IEEE TRANSACTIONS ON
AUTOMATION SCIENCE
AND ENGINEERING

A Learning-Based Framework for Velocity Control in Autonomous Driving

Stéphanie Lefèvre, Ashwin Carvalho and Francesco Borrelli

***Abstract*—We present a framework for autonomous driving which can learn from human demonstrations, and we apply it to the longitudinal control of an autonomous car. Offline, we model car-following strategies from a set of example driving sequences. Online, the model is used to compute accelerations which replicate what a human driver would do in the same situation. This reference acceleration is tracked by a predictive controller which enforces a set of comfort and safety constraints before applying the final acceleration. The controller is designed to be robust to the uncertainty in the predicted motion of the preceding vehicle. In addition, we estimate the confidence of the driver model predictions, and use it in the cost function of the predictive controller. As a result, we can handle cases where the training data used to learn the driver model does not provide sufficient information about how a human driver would handle the current driving situation. The approach is validated using a combination of simulations and experiments on our autonomous vehicle.**

***Note to Practitioners*—The objective of this work is to provide a method for personalizing the driving style of an autonomous car. The driving styles are learned from human drivers. The motivation is twofold. Firstly, autonomous cars can benefit from the experience acquired by human drivers over years of driving. Secondly, different drivers have different expectations regarding how an autonomous car should behave. In this paper we implement and test the proposed approach for autonomous car following. We show that the car is able to learn from human drivers and reproduce their driving style.**

***Index Terms*—Learning by demonstration, driver modeling, model predictive control, car following, autonomous driving.**

I. Introduction

The last decade has seen significant achievements in the field of autonomous driving technologies, 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 pre-defined scenarios (e.g. Adaptive Cruise Control, Autonomous Emergency Braking). However, system architectures have not evolved much; current autonomous vehicles use the same architecture as the Urban DARPA Challenge vehicles did [1], [2], [3], [4]. This architecture comprises three main processing modules, described below and illustrated in Figure 1:

- The "Perception + Localization" module combines data received from sensors and digital maps to estimate some relevant features representing the driving situation (e.g. position of other vehicles, road geometry).
- The "Motion planner" selects the appropriate high-level behavior (e.g. car following, lane changing) and generates a trajectory corresponding to 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.

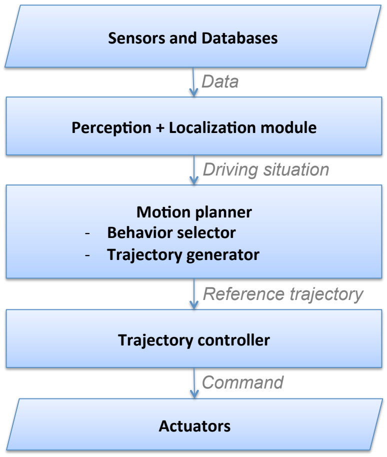

Fig. 1: Standard architecture of an automated ground vehicle (adapted from [3]).

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. Generating reference trajectories which are safe, comfortable, and feasible is a difficult task. As a result, motion planners tend to be very complex and computationally expensive modules [5]. In practice, a lot of tuning is necessary to obtain a good balance of safety, comfort, and feasibility.

A. Related work

A few alternative architectures have been proposed in the literature 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 PieceWise AutoRegressive eXogenous model [6], Maximum Margin Planning [7], Artificial Neural Networks [8], and Inverse Optimal Control [9], [10]. The results demonstrated that it is possible to

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

This work was partially supported by the National Science Foundation under grant No. 1239323 and by the Hyundai Center of Excellence at UC Berkeley.

Digital Object Identifier: 10.1109/TASE.2017.2498192

reproduce human driving behavior in an autonomous car, by learning from example data. However, these approaches do not address the problem of guaranteeing safety. Recent work has shown that Model Predictive Control can be used to enforce safety constraints for autonomous vehicles [11], [12], [13], but these approaches are not robust to errors in the predicted states of the surrounding vehicles. Another issue unaddressed so far is the behavior of a learning-based framework in driving situations which are not present in the training dataset. In such cases, it is not enough to guarantee that the car will be safe. It is also important to ensure that the car will still behave in a reasonable way, or at least to ensure that such situations can be detected.

Another approach, proposed recently by Wei et al. [5], consists of breaking down the "Motion planner" module of Figure 1 into three 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 (PCB) 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; each controller is dedicated to a specific task such as cruise control or lane keeping.

B. Contributions

In this paper, we present a learning-based framework for autonomous driving which combines a driver model and a predictive controller. The learning-based driver model can generate commands resembling those of a human driver. The commands generated by the driver model are used as a reference by a model predictive controller, which is in charge of guaranteeing safety. We implement and test the proposed framework for the longitudinal control of an autonomous car. We address two aspects that were not considered in previous work.

The first aspect is robustness to the uncertainty in the predicted motion of the preceding vehicle. Its acceleration is treated as a bounded disturbance, and the predictive controller is designed to be robust to all possible realizations of this disturbance. In addition, a suitable terminal constraint is introduced to ensure persistent feasibility of the predictive control problem.

The second aspect is the behavior of the system when it encounters driving situations which are not present in the training dataset. We propose a method for evaluating the confidence of the reference accelerations computed by the driver model. This confidence value is used in the cost function of the model predictive controller, making the incentive to match the reference acceleration vary with the confidence value.

The remaining of the paper is organized as follows. Section II presents our autonomous driving framework and formulates the longitudinal control problem. The driver modeling approach is described in Section III, and the controller is presented in Section IV. Experimental results are presented in Section V. Section VI concludes the paper.

II. Problem Formulation

A. Autonomous driving framework

Our autonomous driving framework 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, and provides safety and stability. Our framework inherits the advantages of both. It is illustrated in Figure 2, and the differences with the architecture in Figure 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 commands which imitate the driving style of the driver it learned from. In addition to the reference command, the motion planner outputs a confidence value representing how reliable the reference command is. The confidence will be higher for driving situations which are similar to the training dataset.
- The *trajectory controller* is replaced by a *model predictive controller* which takes as an input the reference command generated by the driver model, and the associated confidence value. The role of the controller is to guarantee safety while trying to match the reference command sent by the driver model. When the confidence value is high and the safety constraints are respected, the command computed by the controller will match the reference command. The controller's command will deviate from the reference command when the confidence value is low or when a safety constraint violation is anticipated.

