Meta-Learning Online Dynamics Model Adaptation in Off-Road Autonomous Driving

Jacob Levy*, Jason Gibson†, Bogdan Vlahov†, Erica Tevere‡, Evangelos Theodorou†,
David Fridovich-Keil*, Patrick Spieler‡
*University of Texas at Austin †Georgia Institute of Technology
‡Jet Propulsion Laboratory, California Institute of Technology

***Abstract*—High-speed off-road autonomous driving presents unique challenges due to complex, evolving terrain characteristics and the difficulty of accurately modeling terrain-vehicle interactions. While dynamics models used in model-based control can be learned from real-world data, they often struggle to generalize to unseen terrain, making real-time adaptation essential. We propose a novel framework that combines a Kalman filter-based online adaptation scheme with meta-learned parameters to address these challenges. Offline meta-learning optimizes the basis functions along which adaptation occurs, as well as the adaptation parameters, while online adaptation dynamically adjusts the onboard dynamics model in real time for model-based control. We validate our approach through extensive experiments, including real-world testing on a full-scale autonomous off-road vehicle, demonstrating that our method outperforms baseline approaches in prediction accuracy, performance, and safety metrics, particularly in safety-critical scenarios. Our results underscore the effectiveness of meta-learned dynamics model adaptation, advancing the development of reliable autonomous systems capable of navigating diverse and unseen environments.**

I. Introduction

High-speed off-road autonomous driving presents a unique set of challenges where precise and reliable control is essential for traversing complex and unseen environments. These settings are characterized by diverse terrain types such as sand, snow, and dense vegetation, as well as varying terrain conditions like wetness, deformability, and roughness. Such variability can significantly alter vehicle dynamics, introducing substantial uncertainties [13, 1, 25, 20]. Just as a human driver adjusts their driving policy based on how the vehicle responds to the terrain, autonomous systems must adapt dynamically to maintain both performance and safety.

Autonomous systems are increasingly relied upon in scenarios where human intervention is impractical, too slow, or dangerous. For instance, Mars rovers operate with onboard autonomy to identify obstacles and plan safe paths, as communication delays prevent real-time teleoperation [26, 23]. On Earth, autonomous vehicles hold significant potential for disaster response, where hazardous environments could render human operation unsafe [15, 16, 18]. Similarly, in mining and resource extraction, they can transport materials across rugged and hazardous terrains, improving operational efficiency and worker safety [3]. Safety is paramount in these scenarios, as entering dangerous zones or tipping the vehicle over could have severe consequences for mission success or operational integrity.

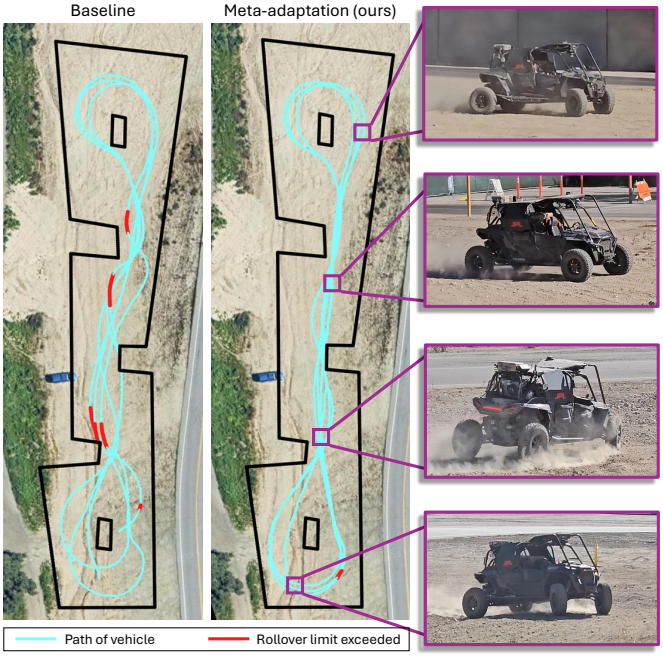

Fig. 1: Trajectories for a single 3-lap run, with insets displaying video stills. The baseline configuration shows erratic trajectories with frequent course boundary and rollover limit violations. In contrast, our adaptation configuration demonstrates more deliberate and compliant trajectories as the car learns the terrain dynamics in real time.

When system dynamics are known, model-based control techniques are widely used for the effective control of autonomous systems. For instance, model predictive path integral (MPPI) control [28] is an algorithm that rolls out sampled control inputs on a dynamics model to identify optimal trajectories. This method has demonstrated significant success in off-road autonomous driving applications [4, 12, 17, 9, 21, 27], but its performance hinges on the accuracy of the underlying dynamics model. Accurately modeling interactions between vehicles and terrain remains a significant challenge, particularly in high-speed off-road driving, where changing terrain and operational conditions push the vehicle to its performance limits. When the dynamics model fails to capture these interactions, it can lead to degraded performance and compromised safety.

