2023 IEEE 26th International Conference on
Intelligent Transportation Systems (ITSC)
24-28 September 2023. Bilbao, Bizkaia, Spain

Inverse Reinforcement Learning and Gaussian Process Regression-based Real-time Framework for Personalized Adaptive Cruise Control

Zhouqiao Zhao, Xishun Liao, Amr Abdelraouf, Kyungtae Han, Rohit Gupta, Matthew J. Barth and Guoyuan Wu

2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC) | 979-8-3503-9946-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/ITSC57777.2023.10422413

Abstract— **Adaptive Cruise Control (ACC) has become increasingly popular in modern vehicles, providing enhanced driving safety, comfort, and fuel efficiency. However, predefined ACC settings may not always align with a driver's preferences, leading to discomfort and possible safety hazards. To address this issue, Personalized ACC (P-ACC) has been studied by scholars. However, existing research mostly relies on historical driving data to imitate driver styles, which ignores real-time feedback from the driver. To overcome this limitation, we propose a cloud-vehicle collaborative P-ACC framework, which integrates real-time driver feedback adaptation. This framework consists of offline and online modules. The offline module records the driver's naturalistic car-following trajectory and uses inverse reinforcement learning (IRL) to train the model on the cloud. The online module utilizes the driver's real-time feedback to update the driving gap preference in real-time using Gaussian process regression (GPR). By retraining the model on the cloud with the driver's takeover trajectories, our approach achieves incremental learning to better match the driver's preference. In human-in-the-loop (HuiL) simulation experiments, the proposed framework results in a significant reduction of driver intervention in automatic control systems, up to 70.9%.**

I. Introduction

Over the last decade, there has been a significant increase in the development and adoption of vehicle automation. With advancements in technology, vehicles have become more intelligent and capable of performing tasks that were previously the sole responsibility of the driver. Also, Advanced Driver Assistance Systems (ADAS) has been remarkably improved, providing drivers with a range of features, including Adaptive Cruise Control (ACC), Lane Departure Warning (LDW), and Automated Emergency Braking (AEB) [1]–[3]. These technologies not only improve the driving experience but also have the potential to reduce accidents, injuries, and fatalities on the road [4].

Despite the potential benefits of ADAS and vehicle automation in enhancing road safety, energy efficiency, and driving comfort, the lack of personalization may lead to various problems. When the settings are not personalized, drivers may experience discomfort, reduced trust in the automation system, decreased usage, and increased risk of accidents due to unintended operations. Moreover, drivers have unique habits and preferences, so a universal approach may not be suitable for everyone [5]. Personalization of ADAS vehicle automation can help solve the aforementioned issues by adapting the system to meet the specific needs and preferences of each driver [6].

In this paper, we present a novel P-ACC framework that combines both offline and online learning. The offline learning is achieved through the Inverse Reinforcement Learning (IRL) algorithm, which is trained from the naturalistic car-following trajectories of individual drivers to infer their driving style and create a driving gap preference table (DGPT) that is used as a reference for the P-ACC system. Unlike other P-ACC methods that simply clone the driver's behavior from the historical data, our offline model takes into account the driver's task-specific preferences. The online learning component adapts the DGPT in real-time based on the driver's feedback through ACC overrides. This is achieved using Gaussian Process Regression (GPR) [7], which is a statistical method that can be used for model-free prediction and estimation tasks. The driver's feedback data is also used to update the offline personalization module as an incremental learning scheme. Our proposed P-ACC system provides a personalized driving experience for each driver, resulting in enhanced driving comfort and safety.

Compared to the existing literature that studied the personalization of ACC systems, we make the following contributions in this work:

- The proposed framework is embedded with a cloud-vehicle architecture. The cloud side enables the P-ACC system to learn from a large number of drivers' data and adapt to their unique driving styles in different scenarios. The vehicle side enables flexible and rapid adaptation to drivers' real-time feedback.
- The proposed framework employs an offline-online scheme to personalize driving. The offline module uses the IRL algorithm to extract the individualized driving behavior of each driver using demonstration trajectories. The online module utilizes a GPR algorithm to dynamically modify the P-ACC system's actions based on the driver's real-time feedback. By combining the IRL and GPR algorithms, the P-ACC system provides a

Z. Zhao, X. Liao, M. Barth, and G. Wu are with College of Engineering, University of California, Riverside, 1084 Columbia Avenue, Riverside, CA 92507 (e-mail: zzhao084@ucr.edu; xliao016@ucr.edu; barth@ece.ucr.edu; gywu@cert.ucr.edu).

A. Abdelraouf, K. Han, and R. Gupta are with InfoTech Labs, Toyota Motor North America R&D, 465 Bernardo Avenue, Mountain View, CA 94043 (e-mail: amr.abdelraouf@toyota.com; kyungtae.han@toyota.com; rohit.gupta@toyota.com).

customized driving experience that matches the driver's preferences.

- The proposed framework is evaluated in HuiL driving simulator to evaluate its effectiveness. The experiment involves multiple drivers with diverse driving styles, and the results demonstrate the superior performance of our P-ACC system compared to existing ACC systems.

The remainder of this work is organized as follows: Section II reviews the latest literature in the related field. Section III introduces the problem formulation. Section IV elaborates on the proposed system. In Section V, we conduct numerical experiments on the naturalistic driving data and human-driving experiments on a game engine-based simulator to test the validity of the model. Finally, the study is concluded with some future directions in section VI.

