IEEE Open Journal of
Intelligent Transportation
Systems

Received 13 March 2022; revised 25 September 2022 and 8 December 2022; accepted 3 January 2023. Date of publication 18 January 2023;
date of current version 10 February 2023.

Digital Object Identifier 10.1109/OJITS.2023.3237977

Learning Policies for Automated Racing Using Vehicle Model Gradients

NATHAN A. SPIELBERG[1], MAXIMILIAN TEMPLER[2], JOHN SUBOSITS[1], AND J. CHRISTIAN GERDES[1] (Member, IEEE)

[1]Department of Mechanical Engineering, Stanford University, Stanford, CA 94305, USA

[2]Volkswagen Group Innovation Driving Dynamics, Volkswagen AG, 38440 Wolfsburg, Germany

CORRESPONDING AUTHOR: N. A. SPIELBERG (e-mail: nspielbe@stanford.edu)

This work was supported by the National Science Foundation Graduate Research Fellowship Program under Grant DGE–1147470.
The work of Nathan A. Spielberg was supported by the NSF Graduate Research Fellowship and William R. and Sara Hart Kimball Stanford Graduate Fellowship.

ABSTRACT Safe autonomous driving approaches should be capable of quickly and efficiently learning as professional drivers do, while also using all of the available road-tire friction for safety. Inspired by how skilled drivers learn, we demonstrate improvement from an initial optimization-generated racing trajectory using model-based reinforcement learning. By using a simple physics-based dynamics model and gradients of the performance objective, we show that a full-scale automated race car is capable of improving lap time in experiments on high- and low-friction race tracks. Using recorded vehicle data, this approach improves a twenty nine second lap time by almost two full seconds. Beyond improving upon the initial optimization-based solution, it uses only two laps worth of ice track data where conditions can constantly change from lap-to-lap. These results suggest that by combining an approximate model with simple learning techniques, significant improvement to automated racing strategies is possible.

INDEX TERMS Racing, iterative learning, automated driving, reinforcement learning.

I. INTRODUCTION

SOME of the most challenging motion control problems involve operating a system near the absolute limits of its physical capabilities. One approach to such control problems is to leverage detailed models of the system's dynamics and limitations. Such model-based approaches have found wide application in highly dynamic problems in automated driving and flight control [1], [2]. Leveraging a physical model can be very data efficient, an especially important consideration when operating near a system's limits where data collection is expensive and the model can often be sensitive to small parameter changes. Learning approaches, which act to leverage data from the system to develop models or control policies instead of using an *a priori* physical model, provide a contrasting approach with successes in a variety of domains [3], [4], [5], [6]. Though these approaches can require a large amount of data to learn policies or data-intensive models, they have the benefit of learning from the true system without the simplifications inherent in any physical modeling approach [7], [8], [9], [10]. In particular, policy gradient methods have proven successful in real and simulated robotics problems from locomotion to manipulation [11], [12], but typically require numerous trials of interaction with the environment. This paper presents a simple policy gradient approach that leverages a dynamic model to achieve high data efficiency while learning the performance limits of a real automated race car.

The task of completing a racing circuit in the minimum amount of time provides an excellent example of motion control near a system's dynamic limits. Racing furthermore presents an avenue for the comparison of human vs. machine as in previous AI competitions [13], [14], [15]. Skilled drivers represent a difficult benchmark for automated systems because they complete a circuit in minimum time by fully utilizing the available road-tire friction, understanding a model of the vehicle, stabilizing the vehicle, and learning, all while staying on the road. Skilled drivers learn by making subtle

The review of this article was arranged by Associate Editor Abel C. H. Chen.

changes to a vehicle's control inputs such as gradually pushing back brake points and relying "on segment timing and overlaid data to eventually decipher which deviations are better", with segment timing often only differing by tenths of a second between iterations [16]. They also must maintain data efficiency because as they operate at the vehicle's limits, the tire and brake performance of the vehicle are only consistent for a few attempts at a circuit.

Automated approaches to vehicle control have demonstrated the effectiveness of model-based control at the limits but have not yet demonstrated the performance of a professional race car driver. By first planning and subsequently tracking a racing trajectory, Kapania and Gerdes showed that the combination of a simple vehicle model for feedforward control and linear feedback could control an Audi TT-S near the friction limits [17]. This approach generated comparable lap times to a skilled amateur driver but was slower than a professional [18]. Similarly, by calculating and tracking minimum curvature trajectories that respect vehicle limits of the Roborace DevBot platform, Heilmeier et al. show that these approaches can come within a tenth of a second of human drivers [19]. By training with over 50 hours of simulation data, Fuchs et al. are able to outperform humans in Gran Turismo with model-free learning approaches [20]. In contrast, Brunnbauer et al. are able to outperform model-free RL algorithms in simulation through learning from an imperfect state transition model in a model-based reinforcement learning environment with a fourth of the data [21].

While certain basic characteristics of the model are independent of the model parameters, such as the fact that braking earlier causes the vehicle to move slower later in a turn, the operating characteristics at the friction limits are highly sensitive to small parameter variations [22]. Although one approach is to directly learn these variations, such as spatially varying friction maps, skilled drivers are still able to drive the vehicle at its physical limits without an exact map of friction [23]. Building on the success of model-based approaches for automated racing, data efficient learning can offer the opportunity to improve lap after lap as the best human drivers do.

Learning techniques should provide a data-efficient method to increase performance that aims to minimize time and utilize both lateral and longitudinal inputs to the vehicle. To learn and increase tracking performance, Kapania and Gerdes demonstrated iterative learning control using steering on an automated race car [24]. They showed increased path tracking performance but only used steering as an input and did not directly minimize lap time. Rosalia et al. successively minimized lap time using iterative learning model predictive control over tens of trials - about an order of magnitude more than professional drivers use [25]. They showed that starting from an initial safe trajectory and speed profile, the controller could learn a cost-to-go model, a safe terminal state set, and a number of locally linear models. Kabzan et al. are use simple vehicle models in model predictive control and use data to account and learn the model inaccuracies via gaussian process regression, leading to a lap-time reduction of up to 10 % [26]. Similarly, Georgiev et al. use model predictive control with a combination of a parametric vehicle model estimator and a non-parametric neural network model to learn model residuals [27].

In contrast, Abbeel et al. demonstrated that data efficient policy improvement is possible by using the gradient of the performance objective and a simple approximate model [28]. By achieving improved tracking performance on a fixed wing aerial vehicle and RC car, these results demonstrate how vehicle models can provide useful knowledge even when simplified. Furthermore, Kolter demonstrated that even just knowing the correct sign of the model's gradients shows policy improvement is possible [29]. By using an approximate vehicle model in combination with a model-based policy search, this paper demonstrates data efficient learning during the challenging task of automated racing on a real vehicle.

The contributions of this paper are outlined as follows. First, this paper shows a learning method for automated racing trajectories capable of learning longitudinal and lateral feedforward commands. Second, it presents the data efficiency of this approach on a full-size vehicle by showing improvement after just two trials of experiments during both high- and low-friction racing on oval test tracks. This approach improves upon an existing feedforward and feedback path tracking controller, already comparable to skilled amateur drivers, by adding gradient-based learning to update feedforward steering, brake, and throttle commands. Importantly, by including a model-based policy search, this approach decreases lap times by 0.69 s on an 18.46 s oval lap on a high-friction race course and a 1.75 s on a 29.11 s low-friction course. These experiments demonstrate the ability of using recorded data in a model-based policy search for control at the limits on an automated race car where data efficiency is critical.

II. METHODS

In order to improve performance, the gradient of lap time with respect to the control parameters is used to update the feedforward control policy during the next lap of automated racing. Inspired by how professional drivers lack a perfect understanding of how the vehicle exactly responds to control inputs, this approach uses an approximate model to calculate the gradient of the performance metric with respect to the control inputs. Rather than using an elaborate learned neural network or multi-body vehicle dynamic model, we show that gradients can be calculated by a simple single track bicycle model. While learned models are more complex, vehicle models might offer more accurate model predictions. The bicycle model offers a simple and effective model for use in gradient calculations, and as long as the sign of the gradient matches the true policy gradient direction, improvement is still possible [6], [29].

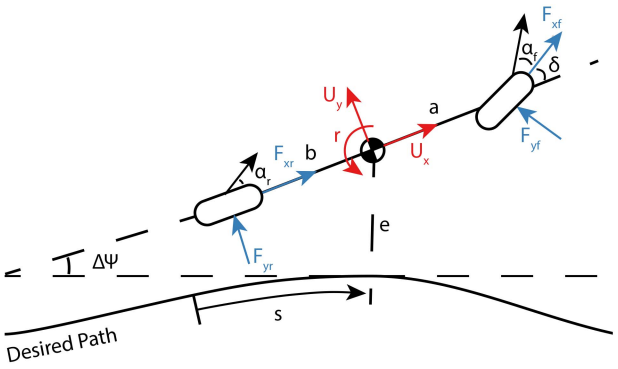


FIGURE 1. The bicycle model is used for both initial trajectory optimization and gradient learning.