The driver model and the model predictive controller can be designed independently, and 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. One could also imagine extending the architecture in Figure 2 to take advantage of cloud-based computing and learn new models based on data collected from millions of drivers.

The framework proposed above is general and can be applied to a variety of driving scenarios. So far, we have implemented and tested it for the longitudinal control of an autonomous car during lane keeping. In this scenario, the commanded input is the acceleration of the vehicle. The problem is defined formally below.

B. Longitudinal control problem

We define the following variables to represent the relative motion of the autonomous vehicle (referred to as the "ego vehicle") and the vehicle located ahead in the same lane as the autonomous vehicle (referred to as the "preceding vehicle"):

- $\xi_t = [d_t, v_t]$ is the state of the ego vehicle at time t, where $d_t \in \mathbb{R}^+$ is the longitudinal position of the ego vehicle

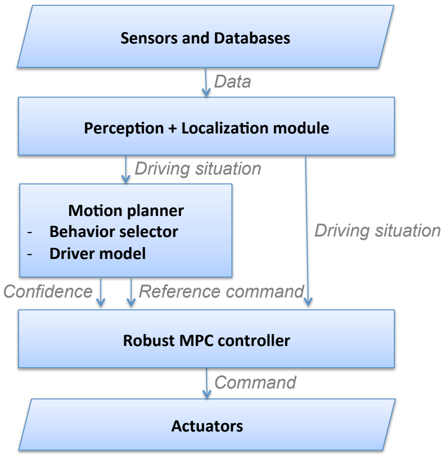

Fig. 2: Proposed architecture combining learning-by-demonstration and model predictive control

in a road-aligned coordinate system, and $v_t \in \mathbb{R}^+$ is the longitudinal velocity of the ego vehicle,

- $\xi_t^p = [d_t^p, v_t^p]$ is the state of the preceding vehicle at time t, where $d_t^p \in \mathbb{R}^+$ is the longitudinal position of the preceding vehicle in a road-aligned coordinate system, and $v_t^p \in \mathbb{R}^+$ is the longitudinal velocity of the preceding vehicle,
- $z_t = [d_t^r, v_t^r, v_t]$ are the features representing the current driving situation at time t, to be used by the driver model to generate an appropriate acceleration command. $d_t^r = (d_t^p - d_t)$ 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 generates an acceleration sequence $\mathbf{a_t^{\text{ref}}} = [a_t^{\text{ref}}, \dots, a_{t+N_d-1}^{\text{ref}}]$, 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 model predictive controller as a reference. The controller solves a constrained optimization problem over the prediction horizon T, and generates a planned acceleration sequence which guarantees the safety of the vehicle. This acceleration sequence is denoted $\mathbf{a_t} = [a_t, \dots, a_{t+N_c-1}]$, 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 how the acceleration sequences are generated by the driver model and the controller.

III. Driver Model

The role of the driver model is to generate an acceleration sequence $\mathbf{a_t^{\text{ref}}}$ which will serve as a reference for the controller, using the history of driving situations $z_{1:t} = [z_1, \dots, z_t]$. 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 [14]. In this work, we use a combination of Hidden Markov Model and Gaussian Mixture Regression (HMM+GMR) [15], a non-parametric regression method which was shown to outperform standard parametric approaches for acceleration prediction in the context of driving [14]. 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.

A. Training phase

During the 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:

- $m_t \in \{1, \dots, M\}$ 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:

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

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 mean and covariance matrix 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, similarly to [15].

B. 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 $z_{1:t}$:

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

where

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

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 $z_{1:t}$:

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

C. Computation of the future accelerations

The driver model must provide the controller with a reference acceleration sequence $\mathbf{a_t^{\text{ref}}} = [a_t^{\text{ref}}, \dots, a_{t+N_d-1}^{\text{ref}}]$ computed using the history of driving situations $z_{1:t}$. This sequence is computed by iteratively applying the driver model defined in (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:

$$d^r_{t+1} = \quad d^r_t + v^r_t \cdot \Delta t_d - \frac{1}{2} a_t \cdot \Delta t_d^2 \tag{4}$$

$$v^r_{t+1} = \quad v^r_t - a_t \cdot \Delta t_d \tag{5}$$

$$v_{t+1} = \quad v_t + a_t \cdot \Delta t_d \tag{6}$$

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

$$z_{t+1} = \quad A z_t + B a_t \tag{7}$$

To summarize, the sequence $\mathbf{a}^{\text{ref}}_{\mathbf{t}}$ is obtained by iterating over the following two equations until the prediction horizon T is reached:

$$\begin{cases} z_{t+1} = & A z_t + B a^{\text{ref}}_t \\ a^{\text{ref}}_{t+1} = & E[a_{t+1} | z_1, \ldots, z_{t+1}] \end{cases} \tag{8}$$

D. Driver model confidence

In practice, it is difficult to collect enough driving samples from human drivers such that the training dataset is representative of all the possible driving situations that an autonomous car may encounter. For example, when collecting training data for lane keeping, we observed that drivers tend to keep their vehicle within a limited set of driving situations they feel comfortable with on a highway. As a result, the training datasets do not contain information about how the drivers would react to uncommon situations, for example if there was a large speed difference between the ego vehicle and the preceding vehicle. In cases like this, the likelihood $P(z_t, a_t | m_t)$ in (1) will be low for every mode, since the observed feature vector z_t is very different from the examples in the training dataset which were used to build the driver model. As a consequence, the output of the Gaussian Mixture Regression in (2) may be inaccurate.