II. Related Work

A. Car-following Model

The design of a P-ACC system involves both driving behavior modeling and personalized car-following controller adaptation. Some methods address these two tasks separately, while most the others integrate them into an end-to-end control scheme. The car-following model is the foundation of an ACC system, which is designed to replicate and optimize the behavior of a human driver in maintaining a safe and comfortable distance from the vehicle in front of them while driving. Existing literature on car-following modeling can be broadly classified into several categories, including Ordinary Differential Equation (ODE), Model Predictive Control (MPC), Inverse Reinforcement Learning (IRL), Gaussian Process Regression (GPR), and Sequential models.

ODE-based policies, as one of the most prevalent methods for vehicle longitudinal control, aim to enable the ego vehicle to follow the movement of the preceding vehicle based on physics laws. However, designing corresponding algorithms requires prior knowledge of the car-following system, making them generic and difficult to personalize. Moreover, ODE-based policies lack expressivity, which makes it hard to capture the nuances of naturalistic human driving. Studies like [8], [9], and [10] fall under this category.

On the other hand, MPC methods optimize predefined objectives like safety, comfort, and fuel efficiency in a receding-horizon fashion [11]. Similar to ODE-based policies, designing MPC policies also requires prior knowledge of the car-following system, making them generic and difficult to personalize.

IRL is another popular approach to learn personalized car-following behaviors. Researchers in [12] and [13] use IRL to learn the reward of car-following demonstration trajectories and implement the recovered reward using controllers. The IRL algorithms used in these studies can recover personalized car-following gap preferences based on different vehicle speed values, which can be used to design the downstream control logic for P-ACC systems.

Gaussian Process Regression is a direct approach that looks into the data and learns from demonstration trajectories. Researchers in [14] propose a Gaussian Process Regression algorithm for P-ACC, where both numerical and human-in-the-loop experiments verify the effectiveness of the proposed algorithm in reducing the interference frequency by the driver. One advantage of GPR is its non-parametric nature, which avoids assumptions about the underlying distribution of driving data and provides a measure of uncertainty in predictions. However, GPR may not be suitable for high-dimensional problems due to computational expense and the curse of dimensionality. Therefore, using GPR to model car-following as an end-to-end problem is not recommended.

Finally, since the decision-making process of human drivers depends on sequential state inputs, Recurrent Neural Network (RNN) and Long Short Term Memory (LSTM) have also been used to model car-following behaviors. Studies like [15] and [16] fall under this category.

B. Personalized Driving Behavior Modeling

While driving behavior and preferences can be diverse among drivers, there is a growing demand to explore personalized driving behavior to enhance the safety and user experience of the current ACC system.

Driving style is widely adopted to modeling the personalized behavior in a high-level, as it can provide valuable insights into a driver's habits, preferences, and tendencies. Considering the driving style divergence among drivers, [17] proposed a P-ACC with driving style identification and corresponding personalized speed-distance control. The driving style is characterized by fitting driving data of each individual driver into a Gaussian mixture model (GMM) and clustered by Kullback–Leibler (KL) divergence. Instead of only considering ego driver, personalized driving style is also depend on the environment. [18] employed Conditional Variational Auto-Encoder to model a probabilistic distribution of the individual's driving style considering surrounding vehicles, to facilitate the prediction for a driver's longitudinal acceleration and speed.

Besides modeling a high-level driving style, Imitation Learning is another popular approach to model personalized driving behavior from demonstration. By observing the demonstration of the studied individual, IRL was implemented to recover the cost function [19] [20] for representing a driver's preference or a reward [12] for optimal policy. Similarly, Generative Adversarial Imitation Learning (GAIL) was used to learn the personalized car-following strategy only based on drivers' demonstrations but without specifying the reward [21].

Moreover, researchers developed end-to-end approaches with integrating personalized behavior implicitly. To model the uncertainty of human behavior, a Gaussian Process Regression [14] was adopted to learn the personalized longitudinal driving behavior model, which is a joint Gaussian distribution mapping from the driver's perceived states to