To address these limitations, hybrid approaches have emerged, integrating first-principles dynamics models with

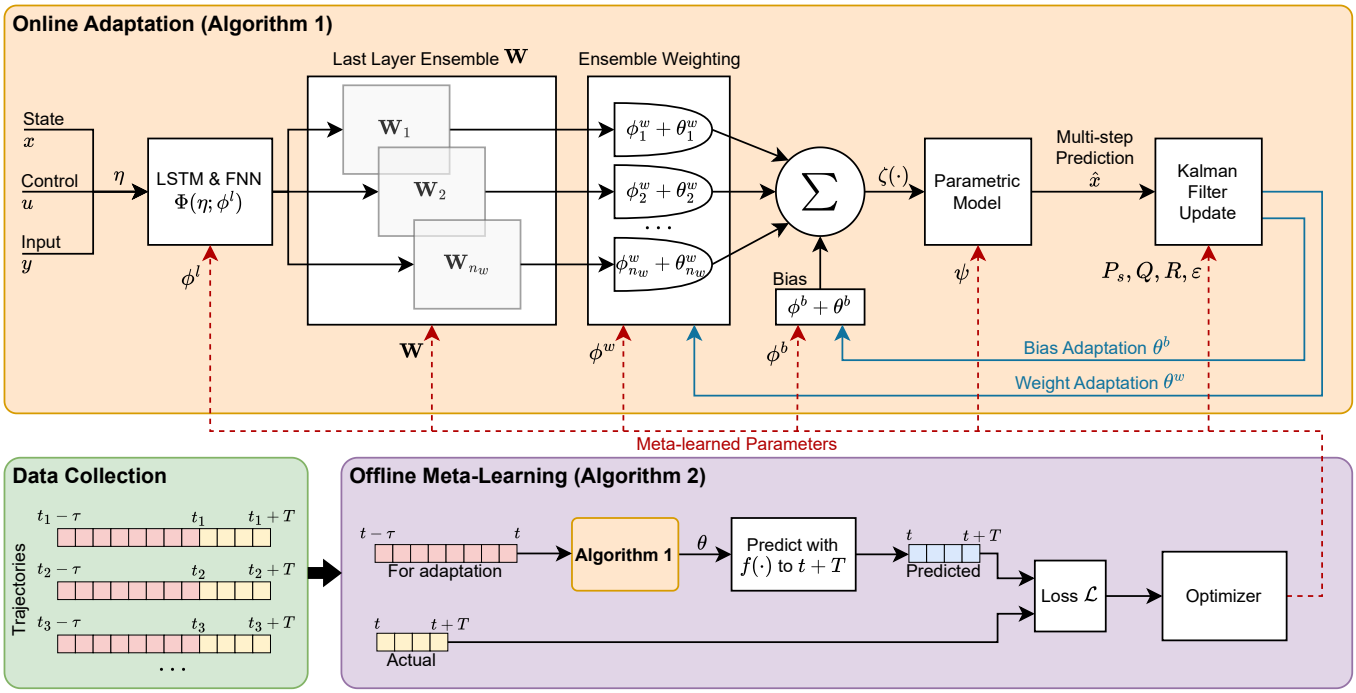

Fig. 2: Meta-learning online dynamics model adaptation. Online, a Kalman filter updates the linear combination of an ensemble of last-layer weights (Algorithm 1). Offline, trajectory segments are used to meta-learn model parameters, the last-layer ensemble, and the Kalman filter parameters (Algorithm 2).

learned components to improve predictive accuracy [5, 14, 6, 9, 19]. While these models are effective under nominal conditions, they can struggle to generalize to previously unseen terrains or adapt to evolving dynamics during operation. This limitation underscores the need for adaptive models capable of online adjustment.

We propose a novel framework (Fig. 2) for meta-learned online dynamics model adaptation designed to address these challenges. By leveraging meta-learning techniques, our method enables efficient and effective adaptation of the dynamics model to real-time sensor data. This approach allows the system to adjust to changes in terrain and vehicle behavior dynamically, providing accurate predictions essential for safe and efficient control. Our contributions include:

- A meta-learning framework for *offline* optimization of adaptation basis functions and parameters, along with dynamics model parameters.
- An efficient Kalman filter-based *online* adaptation scheme to update model parameters in real time, addressing the challenges of noisy and delayed state measurements.
- Empirical validation of our method on a full-scale autonomous off-road vehicle and simulated environments, showcasing improved performance and safety metrics over baseline approaches.

II. Preliminaries

A. Vehicle Dynamics

We model the vehicle dynamics according to [11, 10]; a brief overview is presented here. The dynamics are a discrete-time nonlinear system of the form

$$x_{t+1} = f(x_t, u_t, y_t), \tag{1}$$

where $u_t \in \mathbb{R}^3$ are control inputs (throttle, brake, and steering) and $y_t \in \mathbb{R}^{14}$ are sensor observations (e.g. roll, pitch, surface normals). The state of the vehicle $x_t = [p_t, v_t, z_t] \in \mathbb{R}^{10}$, consists of position $p_t \in \mathbb{R}^3$, velocity $v_t \in \mathbb{R}^3$, and actuator states $z_t \in \mathbb{R}^4$. The acceleration $\dot{v}_t$ is predicted with a hybrid parametric and learned model:

$$\dot{v}_t = g(x_t, u_t; \psi) + \zeta(\eta_t; \phi, \theta_t) \tag{2}$$

where $g(\cdot)$ is a physics-based parametric model with parameters ψ and $\zeta(\cdot)$ contains a long short-term memory (LSTM) network and a feedforward neural network (FNN) with learned parameters ϕ, adaptable parameters θ_t, and measurements η_t.

B. Control Architecture

We use Model Predictive Path Integral Control (MPPI) as specified in [11] to estimate the optimal control sequence $\mathbf{u}_t^* = \{u_t, \ldots, u_{t+T}\}$. The cost function balances several objectives critical for safe and efficient off-road driving, including strong penalties for leaving the track and exceeding rollover safety limits. To assess rollover risk, we estimate the ratio of loading between the left- and right-side tires; significant imbalance indicates a higher likelihood of tipping, and is penalized accordingly.

III. Meta-learning Model Adaptation

In this section, we present our approach (Fig. 2), detailing how we design the adaptable components of the dynamics

model, the online scheme for real-time parameter updates, and the meta-learning framework for optimizing adaptation.

