2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Adaptive Planning and Control with Time-Varying Tire Models for Autonomous Racing Using Extreme Learning Machine

Dvij Kalaria[1], Qin Lin[2], and John M. Dolan[1]

***Abstract*— Autonomous racing is a challenging problem, as the vehicle needs to operate at the friction or handling limits in order to achieve minimum lap times. Autonomous race cars require highly accurate perception, state estimation, planning, and control. Adding to this complexity is the need to accurately identify vehicle model parameters governing lateral tire slip effects, which can evolve over time due to factors such as tire wear and tear. Current approaches to this problem typically either propose offline model identification methods or rely on initial parameters within a narrow range (typically within 15-20% of the actual values). However, these approaches fall short in accounting for significant changes in tire models that can occur during actual races, particularly when pushing the vehicle to its handling limits. We present a unified framework that not only learns the tire model in real time from collected data but also adapts the model to environmental changes, even when the model parameters exhibit substantial deviations. The friction estimation, obtained as a byproduct from the learning results, facilitates the selection of the optimal racing line from a library for adaptive speed planning. We validate our approach through testing in simulators, encompassing a 1:43 scale race car and a full-size car, and also through experiments with a physical F1/10 autonomous race car.**

***Index Terms*— Learning-based control, adaptive motion planning and control, extreme learning machine, autonomous racing**

I. INTRODUCTION

The estimation of tire conditions is critical for the minimal lap times and safety of car racing [1]. Race drivers adapt their driving strategies during each lap [2], adjusting speeds at the turns based on their estimations of how well the tire traction will support them. Estimating the vehicle's handling capacity is particularly challenging for autonomous racing, owing to the multitude of variables present in an actual race, including temperature, weather conditions, and more.

We provide a summary of our comparison with other approaches in Table I. Traditional model-based planning and control frameworks for autonomous racing [3], [4], e.g., model predictive control (MPC), heavily rely on accurate vehicle dynamic models. However, achieving precise dynamics estimation, particularly concerning lateral tire forces, necessitates intensive system identification and verification processes involving interaction with the road material.

Numerous studies have focused on estimating the tire lateral friction given the slip angles [5]–[12]. Some of these research efforts involve conducting system identification to determine parameters of the Pacejka tire model [5] and the Fiala brush model [13]. [14] characterizes tire models using Neural Ordinary Differential Equations (NODEs). Additionally, certain studies employ more sophisticated neural networks to fit the tire curve [8]–[10], [15], as well as the entire vehicle dynamics [11], [12]. However, it's essential to note that the computational time required for these approaches can become a limiting factor when attempting to apply them in real-time for high-speed racing control.

[16] proposes an iterative procedure that uses data from previous laps to correct the vehicle model. [17] proposes using Gaussian process (GP) and linear MPC to enable learning vehicle parameters offline. [18], [19] proposes adjusting linear vehicle model online from collected data. But it assumes the availability of nearly-accurate tire parameters to start with and also all the methods assume the tire parameters to be constant throughout the run and not adapt to any changes. [16] proposes using an extended kinematic model and applies GP to learn the difference between the high-fidelity dynamic model and the extended kinematic model from collected data offline. However, using GP in optimization is computationally expensive (0.25s-0.3s according to [16]), which makes it practically infeasible to update online for an actual car. It is noteworthy that all of these works do not test on an online adaptive setting where the tire-surface interaction can change over time.

Methods	Tolerant to model errors	Adaptive planning to varied friction	Learning	Ref.
Traditional	✗	✗	-	[3], [4]
Learning -based	✓	✗	Offline	[16], [17] [19]
			Online	[18]
Ours	✓	✓	Online	-

TABLE I: Our motivation and comparison with others

An overview of our framework is illustrated in Fig. 1. The process begins with an offline optimization step to generate a library including minimum-curvature optimal racing lines for different surface frictions (see the blue block). Within our learning component (highlighted in red), we adopt a relatively simple kinematic *nominal model*. The discrepancy between the nominal model and the more complex dynamic model, which takes into account tire dynamics, aerodynamic forces, and longitudinal friction, is captured by an extreme learning machine (ELM). The significance of our learning is two-fold: 1) It facilitates model error compensation for the model-based controller (shown in the green block) utilizing the nominal model; 2) Our framework allows the retrieval of friction estimates from the learning results, thereby enabling

[1]The authors are with the Robotics Institute, Carnegie Mellon University dkalaria@andrew.cmu.edu,jdolan@andrew.cmu.edu

[2]Qin Lin is with the Computer Science Department, Cleveland State University q.lin80@csuohio.edu

the selection of the corresponding optimal racing line from the library for adaptive speed planning. This stands in contrast to existing methods that assume time-invariant friction.