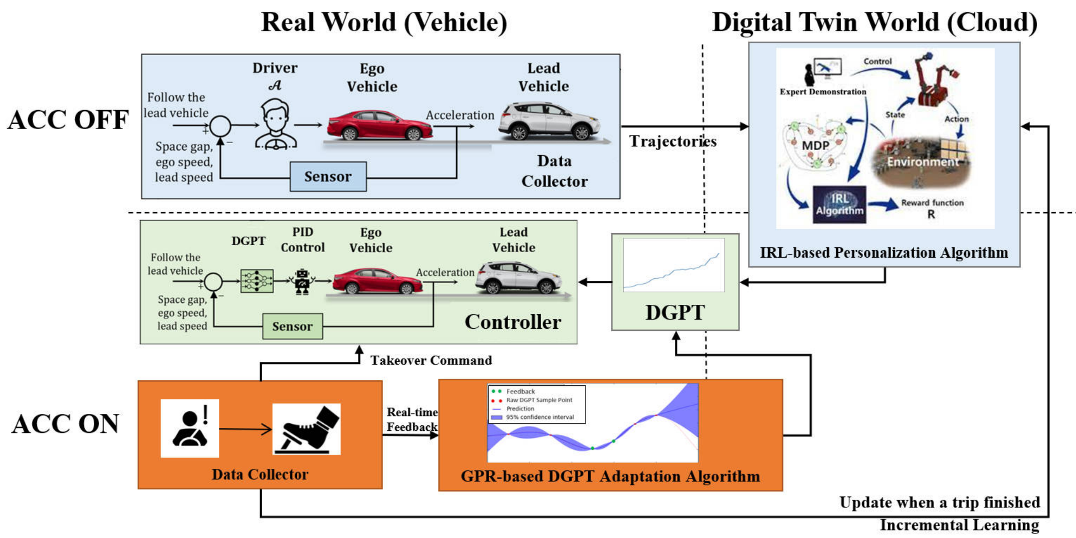

Fig. 1. System architecture: Offline learning (blue blocks), Online learning (orange blocks), Personalized controller (green blocks).

control outputs. [22] utilized constraint Delaunay triangulation to identify a safe area and the fuzzy linguistic preference relation (FLPR) method to determine drivers' driving preferences. With taking the user's personalized objectives as input, this method achieved personalized trajectory planning and lane-change control, meeting users' diverse preferences while ensuring vehicle safety.

However, the mentioned driving behavior modeling can be affected by many factors, such as weather, road conditions, and the emotional state of the driver [23], [24]. Existing literature only relies on historical data for personalized driving behavior modeling, which may not account for changes in external factors [25]. Thus, to improve the flexibility and effectiveness of P-ACC, it is crucial to incorporate real-time data and driver feedback to dynamically adjust the car-following policy.

III. Problem Formulation

A. System Architecture

This paper proposes a vehicle-cloud framework as an extension of our previous study in "Digital Twin" [12]. The Digital Twin system uploads naturalistic driving trajectory data and ACC feedback data to the cloud, where personalized models for different drivers are trained and maintained. By sharing the computational burden of the training process in the cloud, the system is able to efficiently handle the high computational demand. In addition, the system can reuse remote driver models for similar driving behaviors and fine-tune the same model for federated learning. The cloud can also maintain different models for different driving scenarios to achieve more precise personalization for the same or similar types of drivers facing different driving conditions. For a more detailed explanation and implementation of the Digital Twin system, please refer to our previous article.

Fig. 1 shows the general system architecture of the proposed P-ACC framework. Different from our previous work, where the personalization model only relied on the historical demonstration car-following trajectory (blue blocks in Fig. 1), we introduces a novel approach to incorporate the driver's real-time feedback on the ACC system as a dynamic input to adjust the model in this work (orange blocks in Fig. 1). Based on our literature review, no previous studies have considered the driver's real-time feedback on the ACC system. The physical layer of the framework is divided into the real world (vehicle) and the digital twin world (cloud), while the implementation process is divided into two phases: ACC OFF and ACC ON.

At the ACC OFF phase, when the driver manually follows the lead vehicle, the system considers the trajectory as an expert demonstration and transmits it to the cloud along with environmental factors that could potentially impact driving behavior. On the cloud, the IRL algorithm assumes that the collected expert demonstration is near-optimal in terms of the Markov Decision Process (MDP) and infers the reward function that drives the driver's behavior. This reward function is then transferred to the DGPT as the control reference.

At the ACC ON phase, when the driver turns on the automatic following mode, the personalized driving model (i.e., DGPT) is downloaded locally. The DGPT is a discrete table that records the relationship between the preferred following distance and its corresponding speeds. We employ GPR to represent the DGPT, allowing for continuous outputs that can be updated as new data points (driver's feedback) come. The DGPT is designed to describe the driver's preferred following distance at different speeds. The controller (green blocks in Fig. 1) maintains the distance to the lead vehicle. However, due to differences between the scenario of demonstration trajectories and the current driving scenario, as well as changes in the driver's driving habits or mood, the driver may not always be satisfied with the current automated control. Therefore, the driver can provide feedback to the system by pushing the accelerator to shorten the car-following gap or brake pedals to lengthen the gap, leading to a takeover of the vehicle. These takeover segments are used to adjust the DGPT in real-time, responding to the driver's behavior. What's more, when the current ACC ON trip is completed, these takeover segments are sent back to the cloud to fine-tune the IRL model, improving the personalization of the system.

B. Assumptions and Specifications

This paper specifically focuses on the modeling and control of personalized car-following maneuvers based on the states of the ego vehicle and its preceding vehicle. The focus is solely on the car-following stage of an ACC system, where the assumption is made that the preceding vehicle is always present. Additionally, only the longitudinal movement of the vehicle is observed and controlled. The objective of this research is to design a speed control strategy for the ego vehicle that aligns with the driver's preferences.

We assume that the personalized car-following behavior of a driver can be described by the look-up table, DGPT, which represents the preferred following distance of drivers at different speeds. Therefore, we also describe the car-following dynamic model in a 2D space spanned by the speed v and the distance g to the preceding vehicle. We use a second-order approximation and discretize the space and speed, using the following equation:

$$v[t+1] = v[t] + a[t] \cdot \Delta t + \sigma_v \tag{1}$$

$$g[t+1] = g[t] + (v_f[t] - v[t]) \cdot \Delta t + \frac{1}{2} \cdot a[t] \cdot \Delta t^2 + \sigma_g \tag{2}$$

As shown in Equation (1) and (2), we add Gaussian noises, σ_v and σ_g, denoting imperfectness of the driver's observation and control. The action space is discretized to facilitate modeling (ranging from -8m/s^2 to 3m/s^2 with a resolution of 0.1 m/s^2), while maintaining continuity during control.