A. Adaptable Parameters

Selection of the adaptable parameters θ is crucial in online dynamics model adaptation. Adapting all parameters ϕ of the LSTM and FNN in real-time is impractical for online adaptation, as the sheer number of parameters would result in prohibitively slow updates with respect to the timescale of the changing dynamics involved with high-speed off-road driving. On the other hand, adapting only the parameters of the parametric model ψ would be insufficient because this model cannot fully capture complex terrain interactions that the learned model $\zeta(\cdot)$ is designed to address.

To achieve adaptation fast enough for high-speed off-road driving, we adapt the linear combination of an ensemble of last-layer weights (Fig. 2). We represent the ensemble of last-layer weights in the FNN as a tensor $\mathbf{W} \in \mathbb{R}^{n_w \times n_{\texttt{out}} \times n_{\texttt{in}}}$, where n_w denotes the ensemble size, and $n_{\texttt{in}}$ and $n_{\texttt{out}}$ are the input and output feature dimensions of the last layer, respectively. The weight matrices of the ensemble are stacked along the first dimension to form this tensor. Now, we express how the adapted parameters appear in the learned model with:

$$\zeta(\eta; \phi, \theta) = (\phi^w + \theta^w)^\mathsf{T} \mathbf{W} \Phi(\eta; \phi^l) + \phi^b + \theta^b, \tag{3}$$

where $\Phi(\cdot) \in \mathbb{R}^{n_{\texttt{in}}}$ is the output of the second to last layer of the FNN, the learned parameters consist of $\phi = [\phi^l, \phi^w, \phi^b, \mathbf{W}]$, and the adaptable parameters consist of $\theta = [\theta^w, \theta^b]$. The learned ensemble weighting $\phi^w \in \mathbb{R}^{n_w}$ is adapted by θ^w and the last layer bias $\phi^b \in \mathbb{R}^{n_{\texttt{out}}}$ is adapted by θ^b.

Remark 1. *The tensor* $\mathbf{W}$ *is a set of basis functions, with their weighting dynamically adjusted through adaptation of* θ^w.

B. Online Adaptation

Here, we present the online adaptation scheme (Algorithm 1) implemented onboard the vehicle for dynamically updating the adaptable parameters θ. We employ a Kalman filter to quickly adapt the dynamics model to incoming measurements since the adaptable parameters are linear with respect to the dynamics. Propagation of the dynamics (1) occurs at a very quick timescale; we run the Kalman filter at a slower rate to improve parameter adaptation by adapting parameters every h time steps. This allows us to mitigate the impact of noisy and delayed state measurements, account for the minimal changes in integrated quantities over single time steps, and capture longer-term trends in parameter behavior. To form the Kalman filter, we assume the following:

Assumption 1. *The adaptable parameters evolve according to a random walk:* $\theta_{t+h} = \theta_t + w_t^\theta$ *with* $w_t^\theta \sim \mathcal{N}(0, Q)$.

Assumption 2. *Explicitly denoting the dependence on* θ*, we assume the system dynamics have additive Gaussian noise:* $x_{t+1} = f(x_t, u_t, y_t; \theta_t) + w_t^x$*, where* $w_t^x \sim \mathcal{N}(0, R)$.

Algorithm 1 Online Adaptation

1: **Input:** starting step s, initial parameter covariance P_s
2: **initialize** $t \leftarrow s, \theta_0 \leftarrow 0$
3: **while** running **do**
4: $\quad \hat{x}_{t+1:t+h} \leftarrow$ Propagate dynamics (1) h steps with θ_t
5: $\quad H_{t+h} \leftarrow$ Compute multi-step Jacobian with (4)
6: $\quad \bar{P}_{t+h} \leftarrow P_t + Q$
7: $\quad S_{t+h} = C\left(H_{t+h}\bar{P}_{t+h}H_{t+h}^\mathsf{T} + R\right)C^\mathsf{T}$
8: $\quad K_{t+h} \leftarrow \bar{P}_{t+h}H_{t+h}^\mathsf{T}C^\mathsf{T}S_{t+h}^{-1}$
9: $\quad \theta_{t+h} \leftarrow \theta_t + \gamma_t K_{t+h}C\left(x_{t+h} - \hat{x}_{t+h}\right)$
10: $\quad P_{t+h} \leftarrow \bar{P}_{t+h} + K_{t+h}CH_{t+h}\bar{P}_{t+h}$
11: $\quad t \leftarrow t + h$

With the notation $a_{t:t+h} \triangleq \{a_t, a_{t+1}, \ldots, a_{t+h}\}$, where a is any variable, we perform an h-step dynamics propagation to obtain predicted states $\hat{x}_{t+1:t+h}$. Since we propagate the dynamics over multiple steps, we compute the multi-step Jacobian $H_{t+h} = {}^{\partial \hat{x}_{t+h}}/_{\partial \theta_t}$ with the recursion:

$$\begin{aligned} H_{t+i} &= \mathcal{F}_{t+i}^x H_{t+i-1} + \mathcal{F}_{t+i}^\theta \\ H_t &= \mathcal{F}_t^\theta, \end{aligned} \tag{4}$$

where:

$$\begin{aligned} \mathcal{F}_{t+i}^x &= \frac{\partial}{\partial x_{t+i}} f(\hat{x}_{t+i}, u_{t+i}, y_{t+i}; \theta_t) \text{ and} \\ \mathcal{F}_{t+i}^\theta &= \frac{\partial}{\partial \theta_t} f(\hat{x}_{t+i}, u_{t+i}, y_{t+i}; \theta_t). \end{aligned} \tag{5}$$

By performing a multi-step propagation, we include the dynamics Jacobian $\mathcal{F}^x$ in the multi-step Jacobian computation (4). This Jacobian captures how the current state influences future states, and its recursive incorporation over multiple steps provides a richer, long-term understanding of how the adapted parameters θ influence the dynamics, improving the accuracy of adaptation. We make the following assumption to simplify computation of the multi-step Jacobian:

Assumption 3. *Changes in the learned model output with respect to changes in the state are negligible, i.e.,* $\partial\zeta/\partial x \approx 0$.

