined Datasets B, C, D, E).

The motivation behind this is to assess whether a model learned using data from a specific driver is able to imitate this person's driving style, compared with a model learned using a collection of drivers.

2) Evaluation data: We use simulation to investigate how the different driver models would react when confronted with the real scenarios occurring in Datasets A, B, C, D, E. It is then possible to evaluate the similarity in driving style between a human driver and a driver model, by comparing the behavior of the driver in a *real* driving sequence with the behavior of the driver model in the corresponding *simulated* driving sequence. For example, we can compare the behavior of the human driver in Dataset A with the behavior of the *personalized* model of Driver A in a *simulated* driving sequence featuring the same scenarios as in Dataset A.

The procedure used to replicate the scenarios from a *real* driving sequence inside a *simulated* driving sequence is as follows. At each time step the longitudinal position and velocity of the preceding vehicle along the road is captured from the *real* driving sequence, and is replayed in a simulation environment. The position of the ego vehicle at this time step is then simulated, using the acceleration computed by the driver model in (1)–(2). This process is repeated until the end of the *real* driving sequence is reached. The *simulated* driving sequence is the collection of the simulated states for all the time steps. The behavior of the preceding vehicle is the same as in the *real* driving sequence; only the behavior of the ego vehicle is different.

3) KS distance comparisons: Driving, when performed by a human, is by nature a dynamic and stochastic process. For such processes, a static error criterion (such as the Root Mean Square Error) based on the difference between the *real* driving sequence and the *simulated* driving sequence at each time step is inadequate to evaluate the similarity between a human driver and a driver model [13]. Instead, driving styles can be characterized by some safety and energy consumption attributes [14]. In this paper we selected the inverse Time-To-Collision (TTCi) and the Vehicle Specific Power (VSP) as indicators to represent driving style. The TTCi quantifies the level of risk that a driver is willing to take in their interactions with the preceding vehicle. The VSP is an indicator of the instantaneous engine load of the vehicle [15].

When comparing the driving styles of two driving sequences, it is common to assume a normal distribution for these indicators and to compare their means and standard deviations [14]. Here we do not make assumptions about the underlying distributions of the indicators and we use the Kolmogorov–Smirnov (KS) distance to compute the similarity in shape and location between the probability distributions of the indicator in the *real* and the *simulated* sequences. The KS distance for an indicator is defined as:

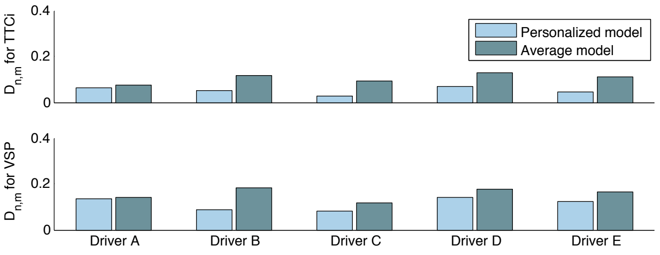

Fig. 3. KS distances obtained for two driving style indicators: TTCi and VSP. A *personalized* model and an *average* model were learned and tested for each driver.

$$D_{n,m} = \max_x |F_n(x) - F_m(x)|$$

where n and m are the number of time steps in the *real* and *simulated* driving sequence (here we have $n = m$), and $F_n(x)$ and $F_m(x)$ are the empirical distribution functions of the indicator for the *real* and *simulated* sequences. For each indicator and for each driver, we compared the KS distance obtained by the *personalized* and the *average* model. To ensure that the testing data is never contained in the training dataset, the results for the *personalized* driver models were obtained with 10-fold cross validation.

Fig. 3 shows the KS distance between the *real* and *simulated* driving sequences for two driving style indicators, for *personalized* and *average* driver models. We notice that the *personalized* models consistently perform better than the *average* models, for all drivers and for both driving style indicators. On average the decrease in the KS distance is 27.0% for the VSP and 49.5% for the TTCi. A smaller KS distance means a higher likeness between two distributions. Therefore the results obtained show that the proposed method for driver modeling is able to reproduce driving styles.