We presume the driver's decision-making process is a MDP defined by a five-tuple $\{S, U, T, r, \gamma\}$, where S is the state space spanned by v and g; U is the one-dimensional action space of all possible acceleration of the ego vehicle; T is the transition probability determined based on Equation (1) and (2); r is the reward function that represents the driver's personalized car-following style; and γ is the discount factor weighting the importance of the historical rewards; At each time step, the process in certain state v and g, and the driver may choose any action a. The process responds at the next time step by moving into a new state s' based on T. Notably, although the speed of the preceding vehicle, v_f, is considered in MDP, it can be observed while driving. It should be noted that we also assume the human driver is rational and his/her actions are optimizing a cumulative reward function formulated as follows:

$$v(\xi) = \sum_{t=0}^{N} \gamma^t \cdot r_t(s) = \sum_{t=0}^{N} \gamma^t \cdot \boldsymbol{\alpha}^T \cdot \boldsymbol{\Phi}(s) \tag{3}$$

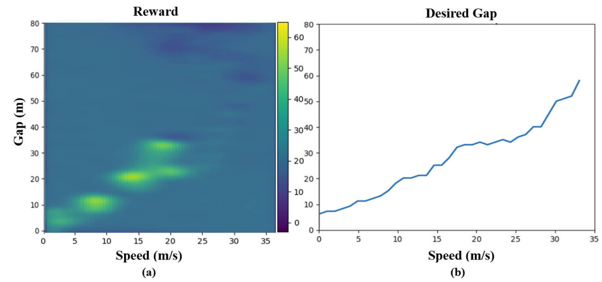

Fig. 2. Personalized driving preference from IRL modeling: (a) recovered reward function from naturalistic driving data using IRL, (b) smoothed *speed-*$g_{desired}$ table.

where N denotes the time horizon, and ξ denotes the trajectory of the ego vehicle. As seen in Equation (3), the instantaneous reward $r_t(s)$ is assumed to be expressed in a span of the reward basis $\boldsymbol{\Phi}$, whose dimension equals the total number of features, and $\boldsymbol{\alpha}$ stands for a vector of weight defining the linear combination. Additionally, it is assumed that the collected trajectory can reflect the drivers' driving style and that drivers are comfortable with their own driving style.

IV. Methodology

In this section, we present a detailed description of the proposed system, which includes modeling the driver's preference using the IRL algorithm, the online adaptation algorithm of DGPT based on drivers' feedback, and controller design for following the preceding vehicle.

A. IRL-based offline personalized DGPT learning

The input of the IRL algorithm is from either the demonstration trajectories when ACC is deactivated (for modeling) or the takeover trajectories while ACC is activated (for fine tuning). As we assume that the demonstration trajectory we collected represents the optimal policy $\pi^*(s, a)$ for the Equation (3), the goal of the IRL algorithm is to recover the linear coefficients $\boldsymbol{\alpha}$. The reward basis $\boldsymbol{\Phi}$ is predefined with sufficient descriptive ability in the space of v and g. By using IRL, we can recover the reward function, as shown in Fig. 2 (a). The detailed process for the algorithm can be found in our previous work [12].

The DGPT (i.e., the preferred gap at different speeds) is calculated using Equation (4) from the recovered reward function r, as shown in Fig. 2 (b). Also, a low pass filter is applied afterward to ensure smoothness.

$$DGPT(v) = g_{\text{desired}}(v) = \arg\max_g r(v) \tag{4}$$

B. Online DGPT Adaptation

Compared to the naturalistic car-following data that is used for training the IRL model, the online feedback data usually has a very short time period and distribute sparsely in the time domain. Therefore, preprocesses are required to use this data to maintain the DGPT, as illustrated in blue blocks of Fig. 3.

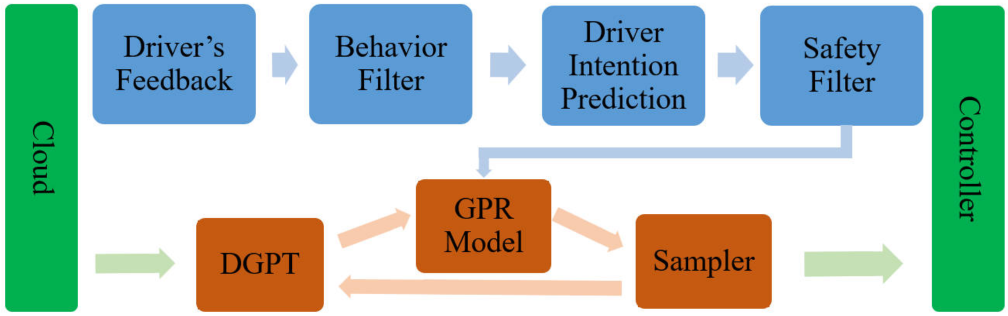

Fig. 3. Flowchart of online DGPT adaptation algorithm.