We use the measurement selection matrix $C \in \mathbb{R}^{3\times 10}$ in Algorithm 1 to select only the velocity measurements v for parameter updates. With $\hat{x}_{t+1:t+h}$ and H_{t+i} available, we perform a Kalman update with lines 6 to 10 of Algorithm 1. To prevent unnecessary parameter updates while the vehicle is moving very slowly, we scale the update in line 9 with:

$$\gamma_t = \frac{\|v_t\|_2^2}{\|v_t\|_2^2 + \varepsilon}, \tag{6}$$

where $\varepsilon > 0$ adjusts the scaling intensity. The updated parameters are then used with the controller (Section II-B) for the next h time steps.

C. Offline Meta-learning

With the adaptation framework established, our goal is to optimize the basis functions $\mathbf{W}$ and the Kalman filter parameters P_s, Q, R, and ε *offline* to maximize the effectiveness

Algorithm 2 Offline Meta-Learning

Input: Dataset $\mathcal{D}$
Output: Meta-learned parameters $\phi, \psi, P_s, Q, R, \varepsilon$
for N_E epochs **do**
  while $\mathcal{D} \neq \emptyset$ **do** ▷ iterate over entire dataset
    Sample N_B trajectories from $\mathcal{D}$ w/o replacement
    for each trajectory in batch **do**
      $\theta_t \leftarrow$ Run Algorithm 1 from $t-\tau$ to t
      Rollout dynamics (1) from t to $t+T$ with θ_t
      Compute trajectory loss $\mathcal{L}_i$ from t to $t+T$
    for each $\xi \in \{\phi, \psi, P_s, Q, R\}$ **do**
      $\xi \leftarrow \xi - \alpha \nabla_\xi \sum_{i=1}^{N_B} \mathcal{L}_i$ ▷ optimizer step

of online adaptation. This is achieved through a two-phase process: first, a data collection phase to gather diverse and representative trajectories, followed by a training phase to refine the parameterization and basis functions.

1) Data Collection: We begin by collecting autonomous driving data on a diverse set of terrains and conditions. Data are collected in discrete *runs*, with each run defined as a continuous period during which the robot is actively operating. Each run is subsequently partitioned into potentially overlapping trajectories, stored in dataset $\mathcal{D}$, where each trajectory is represented as $\{x_{t-\tau:t+T}, u_{t-\tau:t+T}, y_{t-\tau:t+T}\}$, encapsulating the state, control inputs, and external inputs over a period of $\tau + T$ time steps. Here, each trajectory will be associated to a unique reference time t, τ is the adaptation length, and T is the MPPI prediction length (Section II-B). Each trajectory is short relative to the overall duration of the runs—on the order of seconds. Consequently, individual trajectories are generally limited to a single terrain type and a specific terrain condition, providing focused and localized data.

2) Training: We perform offline training (Algorithm 2) with gradient-based meta-learning [8], which consists of an inner θ adaptation phase and an outer loop which optimizes ϕ, ψ P_s, Q, R, and ε (the tensor $\mathbf{W}$ is reshaped as a vector and contained within learned parameters ϕ). For each sampled trajectory, we can identify the adaptable parameters at the t-th time step, θ_t, by running Algorithm 1 for a τ-step adaptation period from $t-\tau$ to t. Once adaptation is complete, θ_t and the dynamics model (1) are used to generate a T-step dynamics prediction ($\hat{x}_{t+1:t+1+T}$). The i-th predicted trajectory is then evaluated against the corresponding ground truth trajectory using the multi-step loss function $\mathcal{L}_i$[1]:

$$\mathcal{L}_i = \frac{1}{T} \sum_{j=0}^{T} \left\| \hat{x}_{t+1+j} - x_{t+1+j} \right\|_2^2, \tag{7}$$

where t is the reference time associated with the i-th trajectory. Importantly, $\mathcal{L}_i$ is indirectly a function of ϕ, ψ, P_s, Q, R, and ε, which enables meta-learning of these parameters through backpropagation. During this process, gradients are propagated through the entire procedure of Algorithm 1, allowing the meta-learning framework to optimize both the adaptation dynamics and the underlying Kalman filter parameters effectively.

[1]For simplicity, we present the mean squared error (MSE) loss, however, in practice, we use the negative log-likelihood (NLL) loss from [11]

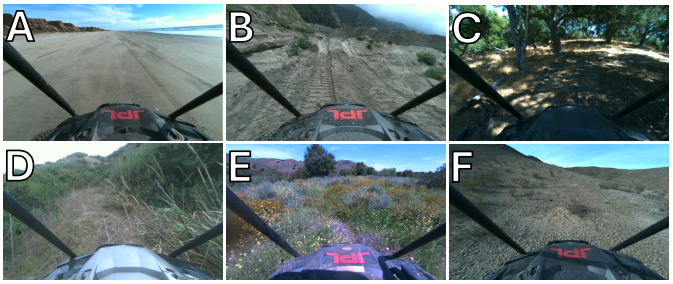

Fig. 3: Forward facing camera stills from the dataset highlighting a diverse range of terrains: A) flat sandy beach with a mixture of packed wet sand and loose dry sand; B) wet dense mud that forms deep ruts; C) dirt trails with low dry grass that weave through dense trees; D) mixed vegetation including dry, dense vehicle-height grass; E) dense overgrown mixed vegetation ranging in crushability; and F) loose gravel, uneven ground, and steep slopes.

TABLE I: Dataset statistics.

