Deploying Traffic Smoothing Cruise Controllers Learned from Trajectory Data

Nathan Lichtlé[1†], Eugene Vinitsky[2†], Matthew Nice[3†], Benjamin Seibold[6], Dan Work[3] and Alexandre M. Bayen[4,5]

***Abstract*— Autonomous vehicle-based traffic smoothing controllers are often not transferred to real-world use due to challenges in calibrating many-agent traffic simulators. We show a pipeline to sidestep such calibration issues by collecting trajectory data and learning controllers directly from trajectory data that are then deployed zero-shot onto the highway. We construct a dataset of 772.3 kilometers of recorded drives on the I-24. We then construct a simple simulator using the recorded drives as the lead vehicle in front of a simulated platoon consisting of one autonomous vehicle and five human followers. Using policy-gradient methods with an asymmetric critic to learn the controller, we show that we are able to improve average MPG by 11% in simulation on congested trajectories. We deploy this controller to a mixed platoon of 4 autonomous Toyota RAV-4's and 7 human drivers in a validation experiment and demonstrate that the expected time-gap of the controller is maintained in the real world test. Finally, we release the driving dataset [1], the simulator, and the trained controller at https://github.com/nathanlct/trajectory-training-icra.**

I. Introduction

The increased availability of automated lane and distance keeping in modern vehicles has rapidly transitioned our roadways into the mixed autonomy regime where autonomous and human drivers all operate together. With the availability of autonomous vehicles (AVs) as mobile traffic actuators, it is now possible to perform Lagrangian traffic control in which control of the highway is dispersed amongst many vehicles in the flow. The ability to perform distributed control has brought closer the long-standing goal of AV research [2]–[6]: to use the programmability and fast reaction time of AVs to improve socially desirable highway metrics like congestion and energy efficiency for both humans and AVs.

Eugene Vinitsky is a recipient of an NSF Graduate Research Fellowship and funded by the National Science Foundation under Grant Number CNS-1837244. Computational resources for this work were provided by the Savio cluster at Berkeley. This material is also based upon work supported by the U.S. Department of Energy's Office of Energy Efficiency and Renewable Energy (EERE) award number CID DE-EE0008872. The views expressed herein do not necessarily represent the views of the U.S. Department of Energy or the United States Government. We would like to thank the International Emerging Actions project SHYSTRA (CNRS). Thanks to Gracie Gumm and Michael Roman for contributions to data collection.

[1] Department of Computer Science, ENS Paris-Saclay, Paris-Saclay University

[2] Department of Mechanical Engineering, UC Berkeley

[3] Department of Civil and Environmental Engineering, Vanderbilt University

[4] Department of Electrical Engineering and Computer Science at UC Berkeley

[5] Institute for Transportation Studies, UC Berkeley

[6] Departments of Mathematics and Physics, Temple University

† **These authors contributed equally to this work.**

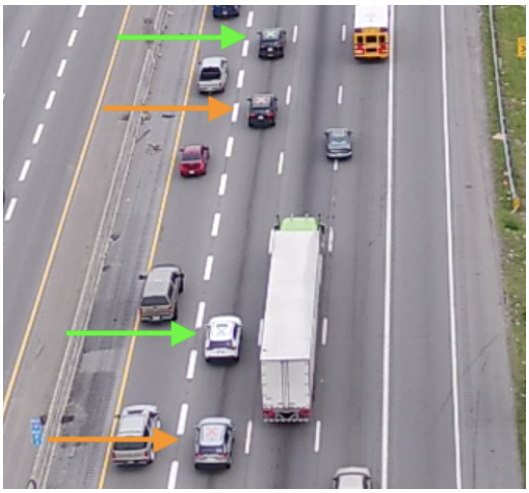

Fig. 1: 4 of 11 vehicles in formation on the roadway. Green arrows and green X on roof indicate AV (AV), orange arrows and orange X on roof indicate human driven sensing vehicle (H). During experiments platoon formed in this order: [H, H, AV, H, AV, H, AV, H, AV, H, H], with no control over traffic flow consistently cutting in and out.

In particular, prior work [7] has shown that even at low penetration rates of less than 4%, empirical and theoretical evidence suggests that AVs can significantly reduce stop-and-go traffic, a pernicious transitory phenomenon in which vehicles alternate between starting and stopping, consuming extra fuel in the process. However, prior approaches have a unifying problem: they are developed and analyzed in simplistic settings such as rings or hand-designed input perturbations. Testing on more complex settings is difficult as: 1) real-world highway sensor data are sparse and lack required resolution and detail needed for accurate modeling; 2) developing simulators that properly reproduce emergent structures from many-vehicle-interactions is challenging.