We handle this issue by computing a confidence value ρ_t associated with the driver model's predictions at time t and taking this confidence value into account in the predictive controller, as illustrated in Figure 2. Several metrics were considered to represent the confidence of the driver model, such as the Mahalanobis distances between the features and the different modes, or the variance associated with the driver model's predictions. However, none of these metrics are bounded, and therefore they cannot be used directly in the controller. In the end, we selected the likelihood of the predicted acceleration given the current features. It is an intuitive representation of the uncertainty of the model, and can be made to belong to [0,1] by scaling the variables appropriately. Its computation is detailed below:

$$\begin{aligned} \rho_t \quad &= P(a^{\text{ref}}_t | z_t) = \sum_{i=1}^{M} P(m_i) \cdot P(a^{\text{ref}}_t | z_t, m_i) \qquad (9) \\ &= \sum_{i=1}^{M} P(m_i) \cdot \frac{1}{(2\pi)^{1/2} |S|^{1/2}} e^{(-\frac{1}{2}(a^{\text{ref}}_t - b)^T A^{-1} (a^{\text{ref}}_t - b))} \end{aligned}$$

with

$$S = \Sigma^{aa}_i - (\Sigma^{za}_i)^T (\Sigma^{zz}_i)^{-1} \Sigma^{za}_i$$

and

$$b = \mu^a_i + (\Sigma^{za}_i)^T (\Sigma^{zz}_i)^{-1} (z_t - \mu^z_i)$$

In the next section, we describe the controller and provide details on how the confidence value ρ_t is used to balance the influence of the driver model on the acceleration applied at time t.

IV. Controller

The controller commands the acceleration of the ego vehicle. Its goal is to follow the reference acceleration sequence $\mathbf{a}^{\text{ref}}_{\mathbf{t}}$ provided by the driver model while ensuring that the vehicle operates below the speed limit and does not collide with the preceding vehicle.

We formulate the problem of achieving the aforementioned objectives as a Model Predictive Control (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. [11], 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, and the process is repeated at the next sampling time using the new measurements.

The predictive control problem is complicated by the fact that the preceding vehicle's future motion is not known at the time instant when the optimal control sequence is computed. Most approaches for car-following in the literature make some assumptions about the future behavior of the preceding vehicle. For instance, constant velocity and constant acceleration assumptions are used in [16] and [17], respectively. The predicted positions and velocities of the preceding vehicle are then used to formulate inter-vehicle spacing constraints based on, for example, a desired time-to-collision (see e.g. [16], [17]). However, these approaches do not account for unlikely scenarios such as hard braking by the preceding vehicle, which could result in rear end collisions.

In order to ensure safety, the control design methodology in this paper is based on Robust MPC (RMPC). The main idea is to treat the acceleration of the preceding vehicle as a bounded disturbance, and ensure constraint satisfaction for all possible realizations of this disturbance. Similar worst-case approaches in the context of traffic modeling or autonomous car-following have been presented in [18], [19], [20]. It is shown that safety can be guaranteed if the controller is designed with the assumption that the preceding vehicle applies maximum braking to come to a stop from its current state.

The vehicle model, safety constraints and cost function used to formulate the RMPC problem are described below.

A. Vehicle model

Ego vehicle: A kinematic point-mass model of the ego vehicle is used for the control design. The state update equations are given by,

$$d_{k+1|t} = d_{k|t} + v_{k|t} \cdot \Delta t_c + \frac{1}{2} a_{k|t} \cdot \Delta t_c^2 \tag{10a}$$

$$v_{k+1|t} = v_{k|t} + a_{k|t} \cdot \Delta t_c, \tag{10b}$$

where the variable $x_{k|t}$ denotes the predicted value of x at time $(t+k)$ based on information available at time t. The actual value of x at time $(t+k)$ is denoted as x_{t+k}. 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:

$$\xi_{k+1|t} = C\xi_{k|t} + D a_{k|t}. \tag{11}$$

Preceding vehicle: The motion of the preceding vehicle is also assumed to be governed by the kinematic equations (10) which describe the ego vehicle motion. That is,

$$\xi^p_{k+1|t} = C\xi^p_{k|t} + D a^p_{k|t}. \tag{12}$$

As the future behavior of the preceding vehicle is not known, the proposed RMPC scheme assumes the acceleration a^p_t of the preceding vehicle at time t to be a disturbance, whose bounds are described as,

$$a^p_t \in \mathcal{A} := \{x : a^p_{\min} \leq x \leq a^p_{\max}\}, \tag{13}$$

where $a^p_{\min}$ and $a^p_{\max}$ are the estimated minimum and maximum accelerations, respectively, of the preceding vehicle. The goal of the RMPC design is to satisfy the safety constraints discussed below for all $a^p_t \in \mathcal{A}$.

B. Safety constraints

Actuator limits: Physical limitations on the actuators impose bounds on the control input:

$$a_{\min} \leq a_{k|t} \leq a_{\max} \tag{14}$$

Speed limit: The speed limit is enforced by constraining the speed of the ego vehicle as,

$$v_{k|t} \leq v_{\max}, \tag{15}$$

where $v_{\max}$ is the speed limit of the road.

Safe following distance: The main role of the controller is to guarantee that the ego vehicle never collides with the preceding vehicle for all possible realizations of the disturbance $a^p_{k|t}$ in the set $\mathcal{A}$. This safety requirement is enforced by the following constraint on the relative distance between the preceding and ego vehicles:

$$d^p_{k|t} - d_{k|t} \geq d_{\text{safe}} \quad \forall a^p_{k-1|t} \in \mathcal{A}, \tag{16}$$

where d_{safe} is the minimum safe following distance.

Due to the simple nature of the dynamics (12), disturbance bounds (13) and relative distance constraints (16), robust satisfaction of (16) can be achieved by assuming that the disturbance $a^p_{k|t}$ takes on its lower bound $a^p_{\min}$ at every time step in the prediction horizon. Intuitively, if the preceding vehicle actually accelerates at a value greater than $a^p_{\min}$, the relative distance will be greater than that computed with the worst-case value $a^p_{\min}$. Hence, (16) will be satisfied for all possible values of the disturbance, as is shown formally below.

The predicted worst-case states of the target vehicle, denoted by $\bar{\xi}^p_{k|t} = [\bar{d}^p_{k|t}, \bar{v}^p_{k|t}]$, evolve as,

$$\bar{\xi}^p_{k+1|t} = C\bar{\xi}^p_{k|t} + D a^p_{\min}. \tag{17}$$

In addition, we constrain the speed of the preceding vehicle to be non-negative. The predicted positions $\bar{d}^p_{k|t}$ from (17) are used to formulate the safety distance constraints over the prediction horizon,

$$\bar{d}^p_{k|t} - d_{k|t} \geq d_{\text{safe}}. \tag{18}$$

From the system dynamics in (17), it is easy to show that

$$\bar{d}^p_{k|t} \leq d^p_{k|t} \quad \forall a^p_{k-1|t} \in \mathcal{A}. \tag{19}$$

Hence, satisfaction of (18) ensures that (16) is satisfied.