A. VEHICLE MODEL

The simple vehicle model used for model-based learning as well as for initial trajectory optimization is the planar single track bicycle model shown below in Fig. 1 [30]. The single track bicycle model represents an approximation of a true four wheel vehicle model in which the vehicle's front and rear wheels are combined into a single "lumped" wheel at each axle. This model also only assumes planar vehicle motion, neglects multi-body dynamics and does not attempt to model the vehicle's suspension dynamics. Additionally, while the parameters of this model are optimized to recorded data, it does not represent a learned neural network model. The vehicle's velocity states consist of U_x, the vehicle's longitudinal velocity, U_y, the vehicle's lateral velocity, and r, the vehicle's angular velocity. The vehicle's sideslip as shown in Eq. (1), is calculated with both the lateral and longitudinal velocities.

$$\beta = \arctan\left(\frac{U_y}{U_x}\right) \tag{1}$$

The vehicle's steering angle is shown in Fig. 1 as δ, and the distance of the vehicle's center of mass (CM) to the parameterized path is e. The heading error from a line tangent to the parameterized path is $\Delta\Psi$, and the distance along the reference path is s.

The lateral forces are denoted as F_y and modeled by the Fiala tire model [31]. The length from the CM to the front axle is a, length of the CM to the rear axle is b, and the longitudinal forces are noted as F_x and act at the front and rear axles. The vehicle equations of motion shown in Eq. (2)-(4) are a function of the longitudinal forces F_x, the lateral forces F_y, and the distances from the CM to the front and rear axles.

$$\dot{U}_y = \frac{F_{yr} + F_{yf}\cos\delta + F_{xf}\sin\delta}{m} - rU_x \tag{2}$$

$$\dot{U}_x = \frac{F_{xr} - F_{yf}\sin\delta + F_{xf}\cos\delta}{m} + rU_y \tag{3}$$

$$\dot{r} = \frac{aF_{\mathrm{yf}}\cos\delta + aF_{xf}\sin\delta - bF_{\mathrm{yr}}}{I_z} \tag{4}$$

TABLE 1. Vehicle model parameters.

Parameter	Symbol	Unit
vehicle mass	m	kg
Distance front axle to CM	a	m
Distance rear axle to CM	b	m
Height of CM	h	m
Yaw moment of inertia	I_z	$\mathrm{kg\,m^2}$
Constant approx. suspension motion	K_{wt}	$\mathrm{s^{-1}}$

The vehicle's velocity states in combination with the local path curvature κ, heading error, and lateral error are used to calculate the derivatives of the kinematic states, where $\dot{s}$ in Eq. (7) is used in calculating the vehicle's lap time.

$$\Delta\dot{\Psi} = r - \dot{s}\kappa \tag{5}$$

$$\dot{e} = U_x \sin\Delta\Psi + U_y\cos\Delta\Psi \tag{6}$$

$$\dot{s} = \frac{U_x\cos\Delta\Psi - U_y\sin\Delta\Psi}{1-\kappa e} \tag{7}$$

In order to denote the gradients of the control policy with respect to the optimization objective we define the state at a particular discrete position along the trajectory as x_s, the control vector along a point in the trajectory as u_s, and the one-step discrete dynamics function as f_s. The control vector consists of δ, the vehicle's steering angle, and the longitudinal forces F_{xf} and F_{xr} which represent accelerating and braking the vehicle. The state vector also includes the change in normal force on each axle ΔF_z, which is computed assuming simple first order dynamics. The longitudinal weight transfer dynamics are shown in Eq. (8), where h represents the height of the vehicle's CM and K_{wt} represent a constant chosen to approximate suspension motion. These dynamics assume the presence of only road curvature and absence of any road topography.

$$\Delta\dot{F}_z = -K_{wt}\left(\Delta F_z - \frac{h}{L}\left(F_{xf}\cos\delta - F_{yf}\sin\delta + F_{xr}\right)\right) \tag{8}$$

By including longitudinal load transfer, the state x_s, the control u_s, and the dynamics function f_s are shown below. The dynamics function f_s represents the spatially discretized form of the continuous temporal dynamics presented in equations (4)-(8). The state x_{s-1}, the control u_{s-1} represent the inputs to the model f_s, whereas the next state x_s represents the model's output.

$$x_s = \left[U_y, r, U_x, \Delta\Psi, e, \Delta F_z\right] \tag{9}$$

$$u_s = \left[\delta, F_{xf}, F_{xr}\right] \tag{10}$$

$$x_s = f_s(x_{s-1}, u_{s-1}) \tag{11}$$

A summary of used vehicle model parameters is listed in Table 1.

B. TRAJECTORY OPTIMIZATION

An offline nonlinear optimization problem using the bicycle model along with a number of constraints is used to generate the initial racing trajectory. In this problem the cost function

J consists of lap time in addition to $g(x, u)$, which represents additional costs on control inputs as described in Subosits and Gerdes and is shown below [32]. This problem uses the bicycle model as well as the states x_s and controls u_s described in Eq. (11).

$$g(x, u) = q_1 P_{tire} + q_2 P_{brake} + q_3 \dot{\delta}^2 \tag{12}$$

The cost terms incorporated in $g(x, u)$ are time independent and represent less than one percent of the total cost. They include the terms P_{tire}, the rate of energy dissipation in the tires, P_{brake}, the rate of energy dissipation in the brakes, and $\dot{\delta}^2$, the steering slew cost. They are weighted by the constants $q_1, q_2,$ and q_3. The additional cost terms lead to trajectories with less control input and less tire and brake wear. These terms additionally act as a means to increase the optimization's robustness to model uncertainty. By weighing excessive wear on the vehicle, the optimization is incentivized to not over-leverage the model's estimate of road-tire friction. The resulting complete cost function is shown below.

$$J = \int_0^{s_{max}} \frac{1 + g(x, u)}{\dot{s}} ds \tag{13}$$

Additionally, the optimization uses constraints on the vehicle's lateral error e to represent track boundaries (e_{lb}, e_{ub}), the vehicle's steering angle δ and steering rate $\dot{\delta}$ to represent actuator constraints on steering. In addition the brake forces are upper bounded by zero, and the engine power is bounded by the engine's minimum and maximum power outputs. The longitudinal force F_x is bounded at each axle by the available force which is a function of the available friction μ and the slip angle α. Lastly the lap is constrained to be continuous. The resulting nonlinear optimization problem is shown below.

$$\begin{aligned}
\underset{x,u}{\text{minimize}} \quad & \sum_{s=1}^{s_{max}} \frac{1 + g(x, u)}{\dot{s}} \Delta s \\
\text{subject to} \quad & x_s = f_s(x_{s-1}, u_{s-1}), \ s = 1 \ldots s_{max} - 1 \\
& e_{lb} \leq e_s \leq e_{ub}, \ s = 1 \ldots s_{max} - 1 \\
& P_{min} \leq F_{x,eng}\big(U_x \cos\delta + \big(U_y + ar\big) \sin\delta\big) \leq P_{max}, \\
& s = 1 \ldots s_{max} - 1 \\
& -\mu F_z \cos\alpha \leq F_x \leq \mu F_z \cos\alpha, \ s = 1 \ldots s_{max} - 1 \\
& \delta_{min} \leq \delta_s \leq \delta_{max}, \ s = 1 \ldots s_{max} - 1 \\
& \dot{\delta}_{min} \leq \dot{\delta}_s \leq \dot{\delta}_{max}, \ s = 1 \ldots s_{max} - 1 \\
& F_{x,brake} \leq 0, \ s = 1 \ldots s_{max} - 1 \\
& x_0 = x_{s_{max}} \\
& u_0 = u_{s_{max}}
\end{aligned}$$

The optimization problem is implemented using CasADi in MATLAB 2016B and solved using IPOPT [33], [34], [35]. An initial reference trajectory is used as an initial guess for the nonlinear solver as well as to provide a coordinate system for vehicle dynamics.