4) Example driving sequence: To further illustrate the ability of the driver models to reproduce the driving styles of the drivers they learn from, Fig. 4 and Fig. 5 show the simulation results obtained on a sample driving sequence, for two different *personalized* models. The driving sequence was taken from Dataset D and is representative of the driving style of Driver D. The longitudinal position and velocity of the preceding vehicle are replayed faithfully in our simulation environment, as described in Sec. IV-B.2. The behavior of the ego vehicle is simulated using the *personalized* model of Driver D[1] in Fig. 4, and using the *personalized* model of Driver E in Fig. 5.

We observed during data collection that the main difference between the driving styles of Driver D and Driver E is the distance they keep with the preceding vehicle. Driver D always keeps a large following distance and increases that distance for higher velocities. Driver E is more aggressive

[1]The testing sequence was removed from the training dataset before learning the *personalized* model of Driver D.

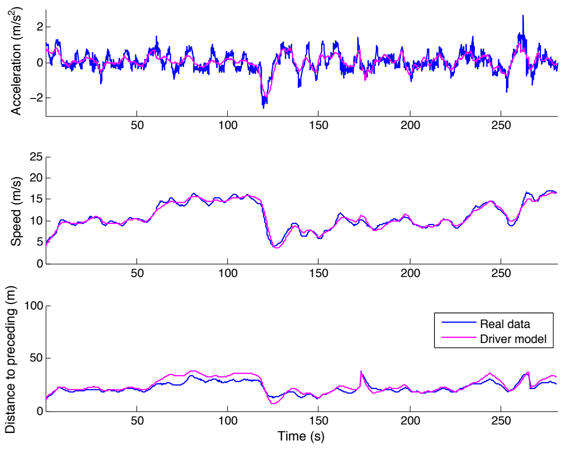

Fig. 4. Replay of real driving sequence from Dataset D, and simulated sequence using the *personalized* model of Driver D. The distance jumps at $t = 175\text{s}$ and $t = 270\text{s}$ correspond to a new preceding vehicle.

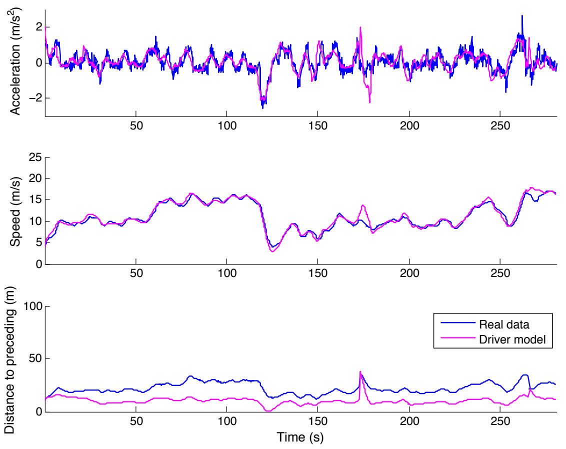

Fig. 5. Replay of real driving sequence from Dataset D, and simulated sequence using the *personalized* model of Driver E. . The distance jumps at $t = 175\text{s}$ and $t = 270s$ correspond to a new preceding vehicle.

and keeps a smaller, relatively constant following distance, regardless of the velocity. The *personalized* models of the two drivers are able to reproduce this difference. Fig. 4 shows that the driver model of Driver D favors large following distances and adjusts this distance based on the velocity similarly to what Driver D did in the real driving sequence. Fig. 5 shows that the driver model of Driver E keeps a constant following distance which is smaller than the distance kept by Driver D in the real driving sequence. It is interesting to note in Fig. 5 is that the ego vehicle is very close to the preceding vehicle at time $t = 120\text{s}$ (30cm). This would not be acceptable for a commercial automated system, and is one of the reasons why our architecture combines Learning-by-Demonstration with Model Predictive Control. In the next section we show with real experiments that the MPC-based controller enforces safe following distances.

C. Controller performance

We demonstrate the ability of the controller to enforce safety constraints through real experiments on our test vehicle. For safety reasons, the experiments were performed at low speed using virtual preceding vehicles. The safety distance was purposefully set to a high value of $d_{\text{safe}} = 5\text{m}$ in order to artificially create situations where the controller needs to deviate from the reference provided by the driver model to keep the vehicle safe. An *average* driver model built using data from multiple drivers was used for the reference generation. We performed two experiments.