The state constraints (15) and (18) are concisely expressed as,

$$h^\xi(\xi_{k|t}, \bar{\xi}^p_{k|t}) \leq 0, \tag{20}$$

and the input constraints (14) are written as,

$$h^a(a_{k|t}) \leq 0. \tag{21}$$

C. Persistent feasibility

In general, there is no guarantee that the safety constraints (20) and (21) will be satisfied in closed-loop. This problem of persistent (or recursive) feasibility is well studied in the literature (see [21] for a survey). In RMPC, persistent feasibility can be ensured by introducing a suitably chosen terminal set in which the system state at the end of the horizon is constrained to lie. A sufficient condition for recursive feasibility is that the terminal set is a Robust Control Invariant (RCI) set, which is defined below.

Definition 1: Consider the system $\xi_{t+1} = f_d(\xi_t, u_t, w_t)$, where the state $\xi_t \in \mathcal{X}$, the input $u_t \in \mathcal{U}$ and the disturbance $w_t \in \mathcal{W}$. A set $\mathcal{R} \subseteq \mathcal{X}$ is said to be an RCI set for the system if,

$$\xi_t \in \mathcal{R} \implies \exists u_t \in \mathcal{U} \text{ s.t. } f_d(\xi_t, u_t, w_t) \in \mathcal{R} \quad \forall w_t \in \mathcal{W} \tag{22}$$

Remark 1: In the absence of disturbances in the system, the corresponding set is called a Control Invariant (CI) set.

Definition 2: The RCI (resp. CI) set which contains all other RCI (resp. CI) sets is called the maximal RCI (resp. CI) set.

An RCI set is the set of states from which there exists a feasible input such that the state at the next time step lies within the set for all possible values of the disturbance. Computing an RCI set for the car-following problem is not trivial due to the fact that the motion of the preceding vehicle cannot be controlled. The set of reachable states corresponding to the model (17) given the disturbance bounds (13) includes negative speeds of the preceding vehicle. Hence, standard methods for computing the RCI set for linear systems (see [22] for a survey) would result in a conservative or empty RCI set. Theoretically, the preceding vehicle behaves as a switched (or hybrid) system where the permissible values of the acceleration are such that the vehicle does not attain negative speeds. This, however, significantly complicates the

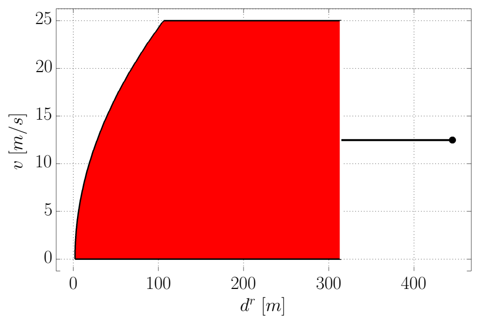

Fig. 3: Projection of the maximal CI set $\mathcal{X}_{ep}$ in the d^r-v space. The black line depicts the direction of unboundedness of $\mathcal{X}_{ep}$.

computation of the RCI set. Moreover, the resulting set usually consists of an union of convex sets, and is non-convex. In this work, we propose a method of computing a polyhedral (hence, convex) terminal constraint for the RMPC problem.

We use the same assumption about the future behavior of the preceding vehicle as is used in the design of the safety constraints (18). Starting from its state $\xi^p_t = [d^p_t, v^p_t]$ at time instant t, the preceding vehicle is assumed to apply maximum braking. Let $(t + k_s)$ denote the time at which the preceding vehicle comes to a stop. The model (17) allows us to compute the predicted worst-case states $\{\bar{\xi}^p_{k|t} = [\bar{d}^p_{k|t}, \bar{v}^p_{k|t}]\}^{k_s}_{k=0}$. Note that $\bar{v}^p_{k|t} = 0$ for $k \geq k_s$. We will show that the worst-case assumption is sufficient to compute the required RCI set.

The computation of the RCI terminal set is broken down into two steps. In the first step, we compute the maximal CI set $\mathcal{X}_{ep}$ for the predicted ego and preceding vehicle states at time $(t + k_s)$, which is defined as follows:

$$\begin{aligned}\mathcal{X}_{ep} = \{[\xi_{k_s|t}, \bar{\xi}^p_{k_s|t}] | \exists a_{k_s|t} \text{ s.t. } &[\xi_{k_s+1|t}, \bar{\xi}^p_{k_s+1|t}] \in \mathcal{X}_{ep},\\ &\xi_{k_s+1|t} = C\xi_{k_s|t} + Da_{k_s|t}, \bar{\xi}^p_{k_s+1|t} = \bar{\xi}^p_{k_s|t},\\ &h^a(a_{k_s|t}) \leq 0, h^\xi(\xi_{k_s|t}, \bar{\xi}_{k_s|t}) \leq 0\}. \end{aligned} \tag{23}$$

The absence of the worst-case disturbance $a^p_{\min}$ in (23) is due to the fact that $\bar{v}^p_{k|t} = 0$ for $k \geq k_s$. Therefore, the preceding vehicle cannot decelerate further. This also allows us to set $\bar{\xi}^p_{k_s+1|t} = \bar{\xi}^p_{k_s|t}$ in (23). The Multi-Parametric Toolbox (MPT) in MATLAB is used to compute $\mathcal{X}_{ep}$ in this paper [23]. The projection of $\mathcal{X}_{ep}$ in the d^r-v space is shown in Figure 3, where $d^r = d^p - d$ is the relative distance between the preceding and ego vehicles. As expected, the maximum permissible ego speed increases with the relative distance. The black line depicts the direction of unboundedness of $\mathcal{X}_{ep}$.

The second step is to compute the set $\mathcal{X}_T$ of states $[\xi_{N_c|t}, \bar{\xi}^p_{N_c|t}]$ at the end of the controller prediction horizon that can be driven into $\mathcal{X}_{ep}$ in $(k_s - N_c)$ number of time steps. Recall that the acceleration of the preceding vehicle is assumed to be constant at $a^p_{\min}$ from time instant $(t + N_c)$ to $(t + k_s)$. The computation of the terminal set $\mathcal{X}_T$ is based on the notion of backward reachable sets, which are defined below.