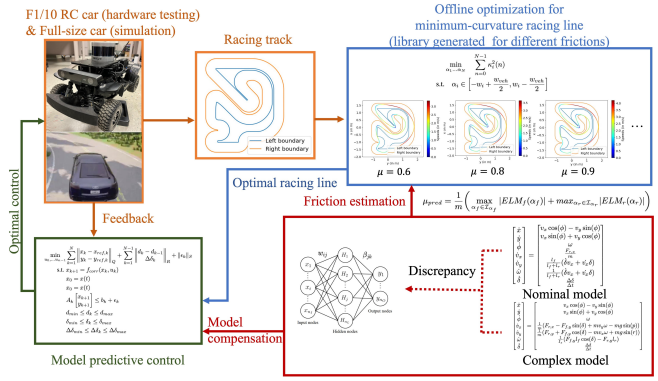


Fig. 1: Overview of our approach. Our approach consists of an online learning component marked in red with fits the residual between actual model and the nominal e-kinematic model and estimates the friction coefficient. The estimated friction coefficient is used to select the raceline that is used by the controller with the updated vehicle model to track.

We first demonstrate our results by using a 1:43-scale open-source numerical simulator, allowing for a comparison with a state-of-the-art approach [3].We then demonstrate in the high-fidelity Carla simulator, incorporating measurement noise and a sloped racetrack. Finally, we test our approach on a real F1/10 RC car.

The contributions of our work are summarized as follows:

- We propose a novel unified optimization framework for online model error compensation and online adaptation for autonomous racing. Our approach outperforms the state-of-the-art GP controller [16] in terms of lap time, safety, and computation efficiency.
- Unlike existing approaches that presume time-invariant friction, our ELM model estimates time-varying friction online to empower adaptive speed planning along the race line for safe racing.

The rest of the paper is organized as follows: Section II briefly goes through some preliminaries. Section III elaborates on our proposed method. Section IV reports the experimental results. In Section V, we make concluding remarks and discuss future directions.

II. PRELIMINARIES

A. Racing Line

Race drivers follow a racing line for specific maneuvers. This line can be used as a reference path by the motion planner to assign time-optimal trajectories while avoiding collision. The racing line is minimum-time or minimum-curvature. They are similar, but the minimum-curvature path additionally allows the highest cornering speeds given the maximum legitimate lateral acceleration [20].

In our work, we calculate the minimum-curvature optimal line, similar to [20]. The race track is represented by a sequence of tuples (x_i,y_i,w_i), $i \in \{0,...,N-1\}$, where (x_i,y_i) denotes the coordinate of the center location and w_i denotes the lane width at the i-th point. The output racing line consists of a tuple of seven variables: coordinates x and y, longitudinal displacement s, longitudinal velocity v_x, acceleration a_x, heading angle ψ, and curvature κ. It is obtained by minimizing the following cost:

$$\begin{aligned} &\min_{\eta_1 \dots \eta_N} \quad \sum_{n=0}^{N-1} \kappa_i^2(n) \\ &\text{s.t.} \quad \eta_i \in \left[-\frac{w_i}{2}+\frac{w_{veh}}{2}, \frac{w_i}{2}-\frac{w_{veh}}{2}\right] \end{aligned} \tag{1}$$

where the vehicle width is w_{veh}, and η_i is the lateral displacement with respect to the reference center line.

To create a velocity profile, we need to consider the vehicle's constraints on both longitudinal and lateral acceleration [20]. Our approach involves generating a library of velocity profiles, each tailored to specific lateral acceleration limits determined by the frictions for the front (μ_f) and rear (μ_r) tires, as well as the vehicle's mass (m) and the gravitational constant (g). In particular, we produce a set of velocity profiles covering a range of maximum lateral forces corresponding to the friction μ_{eff} within the interval $[\mu_{min}, \mu_{max}]$. This library allows us to retrieve a velocity profile that matches a given value of μ. Interpolation is necessary when we encounter a friction value that falls within the valid range but is not explicitly present in the library.

B. Vehicle models

1) Dynamic model: The dynamic model has states x, y, and ϕ in the global frame; longitudinal velocity v_x, lateral velocity v_y, and yaw angular velocity ω in the vehicle's body frame; and steering angle δ. For the dynamic model, $F_{r,x}$ is the longitudinal force on the rear tire in the tire frame assuming a rear-driven vehicle, $F_{f,y}$ and $F_{r,y}$ are the forces on the front and rear tires, respectively, and α_f and α_r are the corresponding slip angles. r and p are the roll and pitch respectively of the vehicle. We denote the mass of the vehicle m, the moment of inertia in the vertical direction about the center of mass of the vehicle I_z, the length of the vehicle from the COM (center of mass) to the front wheel l_f, and the length from the COM to the rear wheel l_r. $B_{f/r}$, $C_{f/r}$, $D_{f/r}$ are the Pacejka tire model parameters specific to the tire and track surface. C_{m1} and C_{m2} are longitudinal parameters that dictate the longitudinal force generated from the vehicle given the throttle command.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\phi} \\ \dot{v}_x \\ \dot{v}_y \\ \dot{\omega} \\ \dot{\delta} \end{bmatrix} = \begin{bmatrix} v_x\cos(\phi) - v_y\sin(\phi) \\ v_x\sin(\phi) + v_y\cos(\phi) \\ \omega \\ \frac{1}{m}(F_{r,x} - F_{f,y}\sin(\delta) + mv_y\omega - mg\sin(p)) \\ \frac{1}{m}(F_{r,y} + F_{f,y}\cos(\delta) - mv_x\omega + mg\sin(r)) \\ \frac{1}{I_z}(F_{f,y}l_f\cos(\delta) - F_{r,y}l_r) \\ \frac{\Delta\delta}{\Delta t} \end{bmatrix} \tag{2}$$

where $F_{r,x} = (C_{m1} - C_{m2}v_x)d - C_{lf} - C_d v_x^2$, $F_{f,y} = D_f \sin(C_f \tan^{-1}(B_f\alpha_f))$, $\alpha_f = \delta - \tan^{-1}\left(\frac{\omega l_f + v_y}{v_x}\right)$, and $F_{r,y} = D_r \sin(C_r \tan^{-1}(B_r\alpha_r))$, $\alpha_r = \tan^{-1}\left(\frac{\omega l_r - v_y}{v_x}\right)$.