1) Feedback Data Preprocessing: First, a behavior filter is needed to ensure that only necessary updates are made to the DGPT since drivers' takeover behaviors can be very noisy and may only last for a short duration or have small inputs. Second, we need to infer the preferred steady state that the driver wishes to achieve through a short period of takeover trajectory. Intuitively, this steady state should correspond to the state when the driver stops the takeover. However, based on extensive experiments and observations, we found that this assumption is not accurate. Drivers may anticipate or prolong takeover behavior based on the speed difference with the preceding vehicle, or they may achieve a steady state through multiple takeovers. Therefore, a robust prediction mechanism is needed to determine the steady state that needs to be updated. Finally, the DGPT may fall into an unreasonable range due to emergency situations or driver's mistakes, leading to potential safety hazards. Therefore, a safety space is defined for the DGPT, and a safety filter is applied to ensure that the DGPT remains bounded within this safe space.

As demonstrated, the prepossessing has the potential to be a highly intricate system. In this study, we propose a simplified heuristic algorithm that adheres to the aforementioned workflow in order to validate its effectiveness. The algorithm is shown in **Algorithm 1**. v and g are the current speed and gap; v_f is the current speed of the preceding vehicle; p is the takeover status; p_t is the takeover time; P_T is minimum takeover time; K_T is the coast-down coefficient; V_D is the maximum speed difference; $Safe\_TG\_max$ and $Safe\_TG\_min$ are the safety time gap bounds; v_out and g_out are predicted steady state feedback.

Algorithm 1: Driver's Feedback Preprocessing

```
Data: Input: DGPT, v, v_f, g, p, p_t
Parameters: P_T, K_T, V_D, window_size, Safe_TG_max, Safe_TG_min
Function: moving_average(), max(), min()
Result: v_out, g_out
for each iteration do
    if not p then
        if (v_f − v) ≥ V_D or p_t ≤ P_T then
            update_flag = False
        end
        else
            update_flag = True
        end
        if update_flag then
            if v_f ≥ v then
                g_desire = g
            end
            else
                g_desire = g(K_T · (v − v_f))
            end
            v_out = v
            g_out = max(g_desire, Safe_TG_min)
            g_out = min(g_out, Safe_TG_max)
        end
    end
end
```

2) GPR-based online adaptation algorithm: Gaussian processes extend multivariate Gaussian distributions to infinite dimensionality. They are a form of supervised learning and the training result represents a nonlinear mapping, $f\_GP(z) : R^{dim}(z) \rightarrow R$, such as Equation 4. Here, the dimension of the input vector is 1. The mapping between the input vector z and the function value $f\_GP(z)$ is accomplished by the assumption that $f\_GP(z)$ is a random variable and is jointly Gaussian distributed with z, which is also assumed to be a random variable [26].

The configuration of the GPR model includes selecting the model regressors, the mean function, and the covariance function (i.e., kernel function). In this study, we apply a commonly used zero-mean and the squared-exponential covariance function that relates two sample input vectors z_i and z_j:

$$c\left(\mathbf{z}_i, \mathbf{z}_j\right) = \sigma_j^2 \exp\left(-\frac{1}{2}\left(\mathbf{z}_i - \mathbf{z}_j\right)^T P^{-1}\left(\mathbf{z}_i - \mathbf{z}_j\right)\right) + \sigma_n^2 \delta_{ij} \tag{5}$$

where $\sigma_{ij} = 1$ if $i = j$ and $\sigma_{ij} = 0$ otherwise, and $P = \operatorname{diag}\left[l_1^2, \ldots, l_{\dim(\mathbf{z})}^2\right]$ contains the characteristic length scale for each dimension of the input vector. The hyperparameters of the covariance function $\theta = \left[\sigma_f, \sigma_n, l_1, \ldots, l_{\dim(\mathbf{z})}\right]^T$ include the measurement noise σ_n, the process standard deviation σ_f, and the characteristic length scales, which are learned by maximizing the likelihood of the observation.

Each time a driver takeover occurs, the GPR-based online adaptation module randomly samples points from the preprocessed DGPT along with the driver's feedback. These sampled points are then used to fit a GPR model. The output of the GPR model is a continuous function mapping the relation between speed and desired gap. To update the DGPT, we then uniformly sample the fitted model according to the resolution of the DGPT. Since the GPR fitting is always based on a fixed number of samples, the model's scale does not increase over time. This guarantees the real-time nature of online adaptation.

It is important to note the significant role of random sampling from DGPT throughout the online adaptation module, as it dictates how the scattered driver feedback signal influences the update of the DGPT. More specifically, the random sampling from DGPT needs to follow a certain distribution: the closer the region is to the driver feedback, the lower the probability it gets sampled, while regions further away from the driver feedback have a higher sampling probability. This sampling strategy ensures that during the GPR fitting, regions distant from the driver feedback hold a higher level of confidence, whereas areas close to the driver feedback hold less confidence. Furthermore, the discrete feedback signal can substantially influence its surrounding area.