Definition 3: Consider the system $\xi_{t+1} = f_d(\xi_t, u_t)$, where the state $\xi_t \in \mathcal{X}$, and the input $u_t \in \mathcal{U}$. The one-step backward reachable set to a given target set $\mathcal{T} \subseteq \mathcal{X}$ is defined as,

$$Pre(\mathcal{T}) = \{\xi \in \mathcal{X} | \exists u \in \mathcal{U} \text{ s.t. } f_d(\xi, u) \in \mathcal{T}\} \tag{24}$$

Remark 2: The N-step backward reachable sets $\mathcal{R}_N$ to a given target set $\mathcal{T}$ are recursively defined as,

$$\mathcal{R}_k = Pre(\mathcal{R}_{k-1}) \quad (k = 1, \ldots, N), \quad \mathcal{R}_0 = \mathcal{T}. \tag{25}$$

For our problem, the set $Pre(\mathcal{T})$ for a given target set $\mathcal{T}$ takes the form,

$$\begin{aligned}Pre(T) = \{[\xi_{k|t}, \bar{\xi}^p_{k|t}] | \exists a_{k|t} \text{ s.t. } &[\xi_{k+1|t}, \bar{\xi}^p_{k+1|t}] \in \mathcal{T},\\ &\xi_{k+1|t} = C\xi_{k|t} + Da_{k|t},\\ &\bar{\xi}^p_{k+1|t} = C\bar{\xi}^p_{k|t} + Da^p_{\min},\\ &h^a(a_{k|t}) \leq 0, h^\xi(\xi_{k|t}, \bar{\xi}_{k|t}) \leq 0\}. \end{aligned} \tag{26}$$

We now state and prove the following theorem which allows us to compute a suitable terminal constraint for the RMPC problem.

Theorem 1: Let $\mathcal{X}_T = \mathcal{R}_{k_s - N_c}$, where $\mathcal{R}_{k_s-N_c}$ is the $(k_s - N_c)$-step backward reachable set to $\mathcal{X}_{ep}$. The RMPC problem is persistently feasible with respect to the safety constraints (21) and (20) if the terminal constraint $[\xi_{N_c|t}, \bar{\xi}^p_{N_c|t}] \in \mathcal{X}_T$ is introduced.

Proof: Let $[\xi_{N_c|t}, \bar{\xi}^p_{N_c|t}] \in \mathcal{X}_T = \mathcal{R}_{k_s-N_c}$ as defined above. From (25) and (26), there exists a feasible sequence of control inputs $\{a_{k|t}\}^{k_s-1}_{k=N_c}$ such that the sequence of states $\{[\xi_{k|t}, \bar{\xi}^p_{k|t}]\}^{k_s}_{k=N_c+1}$ satisfies the state constraints (20) and $[\xi_{k_s|t}, \bar{\xi}^p_{k_s|t}] \in \mathcal{X}_{ep}$ for $a^p_{k|t} = a^p_{\min}$ $(k = N_c, \ldots, k_s - 1)$. As $a^p_{\min}$ is the worst-case acceleration of the preceding vehicle, the above statement holds for all possible $a^p_{k|t} \in \mathcal{A}$ $(k = N_c, \ldots, k_s - 1)$. Moreover, by (23), $[\xi_{k_s|t}, \bar{\xi}^p_{k_s|t}] \in \mathcal{X}_{ep}$ implies that for $k \geq k_s$, there exists a feasible control input $a_{k|t}$ which keeps the ego vehicle safe. In summary, the satisfaction of the constraints (20) and the terminal constraint is guaranteed in closed-loop for all possible values of $a^p_{k|t} \in \mathcal{A}$ $(k \geq 0)$. $\square$

Remark 3: As the backward reachability analysis in (25) is performed in discrete-time, the k-step backward reachable set $\mathcal{R}_k$ to the target set $\mathcal{X}_{ep}$ corresponds to a specific value of $\bar{v}^p$. In particular, $\mathcal{R}_k$ corresponds to $\bar{v}^p = -a^p_{\min} k\Delta t_c$. Figure 4 shows the projection of the sets $\mathcal{R}_k$ corresponding to the target set $\mathcal{X}_{ep}$ in the d^r-v^p-v space for $k = 10, 20, 30, 40, 50$. Each $\mathcal{R}_k$ is a two-dimensional plane in the d^r-v space. In order to account for continuous values of $\bar{v}^p$, we set

$$X_T = \mathcal{R}_{k^\star}, \tag{27}$$

where

$$k^\star = \max_{k=1,2,\ldots} k \quad \text{s.t. } \bar{v}^p_{N_c|t} \geq -a^p_{\min} k\Delta t_c. \tag{28}$$

Intuitively, for a given value of $\bar{v}^p_{N_c|t}$, we choose a k-step backward reachable set corresponding to a value of $\bar{v}^p$ less than $\bar{v}^p_{N_c|t}$. Persistent feasibility is still guaranteed as the above choice is conservative.

Remark 4: If $k_s \leq N_c$, $\mathcal{X}_T = \mathcal{X}_{ep}$ is sufficient to guarantee persistent feasibility. This is the case if the preceding vehicle is expected to come to a full stop with maximum deceleration $a^p_{\min}$ within the prediction horizon of the controller.

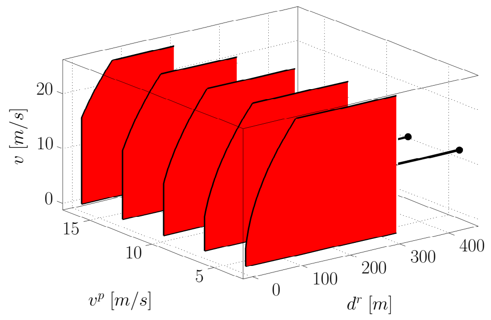

Fig. 4: Projection of the sets $\mathcal{R}_k$ corresponding to the target set $\mathcal{X}_{ep}$ in the d^r-v^p-v space for $k = 10, 20, 30, 40, 50$. Each two-dimensional slice in the d^r-v space corresponds to a particular value of v^p.