2) Extended kinematic model: The extended kinematic model (**E-kinematic model**) [16] extends the commonly used kinematic bicycle model from 4-DOF to 7-DOF to make the dimension consistent with the dynamic model, i.e., v_x, v_y and w are not in the kinematic model. Also, slip angles α_f and α_r are assumed to be 0 in the kinematic model. It is derived just by having $\dot{\alpha}_f = 0$ and $\dot{\alpha}_r = 0$. For longitudinal forces, the effect of aerodynamic forces is ignored.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\phi} \\ \dot{v}_x \\ \dot{v}_y \\ \dot{\omega} \\ \dot{\delta} \end{bmatrix} = \begin{bmatrix} v_x \cos(\phi) - v_y \sin(\phi) \\ v_x \sin(\phi) + v_y \cos(\phi) \\ \omega \\ \frac{F_{r,x}}{m} \\ \frac{l_f}{l_f + l_r}(\dot{\delta} v_x + \dot{v_x}\delta) \\ \frac{1}{l_f + l_r}(\dot{\delta} v_x + \dot{v_x}\delta) \\ \frac{\Delta\delta}{\Delta t} \end{bmatrix} \tag{3}$$

where $F_{r,x} = (C_{m1} - C_{m2} v_x)d$. This model extends the dimension of the kinematic model to obtain the same number of states as the dynamic model. In our work, the E-kinematic model will serve as a **nominal model**. The discrepancy between the dynamic model and the nominal model will be learned and compensated by our learning component.

C. Extreme Learning Machine

For modeling tire friction curves, we use extreme learning machines (ELMs). An architecture of a single-layer ELM is shown in Fig. 2. Let the number of input nodes be n_I, the number of hidden nodes n_L, and the number of output nodes n_O. The mapping from input and output is:

$$\mathbf{y} = \mathbf{H}(\mathbf{x})\beta = g(\mathbf{w}^T \mathbf{x})\beta \tag{4}$$

where $\mathbf{H} = [H_1, H_2, \cdots, H_j, \cdots, H_{n_L}]$ is a nonlinear transformation, $\mathbf{w} = [w_1, \cdots, w_{n_I}]$ are input weights, and $\beta = [\beta_1, \cdots, \beta_{n_L}]$ are output weights. Furthermore, $g(\cdot)$ corresponds to a kernel function that fulfills the condition for universal approximation [21].

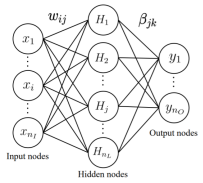


Fig. 2: Architecture of a single layer ELM.

For training, let us consider the dataset to be of size N with each element of the form $(\mathbf{x}_n, \mathbf{t}_n)$, where $\mathbf{t}_i \in \mathbb{R}^{n_O}$ is the true label corresponding to the i-th datapoint. ELM has a fixed input weight denoted as $\hat{\mathbf{w}}$, unlike a traditional feedforward single-layer neural network. The *first reason* for using ELM instead of a single-layer neural network is that the training of ELM is more efficient [22]: if we initialize input weight $\mathbf{w}$ with random values drawn from a distribution and keep them fixed, it can theoretically get the same results even if it is not optimized. The objective of ELM is to obtain the optimal β^* which satisfies the following objective:

$$\beta^* = \arg\min_{\beta} \mathcal{L}\left(\mathbf{y}(\beta), \mathbf{t}\right) = \arg\min_{\beta} \|\mathbf{H}(\mathbf{x})\beta - \mathbf{t}\|_2 \tag{5}$$

In our case, we use separate ELMs for rear and front tire slip-curve estimation. Here the inputs are the slip angle $\alpha_{f|r}$ and unity 1 to allow adding bias on the first linear layer output, thus $n_I = 2$ and the output is the lateral force, $F_{[f|r]y}$, thus $n_o = 1$. n_h is a hyperparameter which we choose as 40 for the experiments. We will discuss further in Sec. III-A.2 how we pass the gradient signal across both front and rear tire ELMs.

III. METHODOLOGY

We start our pipeline with an E-kinematic model as the nominal model $f_{\text{e-kin}}$. Later, through the collected data, we get the residual error f_{err}, i.e., $f_{\text{corr}} = f_{\text{e-kin}} + f_{err}$. The learnable parameters here are the ELM tire slip models, longitudinal friction C_{lf}, and aerodynamic force constant C_d.