C. Controller Design

In this study, we used a PID controller to control the acceleration of the ego vehicle and ensure that it follows the preceding vehicle with a desired space gap. The error between the current gap and the desired gap in the DGPT, as defined in Equation 6, is used as the control input. The PID controller continuously calculates the acceleration of the vehicle based on Equation 7, which takes into account the proportional, integral, and derivative components of the error. The controller aims to minimize the error and maintain a stable and safe distance between the ego vehicle and the preceding vehicle. K_p, Ki, and K_d are the coefficients to balance each term. This approach has been widely used in car-following models and has shown good performance in various scenarios.

$$e(t) = DGPT(v(t)) - g(t) \tag{6}$$

$$a(t) = K_p \cdot e(t) + Ki \cdot \int e(t)dt + K_d \cdot \frac{de(t)}{dt} \tag{7}$$

V. Experiments and Results

This section presents the experimental setup using the HuiL simulation platform, the metrics used to evaluate the proposed algorithm's performance, and the results and analysis of the experiments.

A. Experiment Setup using HuiL Simulation

Validation of autonomous driving algorithms often requires consideration of safety factors, making real-world experiments difficult to conduct. Therefore, a good alternative is to use game engine-based simulations with a human-in-the-loop setup. These simulations provide testers with an

Fig. 4. Human-in-the-loop simulator at the University of California, Riverside.

immersive visual and auditory experience and can collect relatively reliable driving data. Game engines are commonly used by software developers to create video games, which typically include a physics engine, rendering engine, and scene graph for managing multiple game elements. In this study, we conducted human-in-the-loop simulations using the game engine-based driving simulator developed in our previous work [27]. The platform is built with the Unity game engine and a Logitech G27 Racing Wheel (see Fig. 4). The simulation environment features a three-lane freeway scenario with varying weather conditions. During the test, the driver could choose to drive the ego vehicle manually or use P-ACC by monitoring the automation and pressing the accelerator or brake pedal when feeling uncomfortable. We describe the experimental setup in detail in the following subsections. In order to test either manual driving or automatic control, the scenarios are determined by the speed profile of the leading vehicle. There are four requirements for generating the speed profile, including avoiding test driver fatigue, collecting demonstration trajectories covering a wide range of speeds for IRL model training, making the speed profile realistic, and ensuring that the scenario is unpredictable. To meet these requirements, we developed a stochastic scenario generation approach. This approach first samples short trajectory segments from naturalistic driving data, then generates a high-level random speed sequence, where each element of the sequence defines an average speed for a period of time. Next, based on the random speed sequence, the trajectory segments with corresponding average speed are randomly selected and concatenated together. Finally, a filter is applied to ensure the acceleration/deceleration of the synthesized speed profile is within the required bounds. As each driving cycle is only 300 seconds long, and a drastic change in speed may lead to abnormal driving behavior, test drivers need to take multiple driving cycles, each covering a different speed range.

We generate five different scenarios covering a range of average speeds from slow (5m/s) to fast (30m/s) using the above speed profile synthesis method. Among them, we use four speed profiles as manual driving scenarios (ACC OFF) to collect naturalistic car-following trajectories from test drivers for training the IRL-based offline personalized DGPT. A total of five test drivers participated in the experiments. After obtaining the personalized DGPT for each driver, we conducted automatic control tests to validate the proposed algorithm. The driving scenarios for the automatic control tests (ACC ON) are divided into two parts: **Scenario A** consists of the speed profile seen by the drivers during the manual driving tests, and **Scenario B** is a completely new scenario. Each scenario lasts 300 seconds.

As a control group, we also introduced a heuristic algorithm to implement online adaptation. Similar to the GPR-based online adaptation module, the input for the heuristic algorithm also needs to be preprocessed as described previously. Subsequently, the heuristic algorithm directly applies a moving average to disperse the scattered driver's feedback signal throughout the entire DGPT. This heuristic method allows for the diffusing of driver feedback across the DGPT, providing an alternative method for online adaptation.

We test six different combinations of controllers: **Predefined**, **Predefined + Online Adaptation (Heuristic)**, **Predefined + Online Adaptation (GPR)**, **IRL**, **IRL + Online Adaptation (Heuristic)**, and **IRL + Online Adaptation (GPR)**. The **Predefined** ACC controller involves the driver choosing a constant time headway from high (4s), medium (3s), and low (1s) levels as the control reference, which is similar to the ACCs currently equipped on commercial vehicles. The **Predefined + Online Adaptation** (both Heuristic and GPR) setup involves the driver choosing a constant time headway as the control reference, but incorporating an online adaptation algorithm to update the control reference table based on real-time feedback, achieving a certain degree of personalization. **IRL** involves using the DGPT trained through offline IRL to control the vehicle, which has been shown in our previous studies to significantly improve drivers' comfort and trust in the system compared to ACC without personalization. Finally, **IRL + Online Adaptation** (both Heuristic and GPR) is the complete proposed framework, which uses the DGPT obtained through offline IRL as the initial control reference to control the vehicle and incorporates an online adaptation algorithm to continuously refine personalization based on real-time feedback.

B. *Results and Analysis*

In this section, we present the results of the experiments and analyze the data. We use Percentage of Interruption (PoI) and Number of Interruption-per-Minute (NIM) to quantitatively measure the driver's comfort and trust in the P-ACC system. PoI denotes the time percentage when the driver steps onto the acceleration pedal or brake pedal, and NIM denotes the number of times the driver steps onto the pedals. As shown in TABLE I, both the PoI and NIM have been greatly reduced when the complete proposed system is applied and compared with the ACC without online adaptation.