D. Cost function

The cost function is defined as,

$$J = \sum_{k=t}^{t+N_c-1} [\rho_k(a_k - a_k^{\text{ref}})^2 + \alpha(a_k - a_{k-1})^2 + \beta(v_k - v_{max})^2] \tag{29}$$

where the reference accelerations a_k^{ref} are provided by the driver model, ρ_k is the confidence factor of the driver model, v_{max} is the speed limit on the current road. The gains α and β are parameters penalizing the jerk and the deviation from the speed limit, respectively. When the driver model is confident about its predictions, ρ_k is large and deviations from the reference accelerations are strongly penalized. When the driver model is unsure about its predictions, ρ_k is small and the controller focuses on driving close to the speed limit and limiting the jerk. The relative importance of the different terms in the cost function can be calibrated by adjusting the gains α and β.

E. MPC formulation

The control input sequence $\mathbf{a}_t$ is computed as the solution of the following constrained finite-time optimal control problem:

$$\min_{\mathbf{a}_t} \; J + \|\epsilon\|_S^2 \tag{30a}$$

$$\text{s.t. } \xi_{k+1|t} = C\xi_{k|t} + Da_{k|t} \tag{30b}$$

$$\bar{\xi}^p_{k+1|t} = C\bar{\xi}_{k|t} + Da^p_{\text{min}} \tag{30c}$$

$$h^\xi(\xi_{k+1|t}, \bar{\xi}^p_{k+1|t}) \leq \epsilon, \quad \epsilon \geq 0 \tag{30d}$$

$$h^a(a_{k|t}, a_{k-1|t}) \leq 0 \tag{30e}$$

$$[\xi_{N_c|t}, \bar{\xi}^p_{N_c|t}] \in \mathcal{X}_T \tag{30f}$$

$$\xi_{0|t} = \xi_t, \quad \bar{\xi}^p_{0|t} = \xi^p_t \tag{30g}$$

$$a_{-1|t} = a_{t-1} \tag{30h}$$

where the notation $x_{k|t}$ denotes the variable x at time $(t + k)$ predicted at time t, and the input sequence $\mathbf{a}_t = [a_{0|t}, \ldots, a_{N_c-1|t}]$ is the optimization variable. The state constraints (20) are imposed as soft constraints in (30d), where $\epsilon \in \mathbb{R}^2$. A high penalty S on the constraint violation is added to the cost function (30a), with $\|x\|_S^2$ denoting the quadratic function $x^T Sx$. (30g) and (30h) are the initial conditions for the state and input, respectively.

Parameter	Value	Parameter	Value
T	2.6s	d_{safe}	5m
a_{min}	-4m/s^2	a_{max}	1.5m/s^2
a^p_{min}	-2.6m/s^2	v_{max}	30m/s
α	4e-3	β	2.4e-3
S	diag(5e3, 1e3)	Δt_c	0.2s
Δt_d	0.2s		

TABLE I: Control design parameters

V. Experimental Validation

In this section, we evaluate the ability of the proposed approach to imitate different driving styles learned from human drivers for the longitudinal control task. Safety is not addressed here, but readers are referred to [12] for experimental results showing that the MPC strategy enforces the safety constraints presented in Section IV-B.

The proposed autonomous car following system is evaluated in two steps. First, we focus on the performance of our driver modeling approach and evaluate its ability to learn different driving styles from different drivers. Then, we study a case when the training data is insufficient for the driver model to generate a reliable reference acceleration. We show that this case is handled smoothly by the system.

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 (30) is solved using the general purpose nonlinear solver NPSOL [24]. 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.

B. Driver modeling performance

In this section, the driver modeling approach presented in Section III is 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: Each driver is assigned 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 four 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-life 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 human 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 real 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 of the preceding vehicle along the road is captured from the *real* driving sequence and 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 (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) Evaluation metrics: 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 [25]. Instead, driving styles can be characterized by some safety and energy consumption attributes [26]. 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 [27].

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 [26]. Here, we chose not to make any assumptions about the underlying distributions of the indicators and we used 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* driving sequence. The KS distance for an indicator is defined as:

$$D_{n,m} = \max_{x} |F_n(x) - F_m(x)| \tag{31}$$

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.

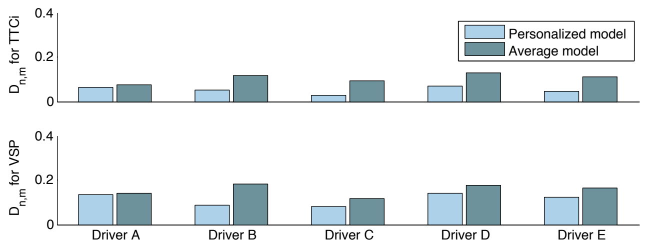

Fig. 5: KS distances obtained for two driving style indicators: TTCi and VSP. A *personalized* model and an *average* model were learned and tested for each driver.

For each indicator and for each driver, we compared the KS distance obtained by the *personalized* and the *average* model. Since the data used for training must be different from the data used for testing, the results for the *personalized* driver models were obtained with 10-fold cross validation.

4) Results: Figure 5 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 learn from drivers and reproduce their specific driving styles.

The datasets used so far for evaluation were real data collected from human drivers and contain only ordinary driving situations. These are situations where the learned driver models performs well and can be trusted. In the next section, we evaluate the performance of the system when faced with less common driving situations.

C. Driver model confidence

The real data used to learn a driver model contains mostly driving situations which are comfortable for the driver. As explained in Section III-D, the driver model may therefore generate unrealistic acceleration commands when faced with a driving situation which is very different from the training dataset. This is the reason why we introduced the confidence value ρ_t, which indicates how reliable the driver model predictions are at time t. In this section, we conduct experiments in real-time on our test vehicle and analyze the behavior of the confidence value.