A. Learning based control

1) Data collection: The collected data can be represented as a pair of states from consecutive time steps, i.e., as $\{\mathbf{x}_k, \mathbf{u}_k, \mathbf{x}_{k+1}\}$. Hence, the error between the actual state x_{k+1} and the one estimated from the discretized E-kinematic model (using 6^{th}-order Runge-Kutta) can be represented as:

$$e(\mathbf{x}_k, \mathbf{u}_k) = \mathbf{x}_{k+1} - f_{\text{e-kin}}(\mathbf{x}_k, \mathbf{u}_k) \tag{6}$$

We collect the data online, hence the dataset at the n-th time step is: $D_n = \{e(\mathbf{x}_k, \mathbf{u}_k) | k \in \{1, 2, .., n-1\}\}$.

2) Training and online adaptation: Our goal is to learn the model discrepancy, $e(\mathbf{x}_k, \mathbf{u}_k)$. The predicted value $e_{\text{pred}}(\mathbf{x}_k, \mathbf{u}_k)$ is obtained by computing the difference between (2) and (3):

$$\begin{aligned} \begin{bmatrix} e_{\dot{v}_x} \\ e_{\dot{v}_y} \\ e_{\dot{\omega}} \end{bmatrix}_{\text{pred}} &= \begin{bmatrix} \frac{1}{m}(-C_{lf} - C_d v_x^2 - F_{f,y}\sin(\delta) + m v_y \omega) \\ \frac{1}{m}(F_{r,y} + F_{f,y}\cos(\delta) - m v_x \omega) \\ \frac{1}{I_z}(F_{f,y} l_f \cos(\delta) - F_{r,y} l_r) \end{bmatrix} \\ &- \begin{bmatrix} m\sin(p) \\ -m\sin(r) + \frac{l_f}{l_f + l_r}(\dot{\delta} v_x + \dot{v_x}\delta) \\ \frac{1}{l_f + l_r}(\frac{\Delta\delta}{\Delta t} v_x + \dot{v_x}\delta) \end{bmatrix} \end{aligned} \tag{7}$$

where $F_{f,y} = ELM_f(\alpha_f)$, $F_{r,y} = ELM_r(\alpha_r)$.

The loss to be minimized is defined as:

$$\begin{aligned} L = \sum_{k \in \{i,..,j\}} \Big(& (e_{\dot{v}_x}(\mathbf{x}_k, \mathbf{u}_k) - e_{\dot{v}_x, pred}(\mathbf{x}_k, \mathbf{u}_k))^2 \\ & + (e_{\dot{v}_y}(\mathbf{x}_k, \mathbf{u}_k) - e_{\dot{v}_y, pred}(\mathbf{x}_k, \mathbf{u}_k))^2 \\ & + (e_{\dot{\omega}}(\mathbf{x}_k, \mathbf{u}_k) - e_{\dot{\omega}, pred}(\mathbf{x}_k, \mathbf{u}_k))^2 \Big) \end{aligned} \tag{8}$$

We minimize L iteratively using stochastic gradient descent with momentum. We use a mini-batch with size K_{batch} for a robust update. The training cycle can be briefly visualized as in Fig. 3. Letting the update rate be γ_{lr} and the momentum be γ_p, the model parameters are updated as:

$$\begin{aligned} p &= \begin{bmatrix} C_{lf} & C_d & \beta_{\text{elm},f}^T & \beta_{\text{elm},r}^T \end{bmatrix}^T \\ v(k+1) &= \gamma_p v(k) + (1 - \gamma_p)\frac{\partial L}{\partial p} \\ p(k+1) &= p(k) - \gamma_{lr} v(k) \end{aligned} \tag{9}$$

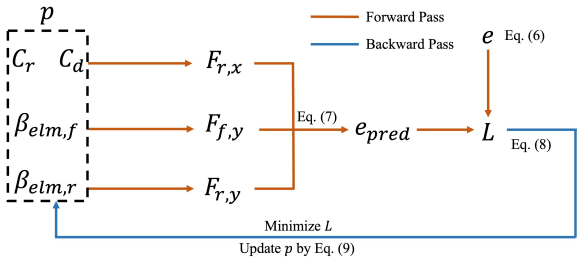


Fig. 3: Training cycle. In the forward pass, given C_{lf}, C_d, and two ELMs, we calculate forces, then a residual error can be obtained to further update the model in the backward pass.

3) Adaptive speed planning: In addition to updating the vehicle model, it is crucial to adjust the reference speeds along the race line. For instance, if the friction coefficient decreases, the vehicle's ability to navigate turns successfully is compromised, necessitating a reduction in the reference speeds. To achieve this, we first create a speed profiles library for varied frictions, as elaborated in Section II-A. We then employ the estimated friction for adaptive speed planning, dynamically assigning reference speeds in real time based on the varying friction conditions. We use the traction circle assumption as described in [23] for raceline calculation, where the friction coefficient dictates the maximum ratio of the force exerted to the normal force. We obtain the predicted friction coefficient as the ratio of the maximum force exerted to the available normal force $N = mg$. As we predict the tire slip vs. force curve, we can obtain maximum force as the maximum magnitude of predicted values from the ELM with slip angle inputs ranging from the observed slip angle values. This is inline with what we use for raceline calculation, as described in [23] (cf. CH. 3.1.3).