Experiments show that the average PoI reduction was 70.9% and the average NIM decreased by 61.7% compared to the predefined ACC settings. This indicates the drivers are more satisfied with automatic car-following based on the proposed P-ACC. This advantage is observed in both seen and unseen driving scenarios, indicating the robustness of both offline and online modules in adapting to new situations.

Notably, the proposed framework, namely, **IRL + Online Adaptation (GPR)**, outperforms other automation control methods in the PoI metric significantly, while the **IRL + Online Adaptation (Heuristic)** has the best NIM performance. It can be asserted that the proposed framework incorporating GPR effectively reduces the proportion of driver interventions, and any presence of online adaptation within the system tends to lower the number of interventions required. When comparing controllers that use online adaptation, those with an initial DGPT obtained through IRL generally exhibit better performance than those using predefined gaps. This indicates that the application of IRL in determining the initial DGPT can enhance the efficacy of the controller, thereby improving the overall performance of the system.

VI. Conclusion and Future Work

In summary, vehicle automation and Advanced Driver Assistance Systems (ADAS) are playing an increasingly important role in enhancing driving safety and comfort. However, pre-defined settings may not always align with individual driver preferences and styles. The emergence of personalized ADAS (P-ACC) aims to solve this problem. While previous research has primarily focused on using historical driving data to create personalized controllers, this study proposes a novel cloud-vehicle collaborative P-ACC framework that includes both offline and online components. By recording the driver's naturalistic car-following trajectories and utilizing IRL in the cloud to train the personalized model (e.g., DGPT), the offline component can obtain the driver's preference before the trip. Then, while the ACC is activated en-route, the online component updates the corresponding DGPT in real time by adapting to the driver's feedback (i.e., takeover of the control). Additionally, with the help of incremental learning achieved through retraining the model based on driver's takeover trajectories, the model gradually becomes more consistent with the driver's driving preferences. Human-in-the-loop (HuiL) simulation experiments demonstrate that this method can significantly reduce driver interventions in the automatic control system, where average PoI has decreased by up to 61.7%, and average NIM has decreased by up to 70.9% for each scenario. This personalized approach can help to ensure a more comfortable experience while also increasing driver trust and usage of this type of ADAS.

In the future, we plan to develop a more sophisticated preprocessing module that can better estimate whether the driver has reached a satisfactory state and can more accurately estimate the preferred state's value. In addition, conducting real-world tests with actual drivers would provide valuable insights into the effectiveness of the proposed framework. Finally, incorporating additional sensor data and considering more complex driving scenarios, such as intersections

TABLE I

RESULTS OF HUMAN-IN-THE-LOOP SIMULATION

		Pred.		Pred. + OL Adapt. (Heru.)		Pred. + OL Adapt. (GPR)		IRL		IRL + OL Adapt. (Heur.)		IRL + OL Adapt. (GPR)	
		PoI	NIM	PoI	NIM	PoI	NIM	PoI	NIM	PoI	NIM	PoI	NIM
Driver 1	Seen	14.1%	6.0	**2.9%**	3.0	11.1%	3.0	17.6%	4.7	4.7%	3.3	5.3%	**1.7**
	Unseen	13.3%	7.0	**7.3%**	4.7	18.4%	4.7	17.1%	5.0	10.6%	**3.0**	14.6%	4.0
Driver 2	Seen	12.0%	5.3	**3.7%**	4.7	12.0%	6.7	4.1%	6.3	9.3%	**4.3**	5.5%	9.0
	Unseen	6.7%	5.7	7.8%	7.0	9.6%	10.3	**5.6%**	5.3	11.2%	**4.7**	8.5%	12.7
Driver 3	Seen	35.4%	29.3	10.9%	15.7	9.8%	4.3	31.4%	11.0	3.2%	3.0	**2.9%**	**1.3**
	Unseen	17.0%	12.7	**3.1%**	6.7	6.8%	4.0	25.9%	13.7	11.3%	9.0	**3.1%**	**3.7**
Driver 4	Seen	26.0%	5.7	10.4%	4.0	9.1%	2.3	3.6%	1.7	**2.6%**	**1.3**	2.8%	1.7
	Unseen	21.9%	5.3	10.0%	3.0	4.5%	2.7	8.7%	2.7	2.4%	**1.3**	**2.3%**	2.3
Driver 5	Seen	29.8%	16.0	13.8%	10.3	**5.4%**	4.3	16.3%	7.3	10.8%	6.3	6.8%	**4.0**
	Unseen	26.6%	14.3	11.6%	4.3	**8.2%**	**3.7**	14.9%	6.0	9.3%	4.3	7.1%	**3.7**
Average		20.3%	10.7	8.2%	6.3	9.5%	4.6	14.5%	6.4	7.5%	**4.1**	**5.9%**	4.3

and merging lanes, will help enhance the performance and adaptability of the P-ACC system.

ACKNOWLEDGMENT

This work is sponsored by the "Digital Twin" project of InfoTech Labs, Toyota Motor North America R&D.