	Mean	Std	Min	5th %	Median	95th %	Max
Fwd. vel. (m/s)	4.55	2.91	-2.79	-0.01	4.56	9.62	15.25
Lat. vel. (m/s)	0.02	0.20	-1.31	-0.32	0.00	0.37	1.34
Yaw rate (rad/s)	0.00	0.19	-1.32	-0.32	0.00	0.31	1.40
Pitch (deg)	-0.4	4.9	-24.2	-8.0	-0.4	7.9	31.3
Roll (deg)	0.2	4.8	-28.7	-7.8	0.2	8.7	28.8

Remark 2. *Since* $\mathbf{W}$ *is optimized through the meta-learning process, meta-learning effectively determines the "directions" in the parameter space along which the Kalman filter can adapt. In other words, it specifies the subset of parameters that the Kalman filter targets for adaptation, guiding the adaptation process to focus on the most relevant and impactful aspects of the system's dynamics.*

Remark 3. *During the meta-learning process, we explicitly learn the covariance matrices* Q *and* R*, which govern how quickly the Kalman filter adapts* θ *to changes in the environment.*

IV. Experiments

We validate our approach through both real-world and simulated experiments to investigate the following hypotheses:

Hypothesis 1. *Meta-learned model adaptation reduces the dynamics model's prediction error during online operation.*

Hypothesis 2. *Effective model adaptation enhances closed-loop behavior, improving both stability and safety.*

A. Real-World Validation

1) Experimental Setup: We conducted real-world validation of our approach on a full-scale autonomous off-road vehicle: a modified Polaris RZR S4 1000 Turbo equipped with a 1.0L twin-cylinder engine, pictured in Fig. 1. The vehicle is outfitted with an extensive suite of onboard computation and sensors, including 2 LiDARs, 4 stereo/RGB cameras, IMUs,

TABLE II: Real-world validation results. The means and standard deviations over 4 runs of each configuration are displayed.

	Completion time, s	Average speed, $\mathrm{m\,s^{-1}}$	Prediction Error, m	# times crossed limit		Time exceeds limit, s		Cost ($\times 10^4$)	
				Track	Rollover	Track	Rollover	Track	Rollover
`Baseline (no adaptation)`	154.6 ± 16.9	5.06 ± 0.58	4.88 ± 0.47	8.0 ± 1.8	13 ± 5.4	5.32 ± 1.81	6.64 ± 2.09	40.8	20.2
`Meta-adaptation (ours)`	**130.9 ± 7.8**	**5.84 ± 0.33**	**3.10 ± 0.18**	**3.3 ± 2.1**	**3.8 ± 1.7**	**0.57 ± 0.53**	**0.85 ± 0.59**	**1.95**	**0.36**

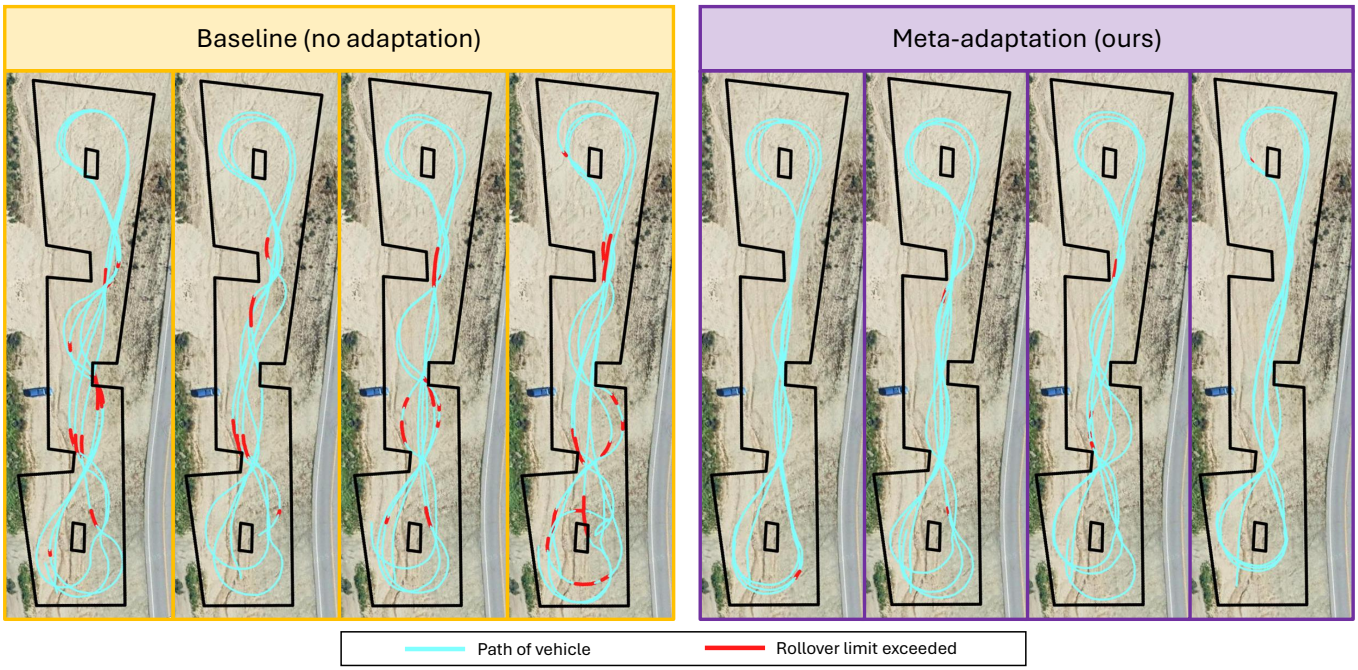

Fig. 4: Trajectories for all 3-lap real-world runs. The baseline configuration exhibits erratic motion, frequently violating course boundaries and rollover limits. In contrast, our adaptation configuration produces more deliberate and compliant trajectories, as the vehicle learns the terrain dynamics in real time.