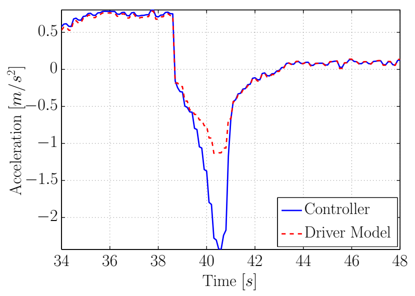

Fig. 6. Experiment 1 - Acceleration commands provided by the driver model and controller. The controller tracks the driver model reference until a violation of the safety distance is predicted around $t = 39\text{s}$. The controller applies harder braking to keep the vehicle safe.

The results for the first experiment are summarized in Fig. 6–9. The vehicle is initially moving with a speed of about 4.3m/s with no vehicle in front of it. The driver model provides a positive acceleration reference which the controller matches closely. At around $t = 38\text{s}$, when the ego vehicle's speed is roughly 6.5m/s, we introduce a virtual preceding vehicle at a relative distance of 8m moving at 5m/s. Although unlikely, such a situation could arise in practice if a vehicle in a neighboring lane makes a sudden lane change into the ego vehicle's lane. The driver model responds by commanding a negative acceleration. However, the controller anticipates a violation of the safety distance and applies a stronger braking. This is seen by the discrepancy between the driver model and controller inputs in Fig. 6. The open-loop predictions of the relative distance to the preceding vehicle made by the controller at $t = 40\text{s}$ are shown in Fig. 7. If the driver model reference was matched exactly, the safety distance of 5m would be violated at $t = 40.8\text{s}$. This is seen by the red line going below the black line in Fig. 7. The controller plans to deviate from the reference input to keep the predicted relative distance (blue line) greater than the safety distance (black line). Fig. 8 shows that the controller does not allow the relative distance to go below the specified safety distance in closed-loop operation. The speeds of the ego and preceding vehicles during the experiment are shown in Fig. 9.

In the second experiment, we show the ability of the controller to handle situations in which the virtual preceding vehicle initially violates the safety constraints. We introduce the virtual preceding vehicle at a relative distance of 4m to the ego vehicle moving at a speed of 5m/s. The initial relative distance is less than the specified safety distance of 5m. Moreover, the ego vehicle is moving faster than the preceding

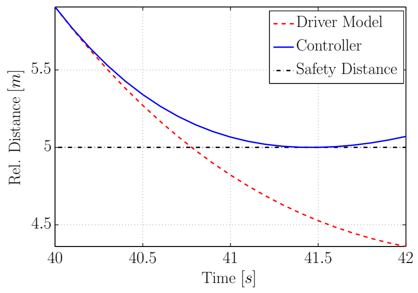

Fig. 7. Experiment 1 - Open loop predictions of the relative distance between the preceding and ego vehicles made at $t = 40$s. The red line depicts the predicted relative distance if the controller tracked the driver model reference exactly. To prevent a constraint violation, the controller plans to deviate from the reference.

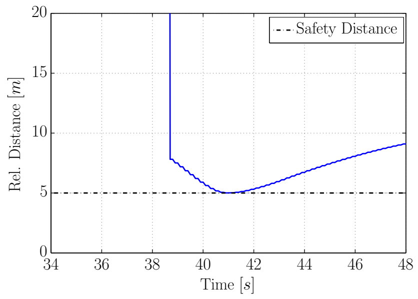

Fig. 8. Experiment 1 - Relative distance between the preceding and ego vehicles. The high values before $t = 38.6$s indicate the absence of any vehicle in front of the ego vehicle in that duration.

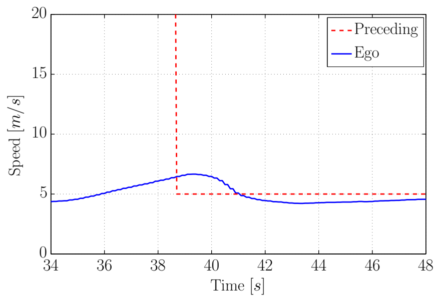

Fig. 9. Experiment 1 - Speeds of the ego and preceding vehicles.