C. PATH TRACKING CONTROLLER

Once the optimal control inputs and resulting path are calculated via solving the nonlinear optimization problem, the optimal inputs are used as feedforward control inputs denoted in Eq. (14)-(16) by $F_{xf,ffw}$, $F_{xr,ffw}$, and δ_{ffw}. The resulting control law for longitudinal control uses speed feedback to the desired speed profile. The amount of longitudinal force on each axle is proportioned using f_r, the longitudinal force distribution that is a byproduct of solving the optimization problem. The speed tracking gain is shown as K_x, the lane-keeping gain is K_{lk} and the lookahead distance is x_{la}. For lateral control, the steering controller from Kapania and Gerdes as shown in Eq. (16) uses both lateral error and heading error as well as feedforward sideslip shown as β_{ffw} to compensate for model inaccuracies and disturbances [17].

$$F_{xf} = F_{xf,ffw} + K_x f_r \big(U_{x,des} - U_x\big) \tag{14}$$

$$F_{xr} = F_{xr,ffw} + K_x (1 - f_r) \big(U_{x,des} - U_x\big) \tag{15}$$

$$\delta = \delta_{ffw} - K_{lk} \big(e + x_{la} \sin\big(\Delta\Psi + \beta_{ffw}\big)\big) \tag{16}$$

D. POLICY LEARNING

For policy improvement, the bicycle model is used to conduct a model-based policy search. From lap-to-lap, the feedforward longitudinal forces and steering are updated. These controls are represented by a vector θ along the full track planning horizon H as shown in Eq. (17). To calculate the update of the feedforward policy, the cost function is decomposed into individual terms as shown in Eq. (18), comprised of the stagewise cost along the planning horizon. In the update, the cost function consists solely of time rather than the additional terms shown in Eq. (13). While additional terms are used in solving the optimization problem for the initial controls and path, these terms account for less than one percent of J. Additionally, the use of only time minimization in the gradient update provides a baseline for model-based policy search. For model-based policy search, the objective is to take small gradient steps around the initial optimal policy so this approach first consider only the objective of lap time.

$$\theta = [u_0, \ldots, u_H] \tag{17}$$

$$J(\theta) = \sum_{s=0}^{H} j(x_s) \tag{18}$$

To estimate the policy gradient, as shown by Abbeel et al., the update can be decomposed as the gradient of the stagewise cost multiplied by the Jacobian of derivatives of each state coordinate with respect to each entry of the feedforward controls [28]. This approach differs from those typically used in policy gradient methods because of its ability to leverage a model to estimate the policy gradient itself. Traditionally in policy gradient methods, such as the likelihood ratio policy gradient, the policy gradient is calculated from rollouts using perturbed policies on the system without the need for knowledge of a dynamics or reward model. The policy gradient observes the reward from each rollout and

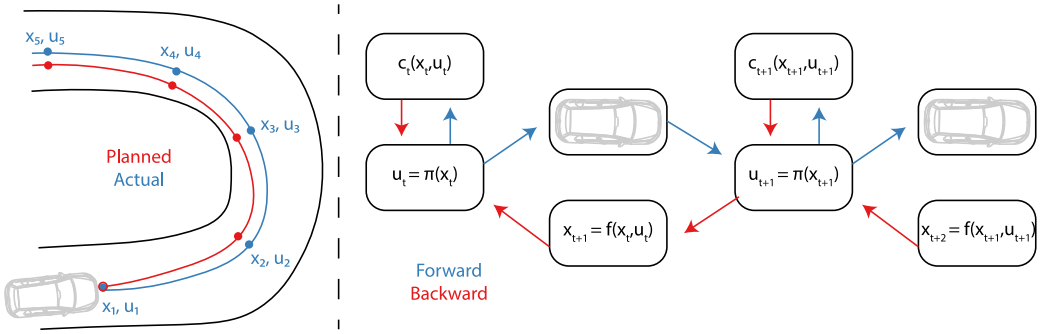


FIGURE 2. Planned vs. actual states are shown on the left. The difference between the forward evaluation pass and backwards pass is shown on the right, where the vehicle is used for real world evaluation rather than a forward pass in simulation.

FIGURE 3. The Volkswagen Golf GTI experimental vehicle platform is shown on the left at a high-friction race track and on the right at a low-friction test facility.

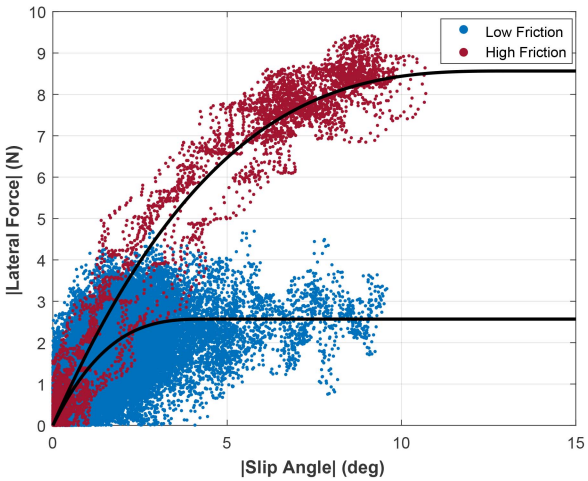


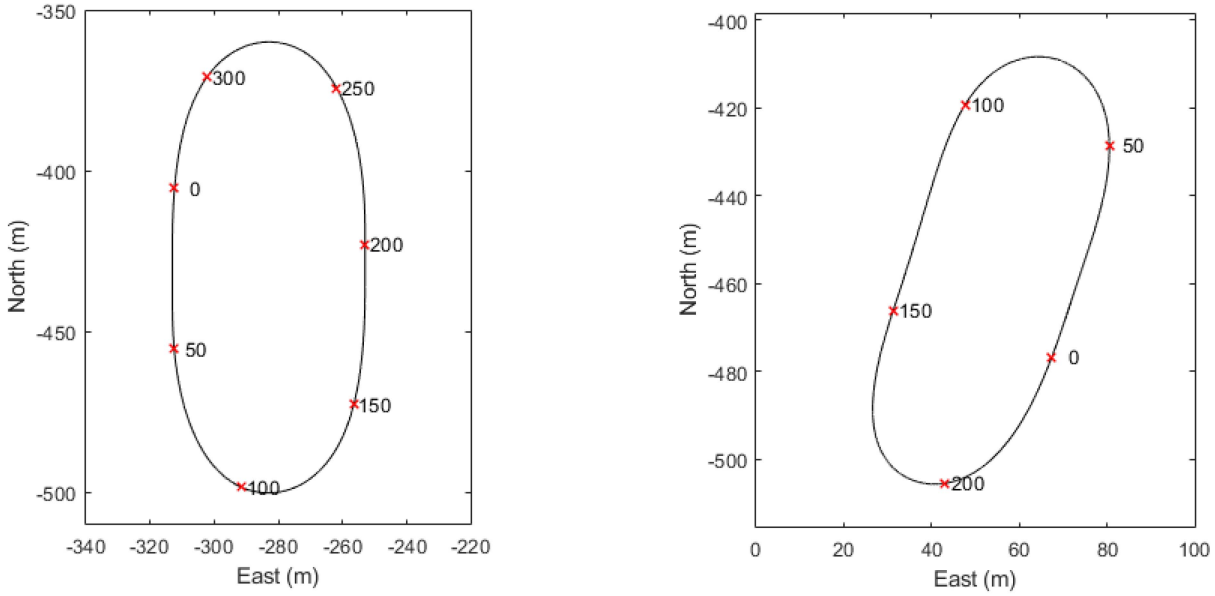
FIGURE 4. Experimental data from testing on high- and low-friction surfaces illustrates the general fit of the model. In practice, there is wide variation in the amount of lateral force for a given slip angle, especially on low-friction surfaces [40].

makes the more rewarding trajectories more likely [36], which makes the policy significantly dependent on the quality of the observed rewards [37]. Similarly, in finite difference policy gradient approaches, the policy is successively perturbed without a model by small amounts in each parameter to calculate the best direction of policy improvement [38]. While these approaches tend to be data intensive, the approach we present of model-based policy search differs by leveraging the approximate vehicle model, a known reward structure, and recorded vehicle data to construct an estimated policy gradient from only a single lap of on vehicle data.

Rather than using the predicted state sequence from the feedforward controls and model as used in direct methods for optimal control for lap-time minimization, the state sequence used in evaluation of the policy gradient consists of the states and controls executed along the true trajectory [39]. By using the true recorded state and control trajectory, the derivatives appearing in the policy gradient update equation more accurately represent those experienced during experiment as shown in Fig. 2. This is equivalently represented in Fig. 2 by the forward pass in policy evaluation, which occurs on the vehicle rather than in simulation. The backward pass and policy update use the model along the recorded trajectory of states and controls. For the policy update, the feedback controllers shown in Eq. (14)-(16) are added in to the bicycle model dynamics f_s and noted as $f_{s,c}$. Inclusion of feedback for longitudinal and lateral control gives the learning process the understanding of the path following behavior of the controller while updating the feedforward control inputs.

To update the feedforward control inputs, the policy gradient as shown in Eq. (19) can be calculated from the recorded data trajectory. The Euler discretized dynamics transition matrices are shown in Eq. (20)-(21) and represented as A_s and B_s denoting the linearized dynamics around the recorded trajectory at a distance along the trajectory s.

$$\nabla_\theta J(\theta) = \sum_{s=0}^{H} \nabla_x j(x_s) \frac{df_{s,c}}{d\theta}|_{x_0,x_1,\ldots,x_{s-1}} \tag{19}$$



a High-friction **b** Low-friction

FIGURE 5. Oval test tracks for testing on high and low-friction represent an experimental scenario for model-based policy search. The oval track represents the simplest track geometry to show the minimum set of braking, turning, and acceleration control actions required for a race car. Red *x*s are shown denoting fixed distances along the path in meters.