Building more complex models is heavily data constrained. Loop detectors only yield macroscopic statistics, while cameras tend to cover only a small portion of the roadway. This lack of available data is a fundamental issue as the trajectories of vehicles traveling through waves depends on the wave speed [8], and yet the wave speed is difficult to estimate with available stationary sensors. However, without an accurate means of reconstructing the stop-and-go traffic that is likely to occur on a particular highway, it is difficult to validate how a controller will perform when deployed on

that highway. Consequently, it is unclear whether progress on control design for real-world smoothing is being made.

The contribution of this paper is a pipeline which avoids these aforementioned modeling challenges and produces a reinforcement learning (RL) controller that is then successfully deployed on four vehicles in dense highway traffic. This pipeline has three parts: (i) the data collected from human driving trajectories, (ii) the RL controller, and (iii) the deployment of the controller on physical vehicles.

We are able to avoid modeling challenges by learning a traffic-smoothing controller directly from data collected from human driving trajectories. Instead of attempting to build a high fidelity simulation, we evaluate and train our controllers on collected highway trajectory data, ensuring that our controllers are learning to smooth a realistic representation of waves from the particular highway on which we intend to deploy AVs. We construct a simplified controller evaluation procedure in which a simulated mixed platoon of AVs and human drivers follows directly behind trajectories collected by a human driver on I-24, an interstate highway in Tennessee, scoring the controllers by their ability to improve energy consumption while maintaining traffic throughput. This approach sidesteps the aforementioned difficulties in calibrating both the waves and the microscopic car following dynamics. Using Proximal Policy Optimization [9], an RL policy gradient algorithm, we learn a controller that decreases the fuel consumption of the platoon in simulation by 16% for the AV and 10% on average for the platoon vehicles. Finally, we deploy the controller on real vehicles in highway traffic, showing the viability of this controller to create real-world energy savings and use of the complete pipeline.

The rest of this paper is organized as follows: in Section II we discuss works informing this research, in Section III we discuss the data collection, cleaning, and analysis, in Section IV we discuss the controller design and structure, algorithm, training details, and deployment pipeline, in Section V we discuss the simulation results, and experimental results, and finally in Section VI we discuss and provide practical considerations to be considered in future work.

II. Related Work

Prior work has investigated the efficacy of traffic smoothing controllers on settings such as rings or hand-designed input perturbations. The most closely related works are [7], [10], [11]. In [7], the authors showed that a single AV could be used to dampen stop-and-go waves on a ring with 21 human drivers, yielding sharply improved fuel efficiency. The work in [10] studies traffic smoothing with connected AVs and demonstrates that the connectivity can be used for more effective dampening of waves on a single-lane, eight-mile-long public road. The work in [11] conducted an experiment in which individual vehicle speeds were controlled to smooth traffic flow. Other works have considered the wave dampening properties of existing commercially-available cruise controllers, with [12], [13], [14] all observing that the vehicles they tested were string unstable. Finally, [15] has studied some of the sim-to-real challenges in deploying RL-learned cruise controllers into more realistic settings.

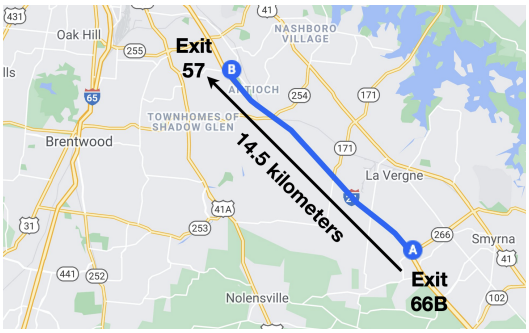

Fig. 2: Portion of the I-24 highway on which we collected most of the dataset (section III-A) and where we ran the experiments described in section V.

Prior work has also studied the use of reinforcement learning (RL) and optimal control for developing micro-level controllers that optimize mixed autonomy traffic. [16] learns memory-based policies that infer ring densities and consequently outputs near-optimal policies for the ring, [17] uses multi-agent reinforcement learning (MARL) to optimize the throughput of a merging region, and [18] employs MARL to investigate the potential impacts of altruistic autonomous driving on a merge scenario. At a network level, RL has been used to learn routing behaviors for AVs that induce the human drivers to select paths that lead to decreased congestion [19].

III. Training Set

Here we detail the human driver data collection procedure. The data serve as the basis for which we train wave smoothing controllers. We then briefly describe the data cleaning process and analyze the distribution of trajectories collected.

A. Data Collection

We collect data by recording trajectory data on a 14.5-kilometer-long segment (displayed in Fig. 2) of I-24 located southeast of Nashville, Tennessee. Each drive is conducted in an instrumented vehicle that logs CAN data via libpanda [20] and GPS data from an onboard receiver. Collected measurements from the vehicle CAN data include the velocity of the *ego vehicle* (the vehicle being driven), the relative velocity of the *lead vehicle* (the vehicle in front of the ego vehicle), the instantaneous acceleration, and the *space-gap* (bumper-to-bumper distance).