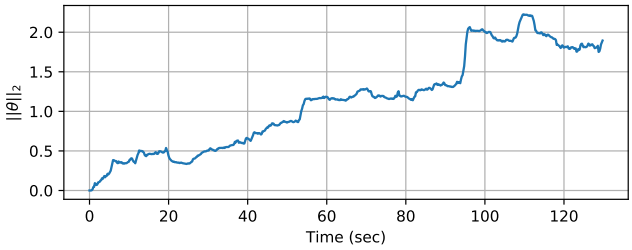

Fig. 5: Norm of the adapted parameters during one of the real-world runs.

GPS, and wheel encoders. State estimation is achieved using a localization module that integrates LiDAR and IMU data through a GTSAM-based Factor Graph Optimization (FGO) framework [7, 24]. A 3D voxel map is constructed by fusing geometric and semantic data to generate traversability and elevation maps, providing track cost and ground slope [2]. The vehicle is controlled via a power-assisted steering actuator, a brake pressure pump, and electronic throttle control.

For offline meta-learning, we collected a dataset comprised of approximately $1,700,000$ trajectories (9.5 hours) of autonomous driving that includes adverse weather conditions like rain. Approximately 60% is from the Mojave Desert near Helendale, CA; 30% from Halter Ranch near Paso Robles, CA; 10% from coastal sage near Oceanside, CA; and 5% from coastal dunes also nearby Oceanside, CA. At these locations, we collect data from a diverse range of terrains that are depicted and described in Fig. 3. We also provide statistics on key dataset quantities in Table I.

For each trajectory, we use an adaptation length of $\tau = 1,000$ steps ($20\,\mathrm{s}$) and a prediction length of $T = 250$ steps ($5\,\mathrm{s}$), with discrete time steps spaced ($0.02\,\mathrm{s}$) apart. Training follows Algorithm 2 for $N_E = 20$ epochs, starting with a 5-epoch pretraining phase without meta-learning (disabling adaptation by setting $\theta_t = 0$), followed by 15 epochs with meta-learning enabled. For comparison, a baseline model was also trained for 20 epochs on the exact same dataset without meta-learning.

We perform online adaptation (Algorithm 1) at a rate of $5\,\mathrm{Hz}$ by updating every $h = 10$ time steps. For control, MPPI (Section II-B) runs at $30\,\mathrm{Hz}$, leveraging the most recent set of adapted parameters θ for forward predictions. The car completes 3 laps around a figure-8 track with additional curves added to increase planning difficulty (see Fig. 1). We test the `Baseline (no adaptation)` and `Meta-adaptation (ours)` configurations for 4 repeated runs each.

To evaluate Hypothesis 1, we calculate the average model prediction error over each run, defined as the Euclidean distance between the endpoints of the predicted and actual trajectories. For performance assessment (Hypothesis 2), we

log the completion time and average speed of each run. Safety (Hypothesis 2) is analyzed by tracking the number of instances the car exceeds safety limits, the duration spent outside these limits, and the associated cost for both track boundary and rollover ratio limit violations.

2) Results: Figure 4 compares trajectories for the `Baseline (no adaptation)` and `Meta-adaptation (ours)` configurations and Fig. 1 shows video stills capturing the adaptation configuration in action. The `Baseline (no adaptation)` configuration results in erratic behavior, with frequent violations of track boundaries and rollover constraints. In contrast, the `Meta-adaptation (ours)` configuration yields more stable and compliant motion, as the dynamics model is adapted in real-time (Fig. 5).

The quantitative results in Table II further support the advantages of adaptation. The adapted model achieves significantly lower prediction error, confirming that online adaptation effectively learns the terrain dynamics, validating Hypothesis 1. Importantly, the `Meta-adaptation (ours)` configuration not only improves performance—achieving faster track completion times and higher average speeds—but also enhances safety metrics. Specifically, the adapted vehicle crosses safety limits (e.g., track boundaries and rollover thresholds) far less frequently and spends significantly less time in unsafe states compared to the baseline. These improvements translate to a dramatic reduction in associated track and rollover costs, highlighting the effectiveness of the adaptation method, confirming Hypothesis 2.

In high-speed off-road driving, an inaccurate dynamics model can lead MPPI to generate unsafe trajectories that fail to account for real-world dynamics. By incorporating accurate, environment-adapted predictions, our adaptation method enables MPPI to generate control sequences that balance performance and safety. These results underscore the importance of real-time model adaptation in ensuring both reliable and secure autonomous vehicle operation in challenging terrains.

B. Simulated Experiments

We perform extensive simulated experiments to further confirm our hypotheses.

1) Experimental Setup: The simulated experiments use the same baseline and meta-learned models trained for the real-world validation (Section IV-A1) and identical adaptation parameters. However, the physics simulator employs distinct dynamics based on the bicycle model, creating a real2sim transfer scenario where adaptation must account for the dynamics mismatch between the real-world training data and the simulated environment. We procedurally generate four diverse maps (Fig. 6), each with unique terrain and obstacle configurations. The maps are categorized as *shallow-sparse*, *shallow-dense*, *steep-sparse*, and *steep-dense*, where the first term denotes the steepness of terrain features, and the second describes the density of obstacles. This variety ensures that the simulated environment tests the adaptation across a wide range of scenarios.