In a first example, we use the *personalized* model of Driver A in a real stop-and-go scenario. Stop-and-go traffic is known to be challenging for longitudinal controllers, because goals of safety and comfort most often oppose each other [28]. Since Dataset A contains examples of traffic jams, we expect that the confidence of the driver model should be high and that the controller should match the driver model reference exactly, unless the safety constraints get violated. Figure 6 shows plots of the distance, speed, acceleration, and driver

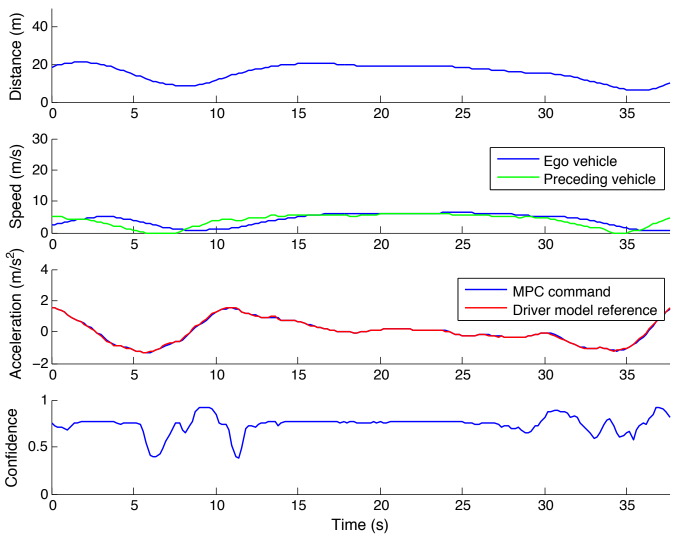

Fig. 6: Testing a situation where the driver model is confident about its predictions. The driver model is able to handle the stop-and-go situation, therefore the controller matches the reference acceleration closely.

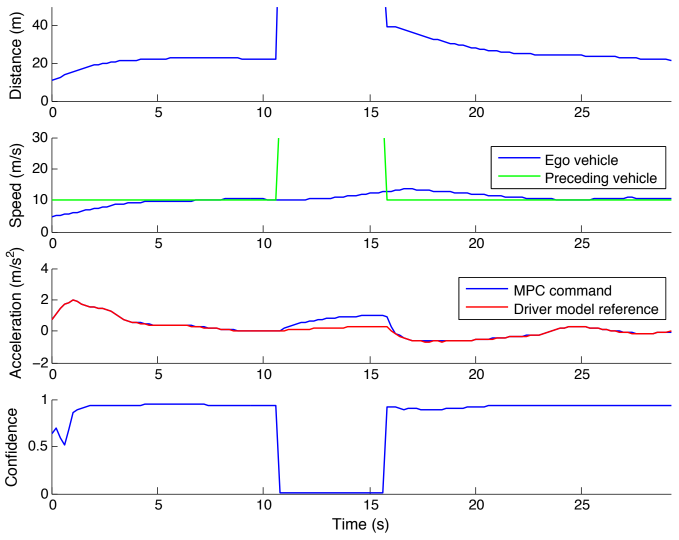

Fig. 7: Testing a situation where the driver model reference should not be trusted. The confidence drops immediately when the ego vehicle finds itself driving at low speed with no preceding vehicle, which is an uncommon situation not present in the training dataset. The controller is able to handle the situation: it disregards the acceleration reference and focuses on bringing the ego vehicle up to the speed limit.

model confidence for this test. Initially, the ego vehicle is 20m behind the preceding vehicle and both vehicles drive at a low speed. Over the next 40s the preceding vehicle speeds up, slows down, and comes to a full stop twice. The driver model adjusts the acceleration to remain at a comfortable distance to the preceding vehicle, ranging from 20m when the preceding vehicle is moving to 8m when it is stopped. The confidence value remains high to moderate the entire time, and the controller matches the driver model's acceleration closely. This is consistent with the goal stated in Section III-D that the driver model's reference should be followed when it is safe to do so and the driver model is confident. From the passengers' perspective, the traffic jam situation is handled very naturally by the autonomous vehicle.

In a second example, we use the *personalized* model of Driver A in a driving situation deliberately created to be very different from any driving situation present in Dataset A. Instead of using radar measurements to detect a preceding vehicle, we create a virtual preceding vehicle driving at a low, constant speed. Figure 7 shows plots of the distance, speed, acceleration, and driver model confidence for this test. The initial conditions are the following: the ego vehicle is 10m behind the preceding vehicle and drives at 5m/s. The preceding vehicle drives a little faster, 10m/s. This is a common driving situation, and the acceleration references generated by the driver model come with a large confidence value. Since the reference accelerations do not violate the safety constraints, and the confidence is high, the controller matches the reference closely. After a few seconds, the ego vehicle has stabilized itself at a distance of 20m behind the preceding vehicle, and drives at the same speed. At time t=11s, the virtual preceding vehicle is removed, creating an unusual driving situation where the ego vehicle is driving at a low speed and there is no vehicle in front. As expected, the confidence of the driver model predictions immediately drops. For the next 5s, the driver model generates a very low acceleration reference, which is clearly not appropriate for the situation. Because of the low confidence of the driver model, the controller does not get penalized much for deviating from the acceleration reference, and commands an appropriate, stronger acceleration which eventually would bring the speed of the car up to the speed limit. At time t=16s, the virtual obstacle is reintroduced, making the driving situation familiar again for the driver model. The reference accelerations generated by the driver model can be trusted; they are matched by the controller. After a few seconds, the ego vehicle once again reaches a stable speed and distance to the car in front.

VI. Conclusions

We presented a learning-based architecture for autonomous driving which combines a driver model and a model predictive controller. The driver model is learned from data and can imitate different driving styles. The model predictive controller guarantees safety by enforcing a set of constraints on the vehicle state. The system can detect and handle driving situations where the driver model should not be trusted. We implemented and tested the proposed framework for the longitudinal control of an autonomous car. The ability of the system to reproduce driving styles learned from human drivers was demonstrated in simulation with replayed real data. We also analyzed the behavior of the system in two example experiments in our autonomous car, which demonstrated that the proposed approach is able to handle both ordinary driving situations and uncommon ones. These results demonstrate that the proposed approach performs well on the task it was designed for, that is, learn from driving data and imitate driving styles while ensuring safety. In order to evaluate the proposed approach in terms of benefits perceived by human drivers, it will be necessary to conduct a naturalistic study comparing it with a commercial ACC system.