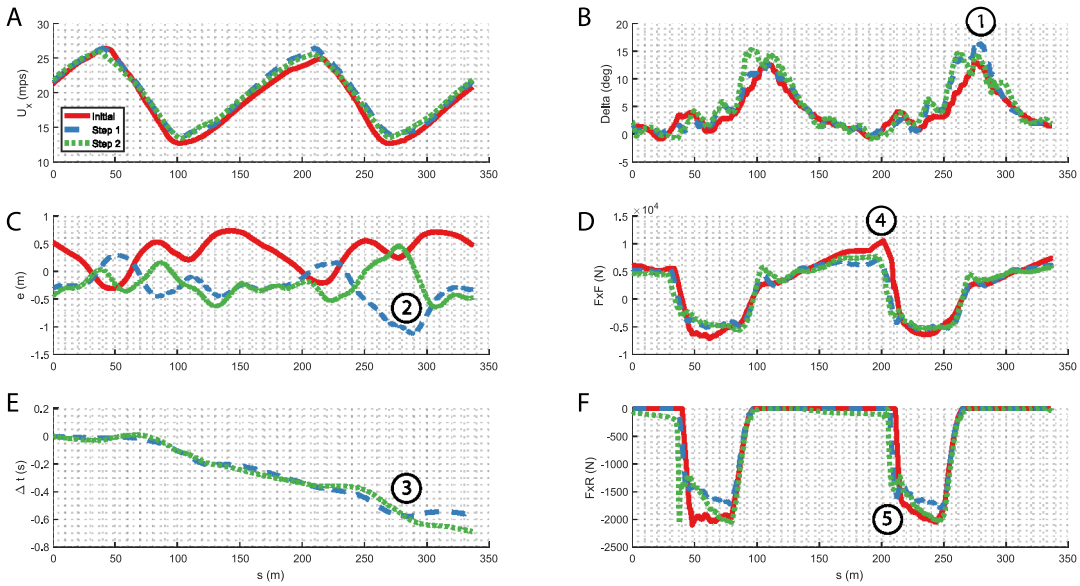


FIGURE 6. The velocity profile, lateral deviation from the planned path and the time advantage are shown on the left from high-friction testing. The applied control sequence to decrease lap time is presented on the right.

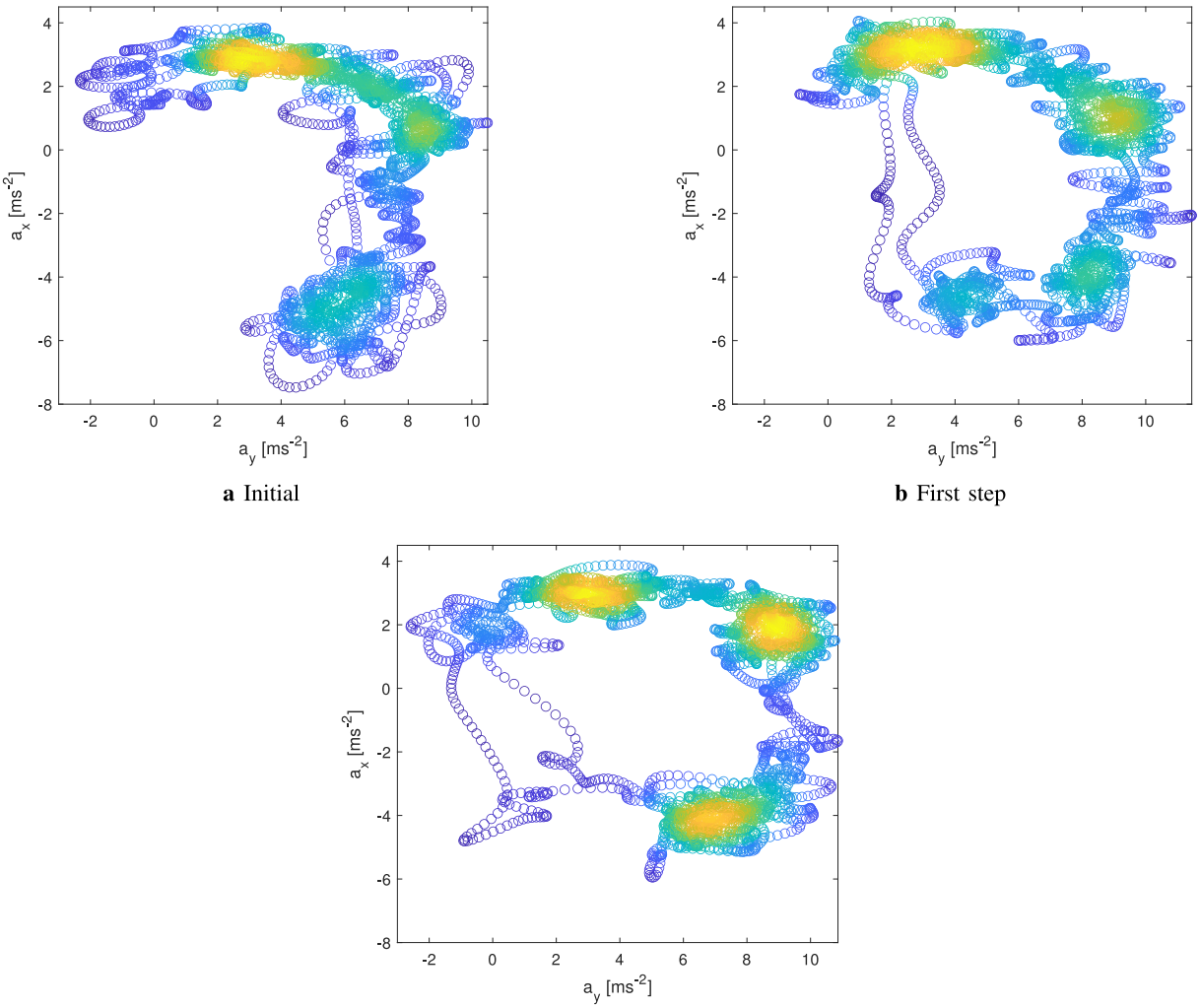
$$A_s = \frac{\partial f_{s,c}(x_{s-1}, u_{s-1})}{\partial x_{s-1}} \tag{20}$$

$$B_s = \frac{\partial f_{s,c}(x_{s-1}, u_{s-1})}{\partial u_{s-1}} \tag{21}$$

The partials appearing in $\frac{df_{s,c}}{d\theta}$ consist of evaluating the chain rule along the trajectory from each control input to the resulting output state along the horizon. For example as shown in Eq. (23), when calculating the update for the initial control inputs, terms for the cost for each segment in the planning horizon appear because the first control affects all subsequent states in the Markov process. Partials of the feedforward policy with respect to the vehicle state do not appear because the policy is only a function of s along the trajectory and not a function of the vehicle state.

$$\nabla_{u_0} J(\theta) = \nabla_x j(x_H) A_H, \ldots, A_2 B_1 \tag{22}$$

$$+ \nabla_x j(x_{H-1}) A_{H-1}, \ldots A_2 B_1 + \cdots + \nabla_x j(x_1) B_1 \tag{23}$$



a Initial

b First step

c Second step

FIGURE 7. Comparison of the lateral and longitudinal accelerations from experimental testing on a high-friction surface. The diagram is colored by the spatial density of nearby points where yellow indicates a high spatial density. By learning policy updates that empirically minimize time, the vehicle spends more of its time exploiting its maximum acceleration capabilities or limits. This is shown by a higher spatial density of points in yellow around the friction circle.

Once the update is calculated, it is applied to the feedforward control sequence for the following round shown in the gradient update in Eq. (24) where α is the learning rate, and i denotes current round of collected data. This learning approach represents an offline policy update, because the learning process happens in between successive rounds of decision making. Rather than represent the policy as a neural network, the policy is represented as a lookup table parameterized with the discrete distance along the reference path as an input and the desired feedforward control value as an output.

$$\theta_{i+1} = \theta_i - \alpha \nabla_\theta J(\theta) \tag{24}$$

III. RESULTS AND EXPERIMENTS

The goal of the following experiments is to demonstrate the ability of using recorded data to calculate policy gradients used for control on real-world robotics systems where data efficiency is critical. All of the experiments were executed on a full-size vehicle as shown in Fig. 3 and evaluated under multiple environmental conditions as shown in Fig. 4. Each environmental condition is characterized by conducting a ramp steer maneuver, where the vehicle steering is linearly increased until the vehicle slides while driving at a constant speed. Data from this maneuver in Fig. 4 allows for calculation of best initial guesses at the global friction coefficients for testing on multiple environmental conditions. Fitting high-friction ramp steer data leads to a fit friction coefficient of 0.92, and fitting low-friction ramp steer data leads to a fit friction coefficient of 0.25. While high-friction tarmac has consistent grip characteristics, low-friction testing provides greater grip variation.