The drives are varied in the time of day, day of the week, direction of travel on the highway, and level of congestion. Each drive is made up of one or more passes through the highway stretch of interest. The data used to train the algorithm in this work are made publicly available at [1], along with more details on the data.

B. Data Cleaning

The raw data for a given drive were recorded in two files: a CAN data file and a GPS file. The pertinent data are pulled

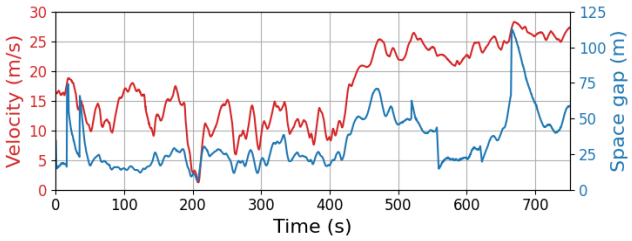

Fig. 3: Velocity of the ego vehicle (blue) and space-gap to the lead vehicle (red) for a single trajectory in the dataset, containing sharp variations in both velocity and space-gap.

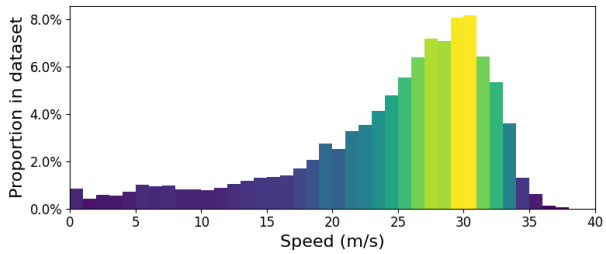

Fig. 4: Histogram showing the distribution of velocities of the ego vehicle in the dataset.

from the CAN data and interpolated to the GPS time, which is measured at 10 Hz. High-frequency CAN data are down-sampled and linearly interpolated to match the GPS time, and low-frequency CAN data undergo linear interpolation to match the 10 Hz GPS time as well. Distance traveled and direction of travel are computed using the GPS position data. Since the westbound data contain more regular congestion, we focus on westbound data for training. The westbound data contain 60 trajectories, representing 8.8 hours and 772.3 kilometers of driving.

C. Dataset Analysis

The data are collected over a wide range of traffic conditions ranging from congested traffic that is nearly stopped to free-flow, max-speed traffic, including many acceleration and deceleration patterns corresponding to stop-and-go traffic. Fig. 3 shows an example velocity and space-gap profile from a trajectory in the dataset, where we can observe the ego vehicle going quite rapidly from low to high speeds. While our main interest is in smoothing high-frequency waves, which occur primarily in congestion, the distribution of speeds in the training dataset, shown in Fig. 4, tends towards higher speeds. While we could filter the dataset to only contain low speeds, likely making the learning problem simpler, Fig. 3 suggests that regions of congestion are often quickly followed by regions of high speed. To ensure our controller behaves appropriately at high speeds and in transitions between high and low speed regions, we keep both low and high velocities in the training dataset.

Fig. 5: Vehicle formation used in simulation. A trajectory leader (in green) driving a speed profile drawn from the dataset is placed in front of an AV (in red) which is followed by a platoon of 5 human vehicles (in white), modeled using the Intelligent Driver Model.

D. Constructing the Training Environment

In order to use the collected data, we build a one-lane training environment where the AV follows behind the trajectory collected from the human drivers. The human driver is placed at the front of a simulated platoon, followed by the AV, followed by five vehicles driving according to the Intelligent Driver Model (IDM) [21] with a set of parameters that are string unstable below $18\frac{\text{m}}{\text{s}}$, which ensures that the waves grow in congestion. Although having a full micro-simulation of the I-24 would allow for training on a model with complex long-range interactions between the vehicles, the simulator proposed here allows us to train on realistic driving dynamics that are representative of both the types of waves on this highway and how drivers react to wave formation. As an additional benefit, this single-lane simulation using half a dozen vehicles achieved 2000 steps per-second while a comparable micro-simulation of the full 14 kilometer road section would have thousands of vehicles in congestion and would be very computationally costly to evaluate.