The predicted friction coefficient is calculated as:

$$\mu_{\text{pred}} = \frac{1}{mg}\left(\max_{\alpha_f \in \mathcal{I}_{\alpha_f}} |ELM_f(\alpha_f)| + \max_{\alpha_r \in \mathcal{I}_{\alpha_r}} |ELM_r(\alpha_r)|\right) \tag{10}$$

where $\mathcal{I}_{\alpha_f} = [\alpha_{f,min}, \alpha_{f,max}]$, $\mathcal{I}_{\alpha_r} = [\alpha_{r,min}, \alpha_{r,max}]$, $\alpha_{(f/r),min}$ and $\alpha_{(f/r),max}$ are the min and max values of the slip angles observed at any of the tires.

The optimization for control is formulated as follows:

$$\begin{aligned}
\min_{u_0,\dots u_{n-1}} \quad & \sum_{k=1}^{N} \left\| \begin{matrix} \mathbf{x}_{k,x} - x_{ref,k} \\ \mathbf{x}_{k,y} - y_{ref,k} \end{matrix} \right\|_Q + \sum_{k=1}^{N-1} \left\| \begin{matrix} d_k - d_{k-1} \\ \Delta\delta_k \end{matrix} \right\|_R + \|\epsilon_k\|_S \\
\text{s.t. } & \mathbf{x}_{k+1} = f_{\text{corr}}(\mathbf{x}_k, \mathbf{u}_k) \\
& \mathbf{x}_0 = \mathbf{x}(t) \\
& A_k \begin{bmatrix} \mathbf{x}_{k+1,x} \\ \mathbf{x}_{k+1,y} \end{bmatrix} \leq b_k + \epsilon_k \\
& d_{min} \leq d_k \leq d_{max} \\
& \delta_{min} \leq \delta_k \leq \delta_{max} \\
& \Delta\delta_{min} \leq \Delta\delta_k \leq \Delta\delta_{max}
\end{aligned} \tag{11}$$

where A_k and b_k are the tangent line coefficient matrices for the left and the right track boundaries at the nearest point from the vehicle's position. The reference trajectory is obtained by sampling points along the target racing line at distances according to reference velocities at those points, similar to [16]. The positive semi-definite matrices Q, R, and S characterize the tracking, actuation, and slack penalties, respectively. Slack variable ϵ_k is introduced to prevent infeasibility for the safety constraint, i.e., linearizing the track boundary at the nearest point. $f_{\text{corr}} = f_{\text{e-kin}} + e_{\text{pred}}$ is the corrected model, which outputs the next state. $f_{\text{corr}} = f_{\text{e-kin}}$ for $t \leq N$ when data are limited. Beyond N, we will adopt an online update for the model. $d_{min}, d_{max}, \delta_{min}, \delta_{max}$ are the actuation limits and $\Delta\delta_{min}, \Delta\delta_{max}$ are the steering rate limits.

IV. Experimental Results

We perform experiments on the 1:43 numeric simulator, the Carla high-fidelity vehicle simulator [24], and a real 1/10-scale RC car. Additional results are presented on a webpage[1] due to limited space.

A. Numeric simulator

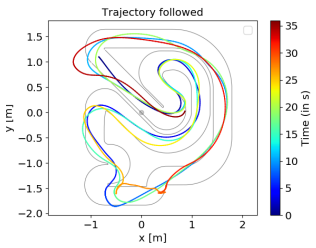
We use the 1:43 autonomous open-source race car simulation [3] to validate our algorithm. We demonstrate our results on the ETHZ track [3] and the ETHZMobil track [16].

1) Constant friction decay: We compare path-following performance without and with the use of adaptive model compensation in Fig. 4a and Fig. 4c. We modify the actual μ, i.e., change parameters D_f and D_r, as shown in Fig. 4d, and the predicted values as well. As can be seen, using our approach with adaptive model compensation clearly yields better performance. Using adaptive speed planning also allows the vehicle to stay within track boundaries. If speed references are not changed (see Fig. 4b), the vehicle is still able to complete its maneuvers, but leaves the track several times. This is because even with a compensated model, unadjusted reference speeds push the vehicle beyond its limits at turns. Not updating the vehicle model leads to the vehicle's completely failing and losing control at a region on the 3^{rd} lap. Finally, we also compare with an oracle (Fig. 5b), where we use the ground truth Pacejka friction parameters. As can be observed, the trajectory followed with our method (Fig. 4c) and the lap times (shown in Table II) are almost the same as those of the oracle (Fig. 5b).

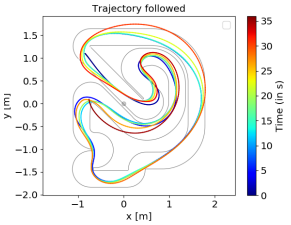
Moreover, we also compare against the use of adaptive Gaussian processes [16], where we adjust the speeds according to the ground truth value of μ (see Fig. 4d), and this yields slightly worse (very close) results than ours but with much higher computation times. Also, as can be observed from Fig. 4c and Fig. 5a, the trajectory followed when using GP goes further away from the reference with decreasing μ while with our method, the trajectory remains nearly the same similar to the oracle (see Fig. 5b). It should also be noted that using GP requires a separate μ prediction process to replace and obtain an estimate close to the ground truth value, while in our method we inherently obtain the predicted value of μ according to (10).