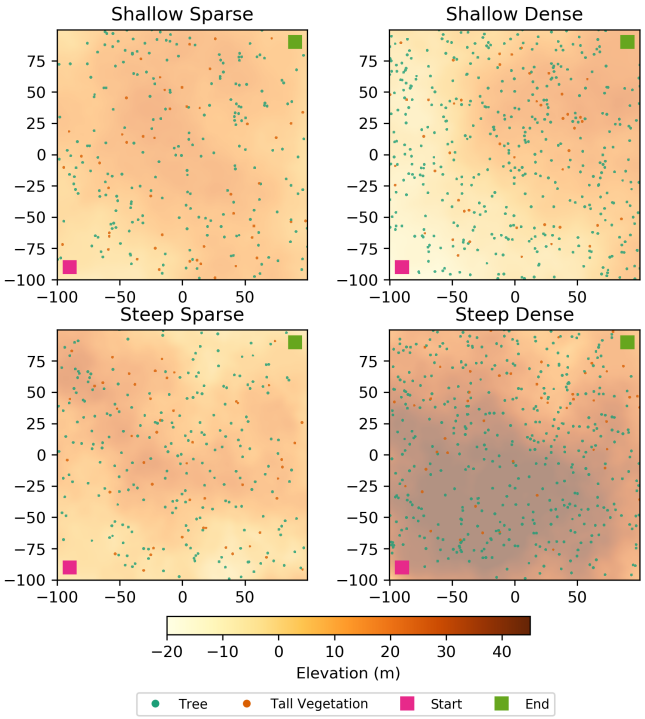

Fig. 6: Procedurally generated maps for the simulated experiments; horizontal and vertical axes are in meters.

Everything else remains consistent with the real-world validation, except for a few key modifications. First, since the maps are open with sufficiently distant boundaries, track costs are omitted from the analysis. Next, we evaluate two new configurations: `Adaptation` and `Sliding LSQ`. In the `Adaptation` configuration, online adaptation is applied to the *baseline* model to assess the impact of not meta-learning the basis functions and Kalman filter parameters. For the `Sliding LSQ` configuration, we implement the method of adaptation from [22], which consists of using sliding-window regularized least squares to adapt the weights of an ensemble model in real time. To ensure statistical significance, we perform 25 runs per configuration on each map.

2) Results: The results of the simulated experiments are summarized in Table III. As expected, prediction errors in simulation are generally higher than in real-world testing due to the real2sim gap. Nevertheless, all adaptation configurations outperform the baseline in prediction accuracy, with the meta-learned adaptation mostly achieving the lowest prediction errors, further confirming Hypothesis 1.

In terms of performance, the baseline configuration generally achieves shorter completion times and higher average speeds. However, this comes at the cost of compromised safety, with the baseline configuration exhibiting higher rollover occurrences, longer time spent exceeding limits, and increased safety costs. In contrast, the adaptation configurations demonstrate significantly improved safety, with the meta-learned configuration generally achieving the greatest reductions across all safety metrics. These findings align with the

TABLE III: Results of the simulated experiments. The means over 25 runs of each configuration are displayed.

	Completion time, s				Average speed, $\mathrm{m\,s^{-1}}$				Prediction Error, m			
Steepness	shallow	shallow	steep	steep	shallow	shallow	steep	steep	shallow	shallow	steep	steep
Obstacle Density	sparse	dense	sparse	dense	sparse	dense	sparse	dense	sparse	dense	sparse	dense
`Baseline (no adaptation)`	**39.2**	44.4	**73.0***	**86.2***	**5.88***	5.32	**4.06***	**3.59***	7.19	7.62	8.17	9.03
`Sliding LSQ`	46.7	46.6	87.7	102.0	5.04	5.06	3.43	3.06	5.71	5.00	5.21	5.35
`Adaptation`	40.4	**43.1***	83.0	95.3	5.70	**5.47***	3.59	3.15	4.54	4.45	4.25	**4.64**
`Meta-adaptation (ours)`	40.1	44.7	88.8	105.9	5.75	5.27	3.34	2.99	**2.96***	**2.19***	**3.67***	4.65
	# times over rollover limit				Time exceeding rollover limit, s				Rollover Cost ($\times 10^4$)			
`Baseline (no adaptation)`	2.7	**3.6**	5.4	9.9	1.30	5.31	5.09	7.50	1.09	1.75	1.82	8.91
`Sliding LSQ`	2.5	4.1	5.4	7.1	2.19	6.12	4.24	5.53	1.19	2.00	4.23	8.59
`Adaptation`	2.9	4.6	4.7	6.7	1.47	7.10	5.71	5.42	1.04	2.49	1.14	2.75
`Meta-adaptation (ours)`	**1.9**	5.3	**3.2***	**6.2**	**0.97**	**4.83**	**3.88**	**3.40***	**0.99**	**1.15***	**0.83**	**2.57**

*Significant best result: bootstrapped 95% confidence intervals of the mean do not overlap with those of any other configuration.

real-world validation results, further supporting Hypothesis 2.

Crucially, the comparison between our meta-learned configuration and the non-meta-learned adaptation configurations, `Adaptation` and `Sliding LSQ`, highlights the importance of meta-learning. Our configuration generally outperforms in prediction error and safety metrics across all map types. The non-meta-learned configurations rely on suboptimally selected basis functions, and the `Sliding LSQ` configuration requires hand tuning of the window length and regularization parameter—directly influencing the adaptation rate. In contrast, our method learns both the basis functions and adaptation dynamics from data, resulting in models that adapt more effectively, yield better closed-loop performance, and bridge the substantial dynamics mismatch from the real2sim gap.

V. Conclusion

In this work, we introduced a meta-learning framework for online dynamics model adaptation applied to high-speed off-road autonomous driving. By combining a Kalman filter-based adaptation scheme with meta-learned parameters, our approach addresses the challenges of unseen or evolving terrain dynamics, enhancing prediction accuracy, performance, and safety. Empirical validation through real-world and simulated experiments demonstrates that our method outperforms baseline and non-meta-learned adaptation strategies, particularly in safety-critical scenarios. Our method is applicable to a broad range of model-based control scenarios beyond off-road autonomous driving, as it can be integrated with any model that is linear in its adaptable parameters and any Model Predictive Control (MPC)-type controller. This contribution represents a step toward more robust and reliable autonomous systems capable of adapting to complex and changing environments.