Our collected dataset contains both the trajectory of our drivers and the vehicles in front of them (via space-gap and relative velocity data logged on the CAN). We discard the lead trajectories and do not use them for simulation as the lead trajectories contain both cut-ins (a vehicle cuts in between the lead vehicle and the ego driver) and cut-outs (the lead vehicle changes lanes). While cut-outs are likely unaffected by the behavior of the ego driver, cut-ins are likely a function of the spacing between ego driver and lead vehicle. Since our trained controller will have different space-gap keeping patterns, it is possible that the observed cut-outs would not occur given the controller's choice of space-gaps; to avoid dealing with this counterfactual we simply do not use the leader data for training and only keep ego vehicle data for our lead trajectories. Since the human drivers who collected the dataset intentionally rarely change lanes, our simulator consequently does not contain lead-vehicle lane changes. Finally, we note that we do not split the data into a train and test set; we train our controller on all of the available trajectories and instead use the deployment as our test set.

IV. Method

In this section we describe the control design and structure, the details on how the controller is trained on the trajectory

data, and the deployment pipeline that enables experiments to be conducted on a real vehicle platform.

A. Controller Design

For the model of the system dynamics, controls, and inputs we adopt the following system. As it is unclear whether the Markov property holds for this system [22], we will assume that the system described below may be slightly non-Markovian.

State space $[v, v_{\text{lead}}, h]$ where v is the AV speed, v_{lead} the speed of the vehicle right in front of it, and h the space-gap. All of these features can be acquired by using the forward-facing radar and the data collection software [20], [23] that we place on our vehicles.

Action space an instantaneous acceleration a, bounded between $[-4.5, 2.6]$ $\frac{\text{m}}{\text{s}^2}$, to be applied to the AV. Note that we do not allow the AVs to lane-change in this work.

Reward function the reward the AV receives at time-step t is a combination of minimizing energy consumption, acceleration regularization and penalties for leaving too small or too large gaps. It is given by

$$r_t = 1 - c_0 E_t - c_1 a_t^2 - c_2 P_t \ .$$

Here E_t is the instantaneous gallons of fuel consumed by the AV (given by a piece-wise polynomial energy model calibrated to a RAV-4 Toyota vehicle; the fitting procedure and function coefficients are given in [24]), a_t the AV's instantaneous acceleration in $\frac{m}{s^2}$ and P_t its gap penalty, all at time-step t. The first term is intended to discourage fuel consumption, the second to encourage smooth driving, and the third to discourage the formation of large gaps that induce cut-ins or small gaps that might lead to driver discomfort. For our reward functions, we use coefficients $c_0 = 1.0\frac{1}{\text{Gal}}$, $c_1 = 0.002\frac{s^2}{m}$ and $c_2 = 2$, and penalize with $P_t = 1$ when the gap is below 7m, above 120m or when the time-gap (i.e., space-gap over speed) to the leader is below 1 second. These particular values were selected via an informal hyperparameter search and found to yield improved fuel consumption of the platoon while maintaining all constraints that might set off the penalty term P_t.

Finally, we note that our reward function does not include the energy consumption of the following platoon. While we experiment with such a reward, we observed more improvement by only optimizing for the energy consumption of the AV.

B. Controller Structure

The RL controller $G(\cdot)$ takes as inputs the current vehicle speed, the speed of the lead vehicle, and the space-gap provided by recorded or real-time CAN-to-ROS translation [23], and outputs a desired acceleration to a supervisory FollowerStopper [25] wrapper controller. The FollowerStopper leverages reachability analysis to verify safety and allows for total avoidance of a collision with the lead vehicle by taking in a desired velocity and returning a safe commanded velocity v_{safe}. The controller output during learning is acceleration-based; to convert it into a desired velocity we return $v_{\text{des}} = v_t + 0.6 \cdot G(\cdot)$ where v_{des} is the speed passed to the FollowerStopper and $G(\cdot)$ is the acceleration output by the RL controller. This desired velocity is then sent via CAN [20] to the vehicle's ECU for actuation. The particular "integration constant" 0.6 corresponds to the vehicle's responsiveness of about $\tau = 0.6s$, which is found by making the mapping from desired speed to realized speed as close as possible to the identity function, a mapping that we get from a transfer function approximating the vehicle's dynamics.

C. Algorithm

We train our policy using Independent Proximal Policy Optimization [9] (PPO), a policy gradient algorithm. We modify the standard PPO algorithm by providing the value function with a few additional inputs: the total distance traveled from start to time t, the total energy consumed by the agent at time t, and time t. The value function V^π estimates the discounted cumulative reward from a given state s_t and a particularly controller π. This quantity is difficult to estimate without the additional information we provide due to the partially observed state described in Sec. IV-A. The non-local information provided to the value function is used exclusively during training for variance reduction (see [9] for details), and these additional inputs are neither available nor needed by the controller during evaluation.

Training was done using the PPO implementation provided in Stable Baselines 3 [26] version 1.0, a Pytorch-based deep RL library. Training details and hyperparameters are provided in the linked code-base.