[1] https://dvij542.github.io/apacrace/

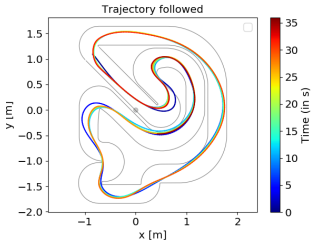
We summarize run statistics in Table II. As can be seen: 1) our lap time is slightly worse than the oracle but the safety (i.e., the violation time) has been significantly improved; 2) the mean deviation from the racing line is least when our method is used with adjusted speeds (close to the oracle); 3) our approach outperforms the state-of-the-art GP-based controller [16] in terms of every criterion as well as the computation efficiency. The computation times for our method are within $0.05s \pm 0.02s$ and for GP is $0.30s \pm 0.05s$ when used online on our machine (AMD Ryzen 7 5000 series CPU, 16 GB RAM).



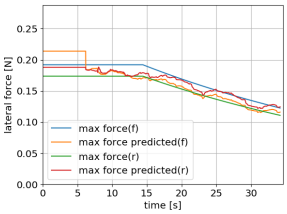
(a) W/o model compensation



(b) W/ compensation, w/o adaptive speed

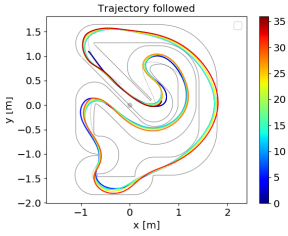


(c) Ours

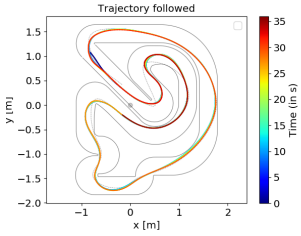


(d) Max force predictions

Fig. 4: Comparison for w/o and w/ learning on numeric sim



(a) Using GP [16]



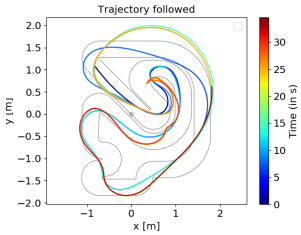
(b) Using ground truth dynamics

Fig. 5: Comparison with other approaches on numeric sim

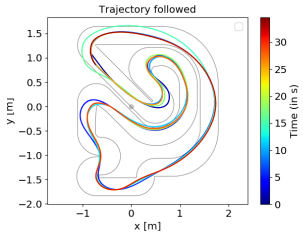
2) Sudden friction change: We test our approach in a situation where the friction coefficient suddenly changes. This is possible, for example, when a sudden heavy rain changes the road's friction coefficient. For this case, we demonstrate the path followed by our approach (See Fig. 6b) and compare it to when the model compensation is not considered (See Fig. 6a). As observed, with model compensation, the vehicle is able to estimate the change in friction coefficient quite well and rapidly. It does cross the outer boundary at the beginning but is able to adjust the speeds to stay safe within the lane limits later, whereas if compensation is not considered, the vehicle completely loses control at several points after μ changes.

Method	Lap 0 time	Lap 1 time	Lap 2 time	Lap 3 time	Mean dev. from racing line	Violation time	Avg. cycle computation time
Without adaptation	8.44s/ 6.74s	8.34s/ 6.08s	8.72s/ 6.50s	- / 7.42s	0.141m/ 0.134m	4.32s/ 4.14s	**0.02s**
With adaptation (Constant speeds)	8.26s/ 6.64s	**7.26s/ 5.78s**	**7.56s/ 5.92s**	**8.14s/ 6.40s**	0.1104m/ 0.125m	2.84s/ 2.66s	0.06s
With adaptation (Var speeds, **ours**)	**8.26s/ 6.64s**	7.42s/ 5.84s	7.76s/ 6.1s	8.46s/ 6.46s	**0.0832m/ 0.103m**	**0.46s/ 1.14s**	0.06s
GP	8.28s/ 6.64s	7.44s/ 5.90s	7.84s/ 6.32s	8.56s/ 6.62s	0.0989m/ 0.115m	0.62s/ 2.02s	0.32s
Oracle	*7.92s/ 6.28s*	*7.40s/ 5.82s*	*7.72s/ 6.02s*	*8.40s/ 6.38s*	*0.0511m/ 0.042m*	*0s/ 0.12s*	0.03s

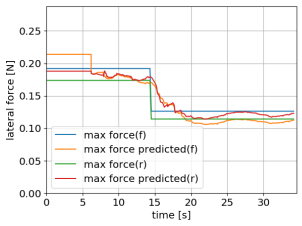
TABLE II: Track 1 (ETHZ track [3])/Track 2 (ETHZMobil track [16]). Comparison in statistics for numeric simulator. Violation time is the total time the vehicle was outside the track limits.