This creates an interesting test case for learning because as the amount of available grip degrades over time, the sample efficiency of the learning approach becomes important. Each experiment follows the same procedure of initially computing

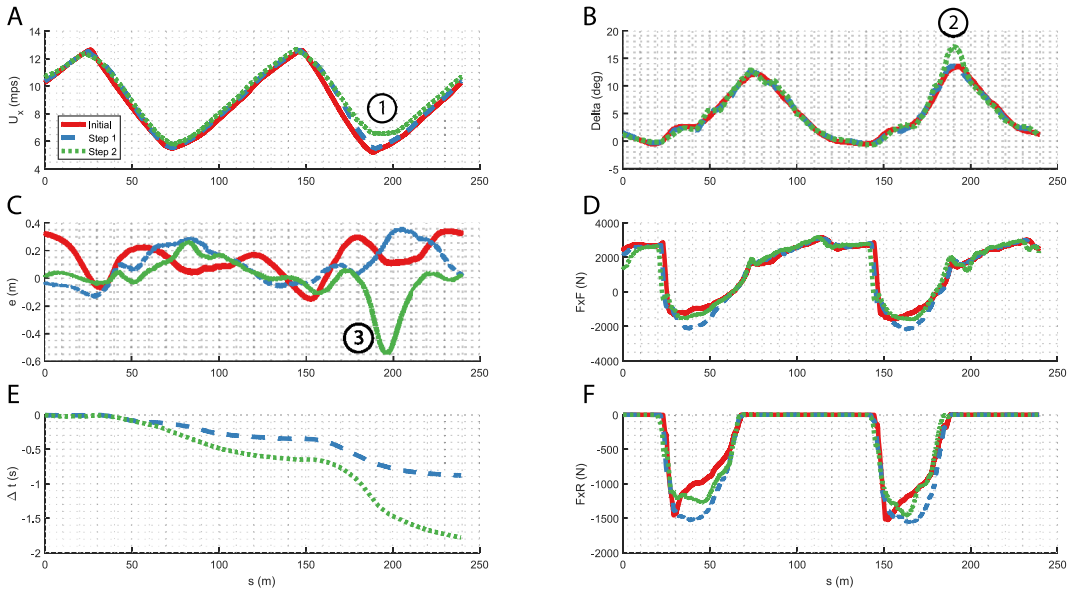


FIGURE 8. The velocity profile, lateral deviation from the planned path and the time advantage from low-friction testing are shown on the left. The track layout was on an icy surface resulting in a friction coefficient below 0.3. The applied control sequence to decrease lap time is also presented on the right.

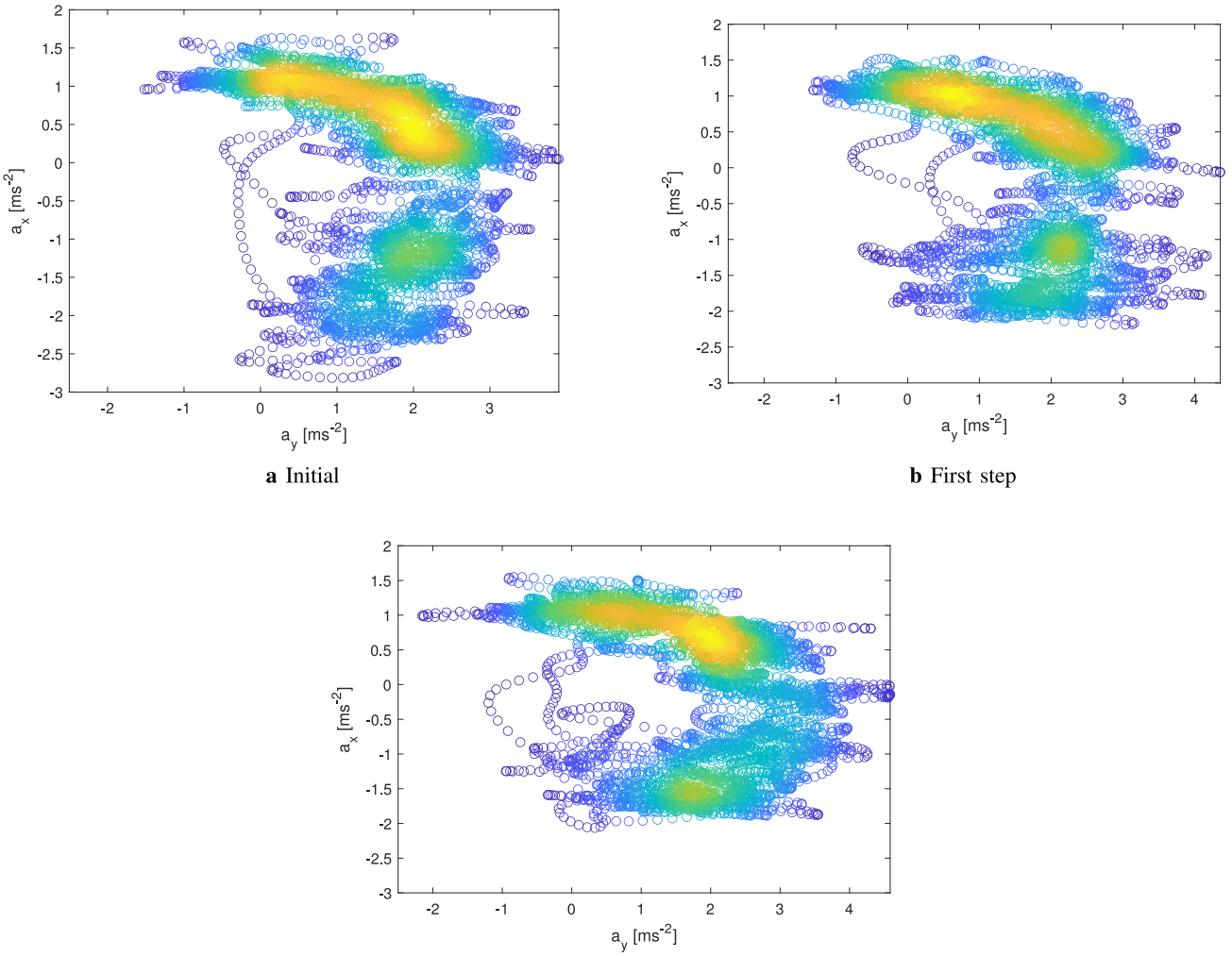
an optimal policy based on a simplified vehicle model, collecting real-world experience based on the initial policy and using the same model to update our policy based on the computed gradients as described in Section II-D. These experiments represent informative tests of the learned policy model's performance because sub-optimal models will learn to brake, accelerate, or steer in sequences that result in sub-optimal lap times. The learning process aims to optimize lap time, and hence maximize the use of the available tire forces which are uncertain due to friction uncertainty. Therefore learning to race at the limits of friction represents a challenging and motivating problem for model-based policy search.

A. HIGH-FRICTION TESTING

First, the model-based policy search is demonstrated under high-friction conditions at Thunderhill Raceway in California. To speed up the learning process, we chose an oval track layout with a total length of 336 m. The planned path can be seen in Fig. 5(a). This path is used as an input to the path tracking controller, which is used to collect the data set of the initial policy using a feedforward-feedback control approach as described in Section II-B and II-C. After collecting one policy rollout, we process the dataset and perform a gradient update step as shown in Section II-D. Using the previously driven path and updated controls, the policy is then reevaluated during the following trial on the vehicle.

The results of learning on high-friction are shown in Fig. 6. We observe a time advantage of 0.57 s after one gradient step which consists of only 18.46 s of data to learn from. After the second gradient step, the lap is 0.69 s faster than the initial trajectory optimization solution showing substantially more than a tenth of a second improvement in only two laps of learning. Furthermore, the controller increases the cornering exit speed as seen in the velocity profile in Fig. 6A. The computation of the policy update took an average of 11.06 seconds with a standard deviation of 0.289 seconds using a Lenovo ThinkPad P43s with an i7 8565U 1.8Ghz processor, an Nvidia Quadro P520 GPU, and 8GB of RAM.