D. Deployment Pipeline

An initial software-in-the-loop (SWIL) step is taken to check functional correctness and interface testing in a 2-vehicle Gazebo simulation [27]. Structured velocity profiles (e.g., constant acceleration, sinusoidal, trapezoidal) are input to the controller to ensure outputs are not unusual. This also checks for software correctness. For hardware-in-the-loop (HWIL) deployment on the physical vehicle, there is a series of three tests to mitigate safety risks from the transition from simulation to physical vehicle before testing the controller on the I-24 segment. All three tests have varied input from the leader vehicle to ensure performance in non-equilibrium states.

First, the controller is tested in a 'Ghost Mode' as in [28] where the vehicle follows a simulated 'Ghost' vehicle as its leader. This provides the opportunity for a bad implementation to fail and crash into a virtual vehicle instead of a real one. The full HWIL setup is used with the modification that the real sensing done by the vehicle is replaced by a spoofed recording of a lead vehicle ahead using [23]. Second, the controller is tested in a 'CAN Coach Mode' as in [28] where the controller feedback is sent through a human-in-the-loop (HIL) for actuation. This second test occurs on a low-traffic, high speed route. Here the vehicle sensors feed real-time data into the controller, and the controller gives feedback to the

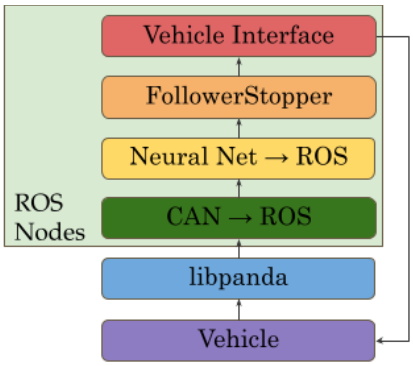

Fig. 6: Diagram showing how information flows though the HWIL system when deployed. The vehicle sensors send data on the CAN bus. Libpanda [20] records the data and data are translated into ROS [23]. The neural net is embedded in a ROS node subscribing to pertinent data, and its output is filtered through a supervisory safety controller to get v_{safe}. This value is sent to the vehicle interface which takes a desired ROS command and sends it via CAN to the vehicle.

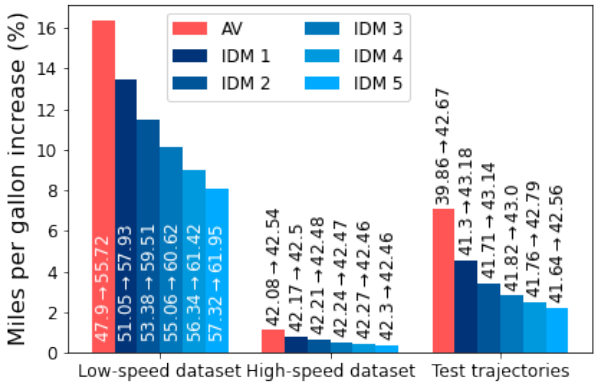

Fig. 7: Percent improvement in MPG relative to a baseline in an IDM vehicle leads the platoon in Fig. 5. Each column contains both percent improvement on the y-axis and MPG values used to compute this improvement inside each column with IDM (AV) on the left (right) of the arrow. High and low speed columns are over the training set. The "Test trajectories" column is the controller evaluated on data from the physical test.

HIL to indicate what input should be provided to the vehicle, but if the controller provides unsafe input to the HIL it is rejected to maintain safety and replaced with human control.

Finally, the controller is used on a low-traffic, high-speed route testing the complete HWIL control loop. Once these are successfully finished, the controller is ready to be tested on the heavy traffic, high speed I-24 roadway segment.

V. Results

A. Simulation Results

Here we analyze the performance of the controller in terms of energy efficiency improvements in miles per gallon (MPG) observed in our simulator. In Fig. 7, we compare the energy consumption of the AV and all vehicles in the platoon (as shown in Fig. 5) when the AV is using our RL controller compared to an IDM controller, over the whole training dataset. We split the trajectories by leader speed, computing the energy savings at leader speeds above and below $18\frac{\text{m}}{\text{s}}$, which is the speed boundary beyond which IDM vehicles with the parameters used in this work go from being string-unstable to string-stable. The results in the left and middle columns indicate that most of the expected energy improvements from the controller will come at low speeds. While these savings are significant, in more complex settings imperfections in actuation, modeling of human drivers, and cut-ins would likely lower the actual improvement. The rightmost column is described in Sec. V-B.

B. Experimental Results

In this section, we describe the validation experiment conducted on the segment of I-24 shown in Fig. 2. We assess the success of the controller deployment onto AVs by showing an accurate match between simulation and reality. Finally, we seek to determine whether our controller improved the energy efficiency of its platoon.