Our future research will be articulated around two axes. The first one is the continuous improvement of the system's performance through feedback from drivers. Indeed, our preliminary results while testing on highways suggest that while drivers appreciate the opportunity to personalize the driving style of the autonomous car, there remain situations where they wish the car would behave differently. In particular, aggressive drivers may not always feel comfortable with the autonomous car driving as aggressively as they did themselves during the collection of the training dataset. We plan to investigate control sharing and lifelong learning techniques which would allow the driver to occasionally take control of the autonomous car to demonstrate how they wish the driving situation to be handled. The autonomous vehicle would use this feedback to adjust its driving style, and this way would be able to match the human's expectations even as their preferences change with time.

Our second objective is be to apply the proposed learning-based framework to the lateral control of the vehicle. Similarly to what was done in this paper for car following, we wish to use demonstrations to learn driver preferences in terms of when and how to change lanes on highways. The autonomous car would then reproduce the learned behavior. However, we expect to find that the approach used in this paper to model drivers is too simple to accurately represent lane change preferences. Instead of using regression, we will consider Inverse Reinforcement Learning approaches. By using these techniques to retrieve cost functions from data, we will model the decision making process of drivers while they change lanes on highways.

References

[1] M. Buehler, K. Iagnemma, and S. Singh, *The DARPA Urban Challenge - Autonomous vehicles in city traffic*. Springer, 2009.

[2] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Kolter, D. Langer, O. Pink, V. Pratt, M. Sokolsky, G. Stanek, D. Stavens, A. Teichman, M. Werling, and S. Thrun, "Towards fully autonomous driving: systems and algorithms," in *Proc. IEEE Intelligent Vehicles Symposium*, 2011, pp. 163–168.

[3] J. Ziegler, P. Bender, M. Schreiber, H. Lategahn, T. Strauss, C. Stiller, T. Dang, U. Franke, N. Appenrodt, C. G. Keller, E. Kaus, R. G. Herrtwich, C. Rabe, D. Pfeiffer, F. Lindner, F. Stein, F. Erbs, M. Enzweiler, C. Knöppel, J. Hipp, M. Haueis, M. Trepte, C. Brenk, A. Tamke, M. Ghanaat, M. Braun, A. Joos, H. Fritz, H. Mock, M. Hein, and E. Zeeb, "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, P. Cerri, S. Debattisti, M. Laghi, P. Medici, M. Panciroli, and A. Prioletti, "PROUD - public road urban driverless test: Architecture and results," in *Proc. IEEE Intelligent Vehicles Symposium Proceedings*, 2014, pp. 648–654.

[5] J. Wei, J. Snider, T. Gu, J. Dolan, and B. Litkouhi, "A behavioral planning framework for autonomous driving," in *Proc. IEEE Intelligent Vehicles Symposium Proceedings*, 2014, pp. 458–464.

[6] T. Lin, E. Tseng, and F. Borrelli, "Modeling driver behavior during complex maneuvers," in *Proc. American Control Conference*, 2013, pp. 6448–6453.

[7] D. Silver, J. Bagnell, and A. Stentz, "Learning autonomous driving styles and maneuvers from expert demonstration," in *Experimental Robotics*, ser. Springer Tracts in Advanced Robotics, J. P. Desai, G. Dudek, O. Khatib, and V. Kumar, Eds. Springer International Publishing, 2013, vol. 88, pp. 371–386.

[8] 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.

[9] P. Abbeel, D. Dolgov, A. Ng, and S. Thrun, "Apprenticeship learning for motion planning with application to parking lot navigation," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008.

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

[11] A. Carvalho, Y. Gao, S. Lefèvre, and F. Borrelli, "Stochastic predictive control of autonomous vehicles in uncertain environments," in *Proc. 12th International Symposium on Advanced Vehicle Control*, 2014.

[12] S. Lefèvre, A. Carvalho, and F. Borrelli, "Autonomous car following: a learning-based approach," in *Proc. IEEE Intelligent Vehicles Symposium*, 2015.

[13] S. Lefèvre, A. Carvalho, Y. Gao, H. E. Tseng, and F. Borrelli, "Driver models for personalised driving assistance," *Vehicle System Dynamics*, 2015.

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

[15] S. Calinon, F. D'halluin, E. Sauser, D. Caldwell, and A. Billard, "Learning and reproduction of gestures by imitation," *IEEE Robotics Automation Magazine*, vol. 17, no. 2, pp. 44–54, 2010.

[16] V. L. Bageshwar, W. L. Garrard, and R. Rajamani, "Model predictive control of transitional maneuvers for adaptive cruise control vehicles," *IEEE Transactions on Vehicular Technology*, vol. 53, no. 5, pp. 1573–1585, 2004.

[17] S. Li, K. Li, R. Rajamani, and J. Wang, "Model predictive multi-objective vehicular adaptive cruise control," *IEEE Transactions on Control Systems Technology*, vol. 19, no. 3, pp. 556–566, 2011.

[18] P. G. Gipps, "A behavioural car-following model for computer simulation," *Transportation Research Part B: Methodological*, vol. 15, no. 2, pp. 105–111, 1981.

[19] C. Chien and P. Ioannou, "Automatic vehicle-following," in *1992 American Control Conference*, 1992, pp. 1748–1752.

[20] L. Alvarez and R. Horowitz, "Safe platooning in automated highway systems part I: Safety regions design," *Vehicle System Dynamics*, vol. 32, no. 1, pp. 23–55, 1999.

[21] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[22] F. Blanchini, "Survey paper: Set invariance in control," *Automatica*, vol. 35, no. 11, pp. 1747–1767, 1999.

[23] M. Herceg, M. Kvasnica, C. Jones, and M. Morari, "Multi-Parametric Toolbox 3.0," in *2013 European Control Conference*, 2013, pp. 502–510.

[24] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright, "User's guide for npsol (version 4.0)," 1986.

[25] 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.

[26] J. Wang, M. Lu, and K. Li, "Characterization of longitudinal driving behavior by measurable parameters," *Journal of the Transportation Research Board*, vol. 2185, pp. 15–23, 2010.

[27] H. Frey, N. Rouphail, and H. Zhai, "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.

[28] J.-J. Martinez and C. Canudas-de Wit, "A safe longitudinal control for adaptive cruise control and stop-and-go scenarios," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 2, pp. 246–258, 2007.