By examining the difference between the first and second gradient step, it is clear that the vehicle is exploring the available road-tire friction capabilities. This exploration is different from traditional exploration in reinforcement learning methods, because in this case the grip exploration is informed directly by the policy update rather than an exploration policy. We can see in Fig. 6B, that the car is slightly understeering as shown by the increased steering angle in ①, leading to a rise in the magnitude of lateral deviation shown in Fig. 6C ②. Understeer occurs when the vehicle's front tires are sliding, resulting in the vehicle turning less than intended by the driver. Because the front tires are sliding, the vehicle experiences accumulated lateral error relative to the reference path. Sometimes understeer off the path can be slow and therefore correlate with a loss of time which can be shown in Fig. 6E ③. As the car is already on its friction limits, increased steering, as shown in Fig. 6B, lacks the ability to generate lateral force. In the next gradient step, the updated policy learns that this degree of understeer is slow and corrects for it. The policy search understands that if there is no available friction margin on the front axle, a decrease in track time can be achieved by reducing corner entry speed and braking earlier. The updated policy both reduces its speed and brakes earlier into the corner as shown in Fig. 6D ④ and 6F ⑤. By changing the braking strategy in the second policy update step, lap time decreases. As a direct correlation, the vehicle achieves higher levels



a Initial

b First step

c Second step

FIGURE 9. Comparison of the lateral and longitudinal accelerations from experimental testing under low-friction conditions. The diagram is colored by the spatial density of nearby points where yellow indicates a high spatial density. Additionally, by learning a policy that minimizes time, the plots show that the vehicle has a higher spatial density of points when the vehicle is fully cornering (zero longitudinal acceleration). These plots show that by having a higher spatial density of points at higher longitudinal and lateral accelerations, the vehicle is able to utilize more of its full control capabilities and minimize lap time.

TABLE 2. Results of the initial and learned policies executed on an oval at Thunderhill race track during high-friction testing.

type	lap time [s]	U_x [m s^{-1}]			e [m]		
		max	min	mean	max	min	mean
initial	18.46	26.40	12.64	18.58	0.74	-0.31	0.45
step 1	17.89	26.36	13.63	19.40	0.29	-1.12	0.49
step 2	17.77	25.90	13.40	19.38	0.47	-0.65	0.35

of lateral acceleration as it operates closer at the vehicle's friction limits to decrease lap time as shown in Fig. 7.

B. LOW-FRICTION TESTING

Lastly, the performance of the model-based policy search is tested in a different environmental condition by driving on a low-friction surface on a frozen lake near the Arctic Circle. For comparison, this paper uses an oval with a track length of 239 m as presented in Fig. 5(b) and similarly evaluates an initial trajectory optimization solution using the single track model along with two gradient learning steps. Fig. 8 shows experimental results from the learning while driving on a low-friction surface. These results take an average of 15.35 seconds with a standard deviation of 0.97 seconds to compute using an HP Z-Book with an i7 7820HQ 2.9Ghz processor with an Nvidia Quadro M2200 GPU.

These results highlight the robustness of learning under different environmental conditions as well as the sample efficiency of the approach. After just 29.11 s of data collected in the initial run, the first gradient step leads to a time advantage of 0.88 s. Another step decreases the lap time by 0.87 s leading to the results presented in Table 3. By calculating the second gradient step, the speed profile is increased as shown in Fig. 8A. As the car moves towards its friction limits in step 2 by increasing the cornering speed in 8A ①, it begins to understeer at the apex. This is shown by the increased steering in 8B ② and lateral deviation

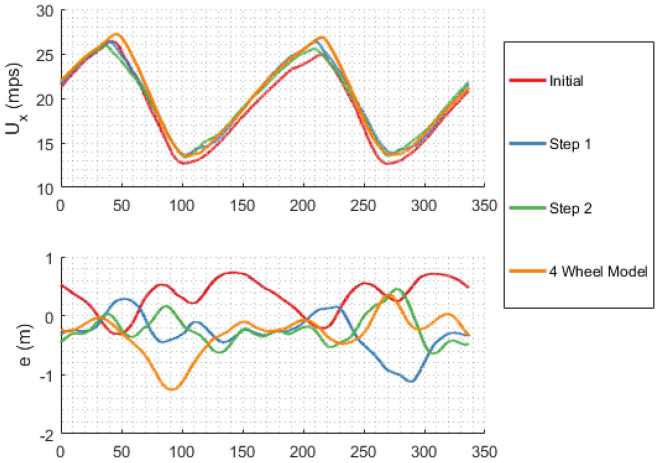


FIGURE 10. Comparison to a high-fidelity four wheel vehicle Model shows that model-based policy search is capable of using a simpler model to surpass the performance of a more complex model. The top panel shows comparison of longitudinal velocity and bottom panel shows tracking error during comparisons near the limits on a high-friction oval at Thunderhill Raceway.

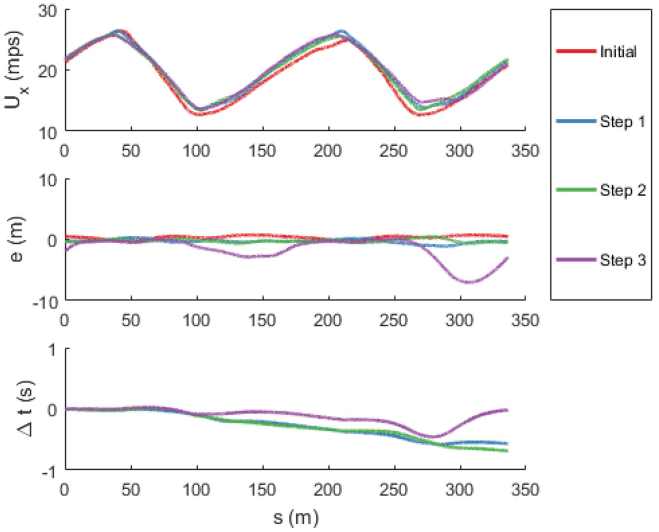


FIGURE 11. Third gradient update step showing effect of increased learning rate.

TABLE 3. Results of initial and learned policies executed on an oval near the Arctic Circle during low-friction testing.

type	lap time [s]	U_x [m s^{-1}]			e [m]		
		max	min	mean	max	min	mean
initial	29.11	12.64	5.19	8.49	0.34	-0.15	0.18
step 1	28.23	12.62	5.50	8.72	0.36	-0.13	0.18
step 2	27.36	12.69	5.76	9.00	0.26	-0.55	0.17

from the planned path shown in 8C ③. Though the vehicle is fully sliding on ice, the increased amount of steering and lateral deviation in practice does not decrease the vehicle's lap time. Because lap time decreases after each gradient step, the vehicle spends more time near its limits as shown by examining the density of measured accelerations in Fig. 9(c). The yellow areas show the increased ability to brake and corner harder after successive gradient updates. This shows that by learning, the control system is operating near the true friction limits just as skilled drivers do.

Testing the algorithm on ice allows for the ability to operate in more uncertain friction conditions. Although initially estimated based on different driving maneuvers, the road-tire friction coefficient on the frozen lake changes with time and location. Driving on the same path polishes the initially snow-covered ice which decreases the available friction. Changing surface conditions can result in a significant gap between initially assumed friction coefficient and actually available grip level. Some of this variation can be seen by comparing the time advantage between high- and low-friction testing and noting the wider variation in advantage in each of the corners during low-friction testing. Though the correct initial friction coefficient is not known because of changing conditions, learning using collected data still retains the ability to decrease lap time.

C. COMPARISON TO HIGHER FIDELITY MODEL

While during the learning process, the bicycle model was used for trajectory optimization and gradient updates, more complex models can lead to increasingly optimal racing trajectories. While using the bicycle model as a feedforward in combination with lookahead feedback has been shown to be comparable to a skilled amateur driver, four wheel vehicle models have demonstrated additional improvement over bicycle models for automated racing [18], [32]. This section similarly compares using a higher fidelity four wheel vehicle model to model-based policy search. In this comparison, the four wheel model is used for trajectory optimization and then subsequently tracking using the feedback controllers shown in Eqs. (14)-(16). This comparison is conducted on the same high-friction oval at Thunderhill Raceway as the comparison shown in Fig. 6.

The results from these tests are shown in Fig. 10. As shown, the four wheel model performs better than both the bicycle model or initial optimization solution and the first step of model-based policy search. During the second step of model-based policy search however, the policy search outperforms the four wheel model by five hundredths of a second. Additionally, the model-based policy search more closely tracks its intended trajectory compared to the four wheel model solution which understeers in the first corner. As shown in Fig. 10, the largest difference between the solutions is the increase in maximum speed and corner entry speed between the four wheel model and model-based policy search. This increase in speed leads to understeer in the corner and corresponding time loss as shown by the four wheel model.