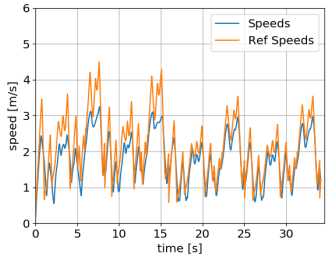
(a) W/o model compensation



(b) Ours



(c) Max friction force predictions for w/ adaptation

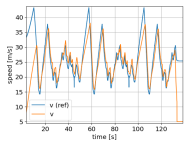


(d) Speeds w/ adaptation

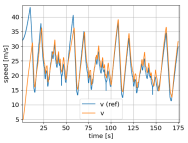
Fig. 6: Online run comparison with model compensation & adaptive speed planning vs. if none are used (sudden friction change experiment)

B. Carla

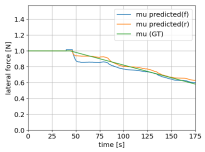
We perform experiments in Carla [24] with time-varying tire friction parameters to test our online adaptation algorithm. The online run with friction decay is shown in Fig. 7c. Both tires have the same tire model. For comparison, we show the results with and without adaptive model compensation (See Fig. 7b and Fig. 7a). Without adaptive model compensation, the vehicle moves out of bounds after about $60s$ once the friction coefficient starts reducing and collides with an obstacle. Due to limited space, more results are provided on the webpage.



(a) Speeds w/o compensation



(b) Speeds, ours



(c) μ predictions

Fig. 7: Online run with model compensation in Carla

C. Hardware experiments in an RC car

We test on a real F1/10 RC car (see Fig. 8) using Vicon [25] for localization.

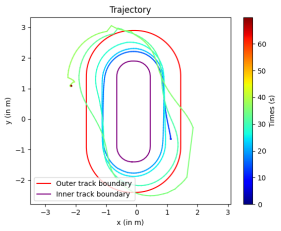
1) Experiment setup: We perform the experiment on a track with a lane width of 0.5 m. The reference line is taken to be the centerline with limiting speeds rendered according to [20]. We conduct the experiment with an initial friction coefficient 1.5 and incrementally increase target speeds from 50% to 100% of the reference speeds from $t = 10s$ to $t = 35s$. This is to ensure physical safety and also allow sufficient time and data for the adaptation algorithm.

2) Results: The final trajectories for a rounded rectangle reference are shown in Fig. 9a and Fig. 9b for the comparison between w/o model and reference speed adaptation vs. ours. As can be clearly observed, without model adaptation, the vehicle goes outside the track and the car physically hits the wall, while with adaptation, the algorithm is able to correctly estimate the friction coefficient of the surface online (see Fig. 9c) and adjust the vehicle model and speeds accordingly. In the beginning, the car does exceed the track boundaries, but is able to effectively use this experience to recover quite well from its subsequent laps. For the experiment with model adaptation, we stop the car at $t = 70s$ and change the surface along the track from a rough original surface to a smoother surface obtained by placing wooden slabs on it. Then, we restart the same experiment but now on this new surface *starting with the previous trained model for testing the adaptation and transition*. The adaptation algorithm is able to reasonably adapt to the new friction parameters and estimate the friction coefficient to be around 15% less than the earlier predicted value of the original surface (see Fig. 10b). The trajectory for the new surface is shown in Fig. 10a. The statistical results are tabulated in Table III. The website displays the results of another, diamond-shaped track with consistent conclusions, as space limits the presentation.

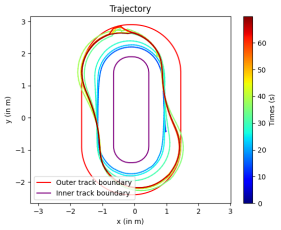
Fig. 8: RC car

V. Conclusion and Future Work

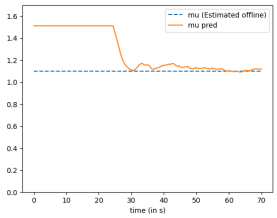
We introduce a novel learning-based control algorithm that not only significantly reduces the effort required for system identification but also enables real-time adaptation. Our approach commences with a simple geometric model and collects training data, which the algorithm utilizes iteratively to compensate discrepancy between the nominal and actual vehicle models. It adapts dynamically to any changes in the system. This capability empowers racing algorithms to craft aggressive strategies without the need for constant concern about vehicle model parameters and tire conditions. We showcase the algorithm's robust performance in scenarios featuring model uncertainty and simulated friction variations, leveraging both simulators and a physical testbed. An interesting future direction involves guiding the training process effectively within a reinforcement learning framework. This could push the vehicle beyond its known limits, enabling the exploration of these limits and enhancing the model's confidence in them. An approach akin to adaptive ϵ-greedy Q-learning [26] could be employed, with ϵ dynamically adjusted based on the vehicle's current safety index.



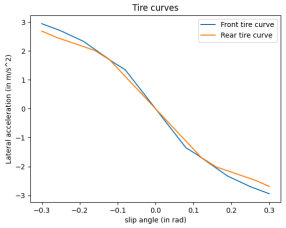
(a) W/o model compensation, w/o adaptive speed



(b) Ours

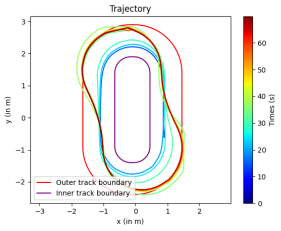


(c) μ estimates

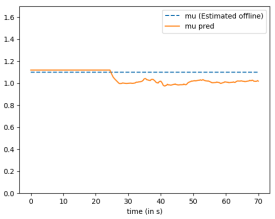


(d) Trained tire curves

Fig. 9: RC car experiment on rounded rectangle trajectory



(a) Ours



(b) μ estimates

Fig. 10: Adaptation to new surface

	Avg. lap time (Last 3 laps)	Mean deviation from reference	Violation time
Without adaptation	5.4s	0.52m	35.6s
With adaptation	**5.05s**	**0.28m**	**2.1s**
With adaptation (new surface)	5.08s	0.29m	7.0s

TABLE III: Statistics for w/ and w/o model adaptation

REFERENCES

[1] W. J. West and D. J. Limebeer, "Optimal tyre management of a formula one car," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 14 456–14 461, 2020, 21st IFAC World Congress.