Fig. 1 shows four vehicles from the eleven-vehicle platoon of alternating humans and AVs that we deployed on I-24. For each test, we got the platoon onto the highway without any non-platoon vehicles lane-changing into it. Once on the highway, non-platoon traffic cut in and out of our platoon. Since our vehicles were only instrumented to sense the vehicle in front of them, the number of vehicles that managed to enter into our platoon is unknown. We ran experiments on August 2nd, 4th, and 6th of 2021, each day launching the platoon of vehicles three times and bringing the vehicles back to the start of the highway section in between each run. The controller presented here was only actuated on 08/06, over three tests that occurred at 6:45, 7:29 and 8:36 AM. Fig. 8 shows individual vehicle trajectories on a time-space diagram from the 6:45 AM test; the two regions of red correspond to congestion events. The deployment of the controller from simulation to real vehicles was overall successful as all tests ran safely and smoothly.

We investigate the effect of the sim-to-real gap induced by the presence of cut-ins and cut-outs, which we did not have when training our controller, as well as imperfect modeling of the transfer function of the AV. First, we attempt to compute a counterfactual baseline in which we replay our controller in simulation behind a trajectory collected during the tests. This mechanism is imperfect as the real-world trajectory has cut-ins and replaying a different controller behind it might affect the cut-in frequency. Without a model of lane changing, we cannot perform this counterfactual perfectly so instead we make the calculations assuming that both the times when cut-ins occur and the space-gap directly after the cut-in are unchanged. Occasionally, we choose to relax this latter condition in order not to experience, in simulation, cut-ins that would be more aggressive than what

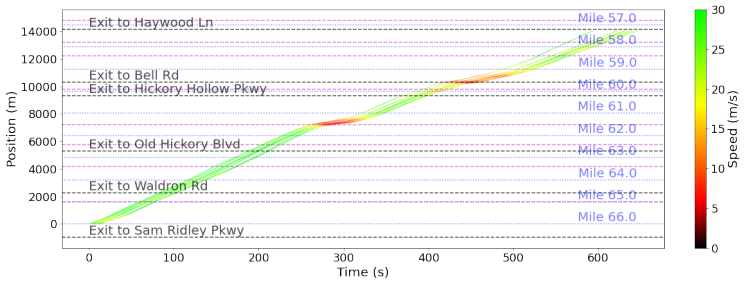

Fig. 8: Time-space diagram showing the trajectories of our platoon of vehicle during the first test. We can observe two low-speed regions of congestion where vehicles following behind the AV could experience wave smoothing.

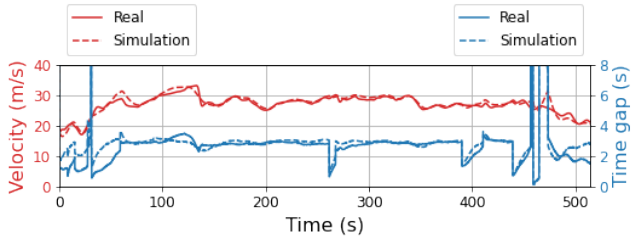

Fig. 9: Comparison of velocity and time-gap between a real (solid) and simulated (dashed) roll-out. There are small divergences that occur around the cut-ins but the car mostly maintains a three-second time-gap in both cases.

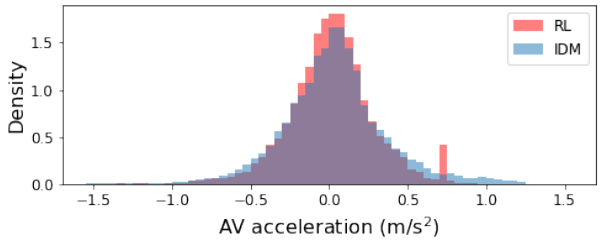

Fig. 10: The density of the AV acceleration when simulating an AV or an IDM vehicle behind leader trajectories from the tests. The AV case places less mass at high, energy-consuming accelerations. The peak observed at 0.75 corresponds to the lead vehicle being out of range due to cut-outs.

the real-world AV experienced. To that end, at each time-step t where a cut-in would leave the AV with a space-gap h_t^{sim} while the real-world AV experienced a space-gap h_t^{real}, we set $h_t^{\text{sim}} = \max(h_t^{\text{sim}}, \min(h_{t-1}^{\text{sim}}, h_t^{\text{real}}))$.

Fig. 9 shows the velocity and time-gap (space-gap divided by velocity) of an AV from the validation experiment as well as the replay of the trajectory in our simulation using the counterfactual cut-in mechanism mentioned above. The velocity profile of the vehicle closely matches its expected behavior computed in simulation. Although there are mismatches around cut-ins and cut-outs (regions where time-gap changes discontinuously), the time-gaps are relatively close and we can observe the vehicle roughly tracking a three-second time-gap in both cases. We observe similar results on the other trajectories we collected during the tests.