This comparison highlights that though using a simpler model, additional gradient information of the model allows for a performance increase relative to using a more complex model for trajectory optimization. Additionally, this highlights how only an approximate model is necessary for trajectory optimization if learning is possible, building in a degree of model robustness to the control process. Further work using gradients of the four wheel model or more complex vehicle models could offer additional time

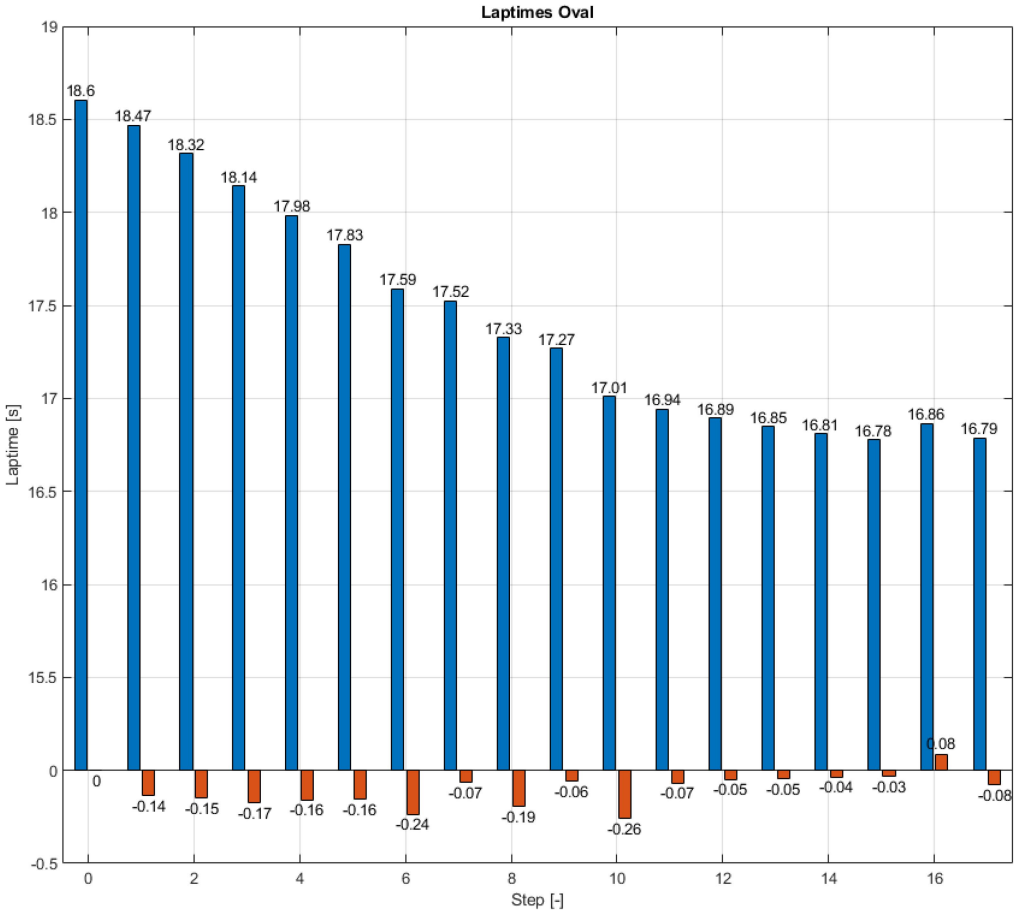


FIGURE 12. Choosing a smaller learning rate increases the number of steps until convergence.

improvement beyond using the bicycle model for gradient updates.

D. CHOOSING THE LEARNING RATE

Selecting the appropriate learning rate to decrease lap time can be a challenging task. Learning rates that are too small lead to policies that take many iterations to converge to an optimal solution while learning rates that are too large may lead to divergence. In practice, when the learning rate is set too high, applying the update will lead to an increase in lap time. This can be shown in Fig. 11 where in the third step of policy learning, the cornering speed in both corners is increased. This leads to the vehicle accumulating negative lateral error, understeering off of the intended path, and losing lap time relative to the previous steps.

In contrast to selecting a learning rate that is too large, another option is to select a much smaller learning rate. Smaller learning rates have the advantage of taking smaller steps on the objective and increasing stability. While smaller steps may lead to increased stability, it comes with the cost of increasing the time to arrive at an optimal policy. Fig. 12 demonstrates the convergence and increased stability of a smaller learning rate. In this experiment, the vehicle learns on a high-friction oval over the course of multiple laps. While the vehicle steadily decreases its lap time, it achieves increased lateral and longitudinal accelerations as shown in Fig. 13. Ultimately choosing the correct learning rate is a compromise between learning speed and system convergence.

IV. DISCUSSION AND CONCLUSION

Although a focus of research in policy gradient methods has been to increase sample efficiency, [5], [11], [41], and enable model-free methods for a variety of real-world applications [42], [43], [44], these methods still often require multiple trials worth of data. In contrast, model-based reinforcement learning has shown to be sample efficient in driving vehicles [45] but influences the learning process by induced model bias [46], [47]. We have shown multiple oval racing experiments at the limits of friction on a real vehicle, showing the sample efficiency and validity of learning using vehicle gradients. Learning from interaction by using a simple model as a gradient estimator is a promising way to further enhance robotics in real-world applications where data efficiency is important.

For future performance gains, the inclusion of a line search in the gradient update may lead to increased performance and stability. Besides using a line search, using baselines in

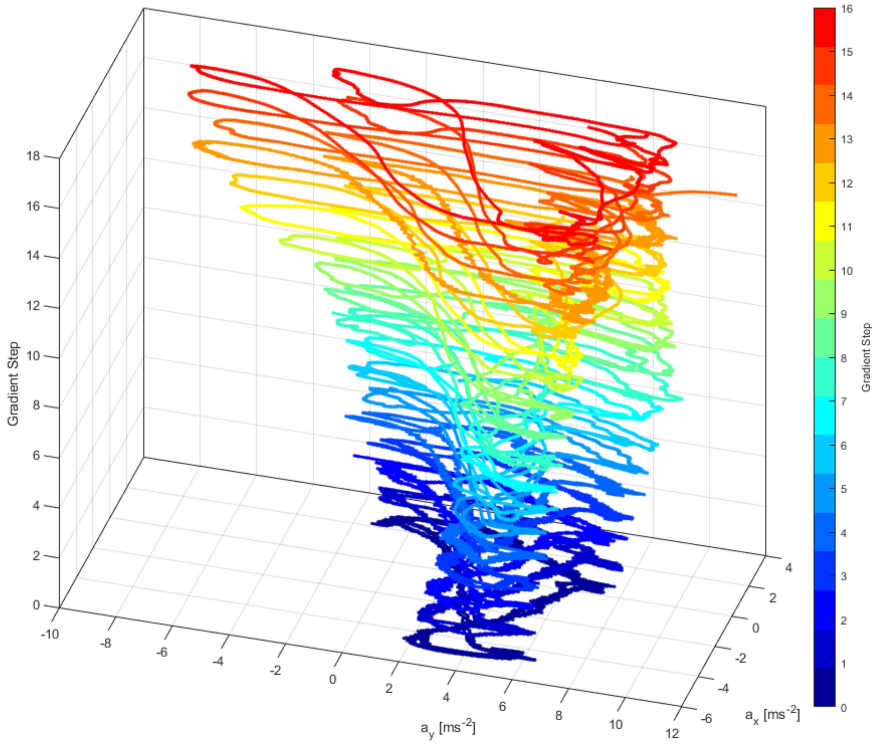


FIGURE 13. By selecting a smaller learning rate, the increase in lateral and longitudinal accelerations with each update step is evident. In each step in learning the vehicle is capable of achieving a larger lateral and longitudinal acceleration.

the update process can further stabilize the learning process when doing multiple gradient steps [48]. While each of these inclusions could lead to increased performance, additional data collection comes at a cost as environmental conditions can change.

The model-based policy search presented shows that using gradients of lap time calculated with an approximate vehicle model can lead to improved performance. This approach shows that uncertainties which are capable of being captured with their mean value and some variation can be used in the calculations of the vehicle model's gradients. As long as the sign of the gradients match the improvement direction, improvement should be possible [29]. While we showed the ability to improve vehicle performance in environments with highly variable friction, gradients can similarly updated due to changes of other vehicle parameters, such as vehicle mass. Though the vehicle's occupant mass accounts for less than 5 % of the total vehicle's mass and fuel consumption accounts for less than 2 % these parameters were not explicitly controlled for during testing, which shows that the approximate models used for calculation of the vehicle gradients still empirically improve performance. Future work is required to include parameter uncertainty explicitly in the calculation of the vehicle gradient update rule.

Inspired by how professional race drivers learn, model-based policy search is capable of learning at the limits of friction. While model-based policy search demonstrated improvement over the initial optimal racing trajectories, ultimately the accuracy of the gradient updates is based on the model. Near the friction limits, the performance of model-based policy search is sensitive to the model's measurement of friction. While model-based policy search can operate using approximate models, using a model with an inaccurate measurement of friction can lead to the vehicle trying to over exploit additional friction that does not exist in reality. Future methods could integrate model-based policy search with online model learning for more accurate model-based updates and performance.

ACKNOWLEDGMENT

The authors thank Volkswagen AG for their generous support by providing research vehicles, test facilities, and fostering research collaboration and discussion. The authors thank the Volkswagen Electronics Research Lab for providing vehicle support, the Dynamic Design Lab for providing support, and Thunderhill Raceway for providing test facilities. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] P. Falcone, F. Borrelli, J. Asgari, H. E. Tseng, and D. Hrovat, "Predictive active steering control for autonomous vehicle systems," *IEEE Trans. Control Syst. Technol.*, vol. 15, no. 3, pp. 566–580, Jun. 2007.