REFERENCES

[1] Toyota. (2021) Toyota Safety Sense: The Standard for Safety. [Online]. Available: https://www.toyota.com/safety-sense/

[2] Volkswagen. (2021) Adaptive Cruise Control. [Online]. Available: https://www.volkswagen-newsroom.com/en/adaptive-cruise-control-acc-3664

[3] Ford. (2021) Adaptive Cruise Control. [Online]. Available: https://www.ford.com/technology/driver-assist-technology/adaptive-cruise-control/

[4] A. Ziebinski, R. Cupek, D. Grzechca, and L. Chruszczyk, "Review of advanced driver assistance systems (adas)," in *AIP Conference Proceedings*, vol. 1906, no. 1. AIP Publishing LLC, 2017, p. 120002.

[5] M. Hasenjäger, M. Heckmann, and H. Wersing, "A survey of personalization for advanced driver assistance systems," *IEEE Transactions on Intelligent Vehicles*, vol. 5, no. 2, pp. 335–344, 2019.

[6] A. Abdelraouf, R. Gupta, and K. Han, "Interaction-aware personalized vehicle trajectory prediction using temporal graph neural networks," *arXiv preprint arXiv:2308.07439*, 2023.

[7] J. M. Wang, D. J. Fleet, and A. Hertzmann, "Gaussian process dynamical models for human motion," *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 2, pp. 283–298, 2007.

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

[9] A. Kesting, M. Treiber, and D. Helbing, "Enhanced intelligent driver model to access the impact of driving strategies on traffic capacity," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 368, no. 1928, pp. 4585–4605, 10 2010.

[10] G. Newell, "A simplified car-following theory: a lower order model," *Transportation Research Part B: Methodological*, vol. 36, no. 3, pp. 195–205, 2002. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0191261500000448

[11] 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.

[12] Z. Zhao, Z. Wang, K. Han, R. Gupta, P. Tiwari, G. Wu, and M. J. Barth, "Personalized car following for autonomous driving with inverse reinforcement learning," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 2891–2897.

[13] H. Gao, G. Shi, G. Xie, and B. Cheng, "Car-following method based on inverse reinforcement learning for autonomous vehicle decision-making," *International Journal of Advanced Robotic Systems*, vol. 15, no. 6, p. 1729881418817162, 2018.

[14] Y. Wang, Z. Wang, K. Han, P. Tiwari, and D. B. Work, "Personalized adaptive cruise control via gaussian process regression," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2021, pp. 1496–1502.

[15] L. Chong, M. M. Abbas, and A. Medina, "Simulation of driver behavior with agent-based back-propagation neural network," *Transportation Research Record*, vol. 2249, no. 1, pp. 44–51, 2011.

[16] X. Huang, J. Sun, and J. Sun, "A car-following model considering asymmetric driving behavior based on long short-term memory neural networks," *Transportation Research Part C: Emerging Technologies*, vol. 95, pp. 346–362, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0968090X1830158X

[17] B. Zhu, Y. Jiang, J. Zhao, R. He, N. Bian, and W. Deng, "Typical-driving-style-oriented personalized adaptive cruise control design based on human driving data," *Transportation research part C: emerging technologies*, vol. 100, pp. 274–288, 2019.

[18] N. Bao, A. Carballo, and T. Kazuya, "Prediction of personalized driving behaviors via driver-adaptive deep generative models," in *2021 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2021, pp. 616–621.

[19] X. Liao, X. Zhao, Z. Wang, Z. Zhao, K. Han, R. Gupta, M. J. Barth, and G. Wu, "Driver digital twin for online prediction of personalized lane change behavior," *IEEE Internet of Things Journal*, 2023.

[20] J. Chen, T. Wu, M. Shi, and W. Jiang, "Porf-ddpg: Learning personalized autonomous driving behavior with progressively optimized reward function," *Sensors*, vol. 20, no. 19, p. 5626, 2020.

[21] Y. Zhou, R. Fu, C. Wang, and R. Zhang, "Modeling car-following behaviors and driving styles with generative adversarial imitation learning," *Sensors*, vol. 20, no. 18, 2020.

[22] C. Huang, H. Huang, P. Hang, H. Gao, J. Wu, Z. Huang, and C. Lv, "Personalized trajectory planning and control of lane-change maneuvers for autonomous driving," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 6, pp. 5511–5523, 2021.

[23] X. Liao, S. Mehrotra, S. Ho, Y. Gorospe, X. Wu, and T. Mistu, "Driver profile modeling based on driving style, personality traits, and mood states," in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2022, pp. 709–716.

[24] A. Abdelraouf, M. Abdel-Aty, and Y. Wu, "Using vision transformers for spatial-context-aware rain and road surface condition detection on freeways," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 10, pp. 18 546–18 556, 2022.

[25] Y. Ma, W. Ye, X. Cao, A. Abdelraouf, K. Han, R. Gupta, and Z. Wang, "Cemformer: Learning to predict driver intentions from in-cabin and external cameras via spatial-temporal transformers," *arXiv preprint arXiv:2305.07840*, 2023.

[26] C. E. Rasmussen, "Gaussian processes in machine learning," in *Advanced Lectures on Machine Learning: ML Summer Schools 2003, Canberra, Australia, February 2-14, 2003, Tübingen, Germany, August 4-16, 2003, Revised Lectures*. Springer, 2004, pp. 63–71.

[27] Z. Wang, K. Han, and P. Tiwari, "Digital twin simulation of connected and automated vehicles with the unity game engine," in *2021 IEEE 1st International Conference on Digital Twins and Parallel Intelligence (DTPI)*. IEEE, 2021, pp. 1–4.