[2] A. Bonomi, E. Turri, and G. Iacca, "Evolutionary f1 race strategy," *Proceedings of the Companion Conference on Genetic and Evolutionary Computation*, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:260119494

[3] A. Liniger, A. Domahidi, and M. Morari, "Optimization-based autonomous racing of 1:43 scale RC cars," *Optimal Control Applications and Methods*, vol. 36, pp. 628 – 647, 2015.

[4] J. Kabzan, M. de la Iglesia Valls, V. Reijgwart, *et al.*, "AMZ driverless: The full autonomous racing system," *Journal of Field Robotics*, vol. 37, pp. 1267 – 1294, 2019.

[5] H. B. Pacejka, "Tire and vehicle dynamics." Elsevier, 2005.

[6] J. Svendenius and M. Gäfvert, "Construction of novel semi-empirical tire models for combined braking and cornering," *Technical Reports; TFRT*, vol. 7606, 2003.

[7] J. Svendenius, "Tire modeling and friction estimation," Ph.D. dissertation, Lund University, 2007.

[8] P. Guarneri, G. Rocca, and M. Gobbi, "A neural-network-based model for the dynamic simulation of the tire/suspension system while traversing road irregularities," *IEEE Transactions on Neural Networks*, vol. 19, pp. 1549–1563, 2008.

[9] N. Xu, H. R. Askari, Y. Huang, J. Zhou, and A. Khajepour, "Tire force estimation in intelligent tires using machine learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, pp. 3565–3574, 2020.

[10] J. Matuško, I. Petrović, and N. Perić, "Neural network based tire/road friction force estimation," *Engineering Applications of Artificial Intelligence*, vol. 21, no. 3, pp. 442–456, 2008.

[11] S. Srinivasan, I. Sa, A. Zyner, V. Reijgwart, M. de la Iglesia Valls, and R. Y. Siegwart, "End-to-end velocity estimation for autonomous racing," *IEEE Robotics and Automation Letters*, vol. 5, pp. 6869–6875, 2020.

[12] N. A. Spielberg, M. Brown, N. R. Kapania, J. C. Kegelman, and J. C. Gerdes, "Neural network vehicle models for high-performance automated driving," *Science Robotics*, vol. 4, 2019.

[13] E. Fiala, "Seitenkraften am rollenden luftreifen," *VdI*, vol. 96, pp. 973–979, 1954.

[14] F. Djeumou, J. Y. Goh, U. Topcu, and A. Balachandran, "Autonomous drifting with 3 minutes of data via learned tire models," in *2023 IEEE International Conference on Robotics and Automation*, 2023, pp. 968–974.

[15] M. Acosta and S. Kanarachos, "Tire lateral force estimation and grip potential identification using neural networks, extended Kalman filter, and recursive least squares," *Neural Computing and Applications*, vol. 30, pp. 3445–3465, 2018.

[16] A. Jain, M. O'Kelly, P. Chaudhari, and M. Morari, "BayesRace: Learning to race autonomously using prior experience," in *Proceedings of the 4th Conference on Robot Learning*, 2020.

[17] L. Hewing, A. Liniger, and M. N. Zeilinger, "Cautious NMPC with Gaussian process dynamics for autonomous miniature race cars," *2018 European Control Conference*, pp. 1341–1348, 2017.

[18] U. Rosolia and F. Borrelli, "Learning how to autonomously race a car: A predictive control approach," *IEEE Transactions on Control Systems Technology*, vol. 28, pp. 2713–2719, 2019.

[19] ——, "Learning model predictive control for iterative tasks. a data-driven control framework," *IEEE Transactions on Automatic Control*, vol. 63, pp. 1883–1896, 2016.

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

[21] G.-B. Huang, L. Chen, C. K. Siew, *et al.*, "Universal approximation using incremental constructive feedforward networks with random hidden nodes," *IEEE Transactions on Neural Networks*, vol. 17, no. 4, pp. 879–892, 2006.

[22] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1-3, pp. 489–501, 2006.

[23] T. SJÖLIN and A. SUNDBERG, "Trajectory planning and control of an autonomous race vehicle," Ph.D. dissertation, KTH Royal Institute Of Technology School Of Industrial Engineering and Management, Stockholm, Sweden, June 2021.

[24] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.

[25] P. Merriaux, Y. Dupuis, R. Boutteau, P. Vasseur, and X. Savatier, "A study of vicon system positioning performance," *Sensors*, vol. 17, no. 7, 2017.

[26] A. dos Santos Mignon and R. L. de Azevedo da Rocha, "An adaptive implementation of ϵ-greedy in reinforcement learning," *Procedia Computer Science*, vol. 109, pp. 1146–1151, 2017, 8th International Conference on Ambient Systems, Networks and Technologies, ANT-2017.