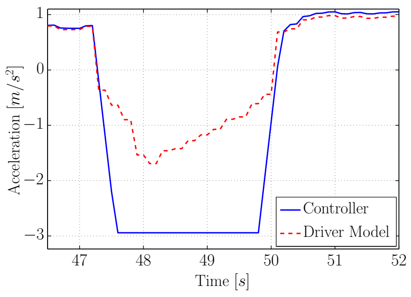

Fig. 10. Experiment 2 - Acceleration commands provided by the driver model and controller. The controller applies maximum braking at around $t = 47$s as the initial relative distance between the ego and preceding vehicles violates the safety distance.

vehicle. As shown in Fig. 10, the controller commands the maximum braking $|a_{\mathbf{min}}|$ as expected.

V. Conclusions and Future Work

The problem addressed in this paper was the longitudinal control of an autonomous vehicle on the highway. Our contribution is to propose a learning-based method which combines a driver model with model predictive control. The driver model generates accelerations which replicate the behavior of a human driver. The controller uses these accelerations as a reference and solves a constrained optimization problem over a finite horizon in order to select the current acceleration to apply. It matches the reference as closely as possible while ensuring that some predefined safety constraints are satisfied. Preliminary results show that we are able to reproduce human driving styles with our autonomous vehicle, while satisfying the safety constraints. Our next step will be to study of how human drivers react to the system. Among other things, this will tell us how the differences in driving styles are perceived by drivers. We will also investigate online learning strategies which would allow the driver model to evolve over time.

References

[1] M. Buehler *et al.*, *The DARPA Urban Challenge - Autonomous vehicles in city traffic*. Springer, 2009.

[2] J. Levinson *et al.*, "Towards fully autonomous driving: systems and algorithms," in *Proc. IEEE Intelligent Vehicles Symposium*, 2011, pp. 163–168.

[3] J. Ziegler *et al.*, "Making bertha drive - an autonomous journey on a historic route," *IEEE Intelligent Transportation Systems Magazine*, vol. 6, no. 2, pp. 8–20, 2014.

[4] A. Broggi *et al.*, "PROUD - public road urban driverless test: Architecture and results," in *Proc. IEEE Intelligent Vehicles Symposium Proceedings*, 2014, pp. 648–654.

[5] J. Wei *et al.*, "A behavioral planning framework for autonomous driving," in *Proc. IEEE Intelligent Vehicles Symposium Proceedings*, 2014, pp. 458–464.

[6] T. Lin *et al.*, "Modeling driver behavior during complex maneuvers," in *Proc. American Control Conference*, 2013, pp. 6448–6453.

[7] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," in *Proc. Advances in Neural Information Processing Systems*, D. S. Touretzky, Ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1989, pp. 305–313.

[8] S. Levine and V. Koltun, "Continuous inverse optimal control with locally optimal examples," in *Proc. International Conference on Machine Learning*, 2012.

[9] S. Lefèvre *et al.*, "Lane keeping assistance with learning-based driver model and model predictive control," in *Proc. 12th International Symposium on Advanced Vehicle Control*, 2014.

[10] A. Carvalho *et al.*, "Stochastic predictive control of autonomous vehicles in uncertain environments," in *Proc. 12th International Symposium on Advanced Vehicle Control*, 2014.

[11] S. Lefèvre *et al.*, "Comparison of parametric and non-parametric approaches for vehicle speed prediction," in *Proc. American Control Conference*, 2014, pp. 3494–3499.

[12] S. Calinon *et al.*, "Learning and reproduction of gestures by imitation," *IEEE Robotics Automation Magazine*, vol. 17, no. 2, pp. 44–54, 2010.

[13] M. Nechyba and Y. Xu, "Stochastic similarity for validating human control strategy models," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 3, pp. 437–451, 1998.

[14] J. Wang *et al.*, "Characterization of longitudinal driving behavior by measurable parameters," *Journal of the Transportation Research Board*, vol. 2185, pp. 15–23, 2010.

[15] H. Frey *et al.*, "Speed- and facility-specific emission estimates for on-road light-duty vehicles on the basis of real-world speed profiles," *Transportation Research Record*, vol. 1987, no. 1, pp. 128–137, 2006.