Finally, we analyze the potential fuel efficiency improvements from the validation experiment. The third column in Fig. 7 depicts the energy savings obtained when replaying in simulation using the trajectories collected during the experiments using the counterfactual cut-in mechanism mentioned earlier. We observe that the fuel efficiency of the AV has improved by 8% with additional small gains for the IDM vehicles. Fig. 10 shows the density of accelerations taken by the IDM vs. the AV; the higher density of large accelerations of the IDM vehicle are likely the reason for the improved fuel efficiency of the RL AV over the IDM AV. Unfortunately, the day of the deployment featured limited congestion so potential improvements are smaller than might be observed in heavier traffic conditions. More experimental testing on a number of days are needed to provide conclusive experimental energy savings results.

VI. Conclusions and Future Work

In this work we propose and test a pipeline that allows for effective validation and training of traffic smoothing controllers. We collect over 700 km of training data that is used to build a controller validation system. This system avoids the fundamental modeling issues that have restricted the learning or design of traffic smoothing controllers to relatively simple settings, or prevented them from deployment on real cars. In our validation system, we use Policy Gradient methods to train a controller that improves the MPG of an AV by 16% and has benefits for the following human vehicles. We then construct a pipeline for porting these controllers to four AVs and perform physical validation experiments over three days. The behavior of the vehicle on the validation experiment closely matches its expected simulation behavior, suggesting that our pipeline is an effective mechanism for validating controllers.

There are a few missing features in our environment that merit further work. First, our simulator lacks counterfactual lane-changes. In future work, this can be addressed using the observed lane changes in the data to build a single-lane lane changing model that can be used to extend our simulation. In terms of the Markov Decision Process we design, our controller is memory-free, which may prevent the agent from learning a predictive model of downstream speeds that can be used for further smoothing. Additionally, we do not penalize the energy consumption of the platoon; the addition of this penalty may lead to qualitatively different behavior. Finally, additional field experiments can support the assessment of our approaches in a range of traffic congestion levels.

References

[1] Nice, M., Lichtle, N., Gumm, G., Roman, M., Vinitsky, E., Elmadani, S., and Bunting, M., Bhadani, R., Gunter,G., Kumar, M., and McQuade, S., Denaro, C., Delorenzo, R., Piccoli, B., Work, D. Bayen, A. Lee, J., Sprinkle, J. and Seibold, B., "The I-24 trajectory dataset," doi.org/10.5281/zenodo.6366761, 2021.

[2] R. Rajamani, *Vehicle dynamics and control*. Springer Science & Business Media, 2011.

[3] P. Ioannou, Z. Xu, S. Eckert, D. Clemons, and T. Sieja, "Intelligent cruise control: theory and experiment," in *Proceedings of 32nd IEEE Conference on Decision and Control*, Dec 1993, pp. 1885–1890 vol.2.

[4] B. Besselink and K. H. Johansson, "String stability and a delay-based spacing policy for vehicle platoons subject to disturbances," *IEEE Transactions on Automatic Control*, vol. 62, no. 9, pp. 4376–4391, Sep. 2017.

[5] C.-Y. Liang and H. Peng, "Optimal adaptive cruise control with guaranteed string stability," *Vehicle System Dynamics*, vol. 32, no. 4-5, pp. 313–330, 1999. [Online]. Available: https://www.tandfonline.com/doi/abs/10.1076/vesd.32.4.313.2083

[6] D. Swaroop and J. Hedrick, "String stability of interconnected systems," *IEEE Transactions on Automatic Control*, vol. 41, no. 3, pp. 349–357, 1996.

[7] R. E. Stern, S. Cui, M. L. Delle Monache, R. Bhadani, M. Bunting, M. Churchill, N. Hamilton, H. Pohlmann, F. Wu, B. Piccoli *et al.*, "Dissipation of stop-and-go waves via control of autonomous vehicles: Field experiments," *Transportation Research Part C: Emerging Technologies*, vol. 89, pp. 205–221, 2018.

[8] M. R. Flynn, A. R. Kasimov, J.-C. Nave, R. R. Rosales, and B. Seibold, "Self-sustained nonlinear waves in traffic flow," *Physical Review E*, vol. 79, no. 5, p. 056113, 2009.

[9] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[10] I. G. Jin, S. S. Avedisov, C. R. He, W. B. Qin, M. Sadeghpour, and G. Orosz, "Experimental validation of connected automated vehicle design among human-driven vehicles," *Transportation research part C: emerging technologies*, vol. 91, pp. 335–352, 2018.