[2] A. Richards and J. P. How, "Model predictive control of vehicle maneuvers with guaranteed completion time and robust feasibility," in *Proc. Amer. Control Conf.*, vol. 5, 2003, pp. 4034–4040.

[3] P. Cai, X. Mei, L. Tai, Y. Sun, and M. Liu, "High-speed autonomous drifting with deep reinforcement learning," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 1247–1254, Apr. 2020.

[4] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*.

[5] V. Mnih et al., "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.

[6] N. A. Spielberg, M. Brown, N. R. Kapania, J. C. Kegelman, and J. C. Gerdes, "Neural network vehicle models for high-performance automated driving," *Sci. Robot.*, vol. 4, no. 28, 2019, Art. no. e1975.

[7] C. Xie, S. Patil, T. Moldovan, S. Levine, and P. Abbeel, "Model-based reinforcement learning with parametrized physical models and optimism-driven exploration," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2016, pp. 504–511.

[8] P. Abbeel, A. Coates, M. Quigley, and A. Y. Ng, "An application of reinforcement learning to aerobatic helicopter flight," in *Proc. Adv. Neural Inf. Process. Syst.*, 2007, pp. 1–8.

[9] M. Cutler and J. P. How, "Autonomous drifting using simulation-aided reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2016, pp. 5442–5448.

[10] T. K. Lau and Y.-H. Liu, "Stunt driving via policy search," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2012, pp. 4699–4704.

[11] J. Peters and S. Schaal, "Policy gradient methods for robotics," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2006, pp. 2219–2225.

[12] S. Kuutti, R. Bowden, Y. Jin, P. Barber, and S. Fallah, "A survey of deep learning applications to autonomous vehicle control," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 2, pp. 712–733, Feb. 2021.

[13] M. Campbell, A. J. Hoane, Jr., and F.-H. Hsu, "Deep blue," *Artif. Intell.*, vol. 134, nos. 1–2, pp. 57–83, 2002.

[14] R. High, *The Era of Cognitive Systems: An Inside Look at IBM Watson and How it Works*, IBM Corporat., Bengaluru, India 2012, pp. 1–16.

[15] D. Silver et al., "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016. doi: 10.1038/nature16961.

[16] J. R. Hildebrand, "Feedback on testing at Thunderhill raceway," Oct. 2015.

[17] N. R. Kapania and J. C. Gerdes, "Design of a feedback-feedforward steering controller for accurate path tracking and stability at the limits of handling," *Veh. Syst. Dyn.*, vol. 53, no. 12, pp. 1687–1704, 2015.

[18] J. C. Kegelman, "Learning from professional race car drivers to make automated vehicles safer," Ph.D. dissertation, Dept. Comput. Sci., Stanford Univ., Stanford, CA, USA, 2018.

[19] A. Heilmeier, A. Wischnewski, L. Hermansdorfer, J. Betz, M. Lienkamp, and B. Lohmann, "Minimum curvature trajectory planning and control for an autonomous race car," *Veh. Syst. Dyn.*, vol. 58, no. 10, pp. 1497–1527, 2020.

[20] F. Fuchs, Y. Song, E. Kaufmann, D. Scaramuzza, and P. Dürr, "Super-human performance in gran turismo sport using deep reinforcement learning," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 4257–4264, Jul. 2021.

[21] A. Brunnbauer et al., "Latent imagination facilitates zero-shot transfer in autonomous racing," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2022, pp. 7513–7520.

[22] V. A. Laurense and J. C. Gerdes, "Speed control for robust path-tracking for automated vehicles at the tire–road friction limit," in *Proc. 14th Int. Symp. Adv. Veh. Control (AVEC)*, Beijing, China, 2018, pp. 1–9.

[23] F. Christ, A. Wischnewski, A. Heilmeier, and B. Lohmann, "Time-optimal trajectory planning for a race car considering variable tyre-road friction coefficients," *Veh. Syst. Dyn.*, vol. 59, no. 4, pp. 588–612, 2021.

[24] N. R. Kapania and J. C. Gerdes, "Path tracking of highly dynamic autonomous vehicle trajectories via iterative learning control," in *Proc. IEEE Amer. Control Conf. (ACC)*, 2015, pp. 2753–2758.

[25] U. Rosolia, A. Carvalho, and F. Borrelli, "Autonomous racing using learning model predictive control," in *Proc. IEEE Amer. Control Conf. (ACC)*, 2017, pp. 5115–5120.

[26] J. Kabzan, L. Hewing, A. Liniger, and M. N. Zeilinger, "Learning-based model predictive control for autonomous racing," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 3363–3370, Nov. 2020.

[27] I. Georgiev, C. Chatzikomis, T. Völkl, J. Smith, and M. Mistry, "Iterative semi-parametric dynamics model learning for autonomous racing," 2020, *arXiv:2011.08750*.

[28] P. Abbeel, M. Quigley, and A. Y. Ng, "Using inaccurate models in reinforcement learning," *Proc. 23rd Int. Conf. Mach. Learn. (ICML)*, 2006, pp. 1–8. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1143844.1143845

[29] J. Kolter, "Learning and control with inaccurate models," Ph.D. dissertation, Dept. Comput. Sci., Stanford Univ., Stanford, CA, USA, 2010.

[30] D. Schramm, M. Hiller, and R. Bardini, "Single track models," in *Vehicle Dynamics*. Berlin, Heidelberg: Springer, 2018, pp. 225–257.

[31] H. Pacejka, *Tire and Vehicle Dynamics*. Amsterdam, The Netherlands: Elsevier, 2005.

[32] J. Subosits and J. C. Gerdes, "Impacts of model fidelity on trajectory optimization for autonomous vehicles in extreme maneuvers," *IEEE Trans. Intell. Veh.*, vol. 6, no. 3, pp. 546–558, Sep. 2021.

[33] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi—A software framework for nonlinear optimization and optimal control," *Math. Program. Comput.*, vol. 11, pp. 1–36, Jul. 2019.

[34] A. Wächter and L. T. Biegler, "Global and local convergence of line search filter methods for nonlinear programming," Dept. Chem. Eng., Carnegie Mellon Univ., Pittsburgh, PA, USA, Rep. CAPD B-01-09, 2001. [Online]. Available: http://www.optim.-online.org/DB_HTML/08/367.html

[35] *MATLAB. version 9.1.0 (R2016b)*, MathWorks Inc., Natick, MA, USA, 2016.

[36] T. M. Moerland, J. Broekens, and C. M. Jonker, "Model-based reinforcement learning: A survey," 2020, *arXiv:2006.16712*.

[37] N. Hamilton, P. Musau, D. M. Lopez, and T. T. Johnson, "Zero-shot policy transfer in autonomous racing: Reinforcement learning vs imitation learning," in *Proc. IEEE Int. Conf. Assured Autonomy (ICAA)*, 2022, pp. 11–20.

[38] N. Kohl and P. Stone, "Policy gradient reinforcement learning for fast quadrupedal locomotion," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, vol. 3, 2004, pp. 2619–2624.

[39] N. Dal Bianco, E. Bertolazzi, F. Biral, and M. Massaro, "Comparison of direct and indirect methods for minimum lap time optimal control problems," *Veh. Syst. Dyn.*, vol. 57, no. 5, pp. 665–696, 2019.

[40] N. A. Spielberg, M. Brown, and J. C. Gerdes, "Neural network model predictive motion control applied to automated driving with unknown friction," *IEEE Trans. Control Syst. Technol.*, vol. 30, no. 5, pp. 1934–1945, Sep. 2022.

[41] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," 2015, *arXiv:1506.02438*.

[42] M. Riedmiller, M. Montemerlo, and H. Dahlkamp, "Learning to drive a real car in 20 minutes," in *Proc. IEEE Front. Converg. Biosci. Inf. Technol.*, 2007, pp. 645–650.

[43] A. Kendall et al., "Learning to drive in a day," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, 2019, pp. 8248–8254.

[44] R. Tedrake, T. W. Zhang, and H. S. Seung, "Learning to walk in 20 minutes," in *Proc. 14th Yale Workshop Adaptive Learn. Syst.*, vol. 95585, 2005, p. 1412.

[45] Z. Xu, J. Chen, and M. Tomizuka, "Guided policy search model-based reinforcement learning for urban autonomous driving," 2020, *arXiv:2005.03076*.

[46] M. P. Deisenroth and C. E. Rasmussen, *Reducing Model Bias in Reinforcement Learning*, Univ. Washington, Washington, DC, USA, Dec. 2010.

[47] A. S. Polydoros and L. Nalpantidis, "Survey of model-based reinforcement learning: Applications on robotics," *J. Intell. Robot. Syst. Theory Appl.*, vol. 86, no. 2, pp. 153–173, May 2017.

[48] R. S. Sutton and A. G. Barto, "Adaptive computation and machine learning series," in *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.