[11] J. Ma, X. Li, S. Shladover, H. A. Rakha, X.-Y. Lu, R. Jagannathan, and D. J. Dailey, "Freeway speed harmonization," *IEEE Transactions on Intelligent Vehicles*, vol. 1, no. 1, pp. 78–89, 2016.

[12] M. Makridis, K. Mattas, B. Ciuffo, F. Re, A. Kriston, F. Minarini, and G. Rognelund, "Empirical study on the properties of adaptive cruise control systems and their impact on traffic flow and string stability," *Transportation research record*, vol. 2674, no. 4, pp. 471–484, 2020.

[13] V. L. Knoop, M. Wang, I. Wilmink, D. M. Hoedemaeker, M. Maaskant, and E.-J. Van der Meer, "Platoon of sae level-2 automated vehicles on public roads: Setup, traffic interactions, and stability," *Transportation Research Record*, vol. 2673, no. 9, pp. 311–322, 2019.

[14] G. Gunter, D. Gloudemans, R. E. Stern, S. McQuade, R. Bhadani, M. Bunting, M. L. Delle Monache, R. Lysecky, B. Seibold, J. Sprinkle *et al.*, "Are commercially implemented adaptive cruise control systems string stable?" *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[15] K. Jang, E. Vinitsky, B. Chalaki, B. Remer, L. Beaver, A. A. Malikopoulos, and A. Bayen, "Simulation to scaled city: zero-shot policy transfer for traffic control via autonomous vehicles," in *Proceedings of the 10th ACM/IEEE International Conference on Cyber-Physical Systems*, 2019, pp. 291–300.

[16] C. Wu, A. R. Kreidieh, K. Parvate, E. Vinitsky, and A. M. Bayen, "Flow: A modular learning framework for mixed autonomy traffic," *IEEE Transactions on Robotics*, 2021.

[17] J. Cui, W. Macke, H. Yedidsion, A. Goyal, D. Urielli, and P. Stone, "Scalable multiagent driving policies for reducing traffic congestion," *arXiv preprint arXiv:2103.00058*, 2021.

[18] B. Toghi, R. Valiente, D. Sadigh, R. Pedarsani, and Y. P. Fallah, "Altruistic maneuver planning for cooperative autonomous vehicles using multi-agent advantage actor-critic," *arXiv preprint arXiv:2107.05664*, 2021.

[19] D. A. Lazar, E. Bıyık, D. Sadigh, and R. Pedarsani, "Learning how to dynamically route autonomous vehicles on shared roads," *Transportation Research Part C: Emerging Technologies*, vol. 130, p. 103258, 2021.

[20] M. Bunting, R. Bhadani, and J. Sprinkle, "Libpanda: A high performance library for vehicle data collection," in *Proceedings of the Workshop on Data-Driven and Intelligent Cyber-Physical Systems*, 2021, pp. 32–40.

[21] 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, 2010.

[22] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[23] S. Elmadani, M. Nice, M. Bunting, J. Sprinkle, and R. Bhadani, "From CAN to ROS: A monitoring and data recording bridge," in *Proceedings of the Workshop on Data-Driven and Intelligent Cyber-Physical Systems*, 2021, pp. 17–21.

[24] J. W. Lee, G. Gunter, R. Ramadan, S. Almatrudi, P. Arnold, J. Aquino, W. Barbour, R. Bhadani, J. Carpio, F.-C. Chou, M. Gibson, X. Gong, A. Hayat, N. Khoudari, A. R. Kreidieh, M. Kumar, N. Lichtlé, S. McQuade, B. Nguyen, M. Ross, S. Truong, E. Vinitsky, Y. Zhao, J. Sprinkle, B. Piccoli, A. M. Bayen, D. B. Work, and B. Seibold, "Integrated framework of dynamics, instabilities, energy models, and sparse flow controllers," in *Proceedings of the Workshop on CPS Data for Transportation and Smart cities with Human-in-the-loop*, 2021.

[25] F.-C. Chou, M. Gibson, R. Bhadani, A. Bayen, and J. Sprinkle, "Reachability analysis for followerstopper: Safety analysis and experimental results," in *Proceedings of 2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[26] A. Raffin, A. Hill, M. Ernestus, A. Gleave, A. Kanervisto, and N. Dormann, "Stable baselines3," https://github.com/DLR-RM/stable-baselines3, 2019.

[27] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3. IEEE, 2004, pp. 2149–2154.

[28] M. Nice, S. Elmadani, R. Bhadani, M. Bunting, J. Sprinkle, and D. Work, "CAN coach: vehicular control through human cyber-physical systems," in *Proceedings of the ACM/IEEE 12th International Conference on Cyber-Physical Systems*, 2021, pp. 132–142.