Acknowledgments

The authors would like to thank Trey Smith and Brian Coltin for their helpful insights and feedback. The research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration (80NM0018D0004). This work was supported by a NASA Space Technology Graduate Research Opportunity under award 80NSSC23K1192, by the National Science Foundation under Grant No. 2409535, and by Defense Advanced Research Projects Agency (DARPA). Approved for Public Release, Distribution Unlimited. ©2025. All rights reserved.

References

[1] Anelia Angelova, Larry Matthies, Daniel Helmick, and Pietro Perona. Learning and prediction of slip from visual information. *Journal of Field Robotics*, 24(3):205–231, 2007.

[2] Deegan Atha, Xianmei Lei, Shehryar Khattak, Anna Sabel, Elle Miller, Aurelio Noca, Grace Lim, Jeffrey Edlund, Curtis Padgett, and Patrick Spieler. Few-shot semantic learning for robust multi-biome 3d semantic mapping in off-road environments. *arXiv preprint arXiv:2411.06632*, 2024.

[3] Gourav Bathla, Kishor Bhadane, Rahul Kumar Singh, Rajneesh Kumar, Rajanikanth Aluvalu, Rajalakshmi Krishnamurthi, Adarsh Kumar, RN Thakur, and Shakila Basheer. Autonomous vehicles and intelligent automation: Applications, challenges, and opportunities. *Mobile Information Systems*, 2022(1):7632892, 2022.

[4] Xiaoyi Cai, Michael Everett, Jonathan Fink, and Jonathan P How. Risk-aware off-road navigation via a learned speed distribution map. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2931–2937. IEEE, 2022.

[5] Franck Djeumou, Cyrus Neary, Eric Goubault, Sylvie Putot, and Ufuk Topcu. Neural networks with physics-informed architectures and constraints for dynamical systems modeling. In *Learning for Dynamics and Control Conference*, pages 263–277. PMLR, 2022.

[6] Franck Djeumou, Jonathan YM Goh, Ufuk Topcu, and Avinash Balachandran. Autonomous drifting with 3 minutes of data via learned tire models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 968–974. IEEE, 2023.

[7] Seyed Fakoorian, Kyohei Otsu, Shehryar Khattak, Matteo Palieri, and Ali-akbar Agha-mohammadi. Rose: Robust state estimation via online covariance adaption. In *The International Symposium of Robotics Research*, pages 452–467. Springer, 2022.

[8] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

[9] Jason Gibson, Bogdan Vlahov, David Fan, Patrick Spieler, Daniel Pastor, Ali-akbar Agha-mohammadi, and Evangelos A Theodorou. A multi-step dynamics modeling framework for autonomous driving in multiple environments. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7959–7965. IEEE, 2023.

[10] Jason Gibson, Anoushka Alavilli, Erica Tevere, Evangelos A Theodorou, and Patrick Spieler. Dynamics modeling using visual terrain features for high-speed autonomous off-road driving. *arXiv preprint arXiv:2412.00581*, 2024.

[11] Jason Gibson, Bogdan Vlahov, Patrick Spieler, and Evangelos A Theodorou. Multistep belief space dynamics learning for risk-aware control. *arXiv preprint*, 2025.

[12] Tyler Han, Alex Liu, Anqi Li, Alex Spitzer, Guanya Shi, and Byron Boots. Model predictive control for aggressive driving over uneven terrain. *arXiv preprint arXiv:2311.12284*, 2023.

[13] Karl D Iagnemma and Steven Dubowsky. Terrain estimation for high-speed rough-terrain autonomous vehicle navigation. In *Unmanned Ground Vehicle Technology IV*, volume 4715, pages 256–266. SPIE, 2002.

[14] Taekyung Kim, Hojin Lee, and Wonsuk Lee. Physics embedded neural network vehicle model and applications in risk-aware autonomous driving using latent features. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4182–4189. IEEE, 2022.

[15] Geert-Jan M Kruijff, Fiora Pirri, Mario Gianni, Panagiotis Papadakis, Matia Pizzoli, Arnab Sinha, Viatcheslav Tretyakov, Thorsten Linder, Emanuele Pianese, Salvatore Corrao, et al. Rescue robots at earthquake-hit mirandola, italy: A field report. In *2012 IEEE international symposium on safety, security, and rescue robotics (SSRR)*, pages 1–8. IEEE, 2012.

[16] Ivana Kruijff-Korbayová, Luigi Freda, Mario Gianni, Valsamis Ntouskos, Václav Hlaváč, Vladimír Kubelka, Erik Zimmermann, Hartmut Surmann, Kresimir Dulic, Wolfgang Rottner, et al. Deployment of ground and aerial robots in earthquake-struck amatrice in italy (brief report). In *2016 IEEE international symposium on safety, security, and rescue robotics (SSRR)*, pages 278–279. IEEE, 2016.

[17] Hojin Lee, Taekyung Kim, Jungwi Mun, and Wonsuk Lee. Learning terrain-aware kinodynamic model for autonomous off-road rally driving with model predictive path integral control. *IEEE Robotics and Automation Letters*, 2023.

[18] Jooyong Lee and Kara M Kockelman. Strategic evacuation for hurricanes and regional events with and without autonomous vehicles. *Transportation research record*, 2675(9):1398–1409, 2021.

[19] Jacob Levy, Tyler Westenbroek, and David Fridovich-Keil. Learning to walk from three minutes of real-world data with semi-structured dynamics models. In *8th Annual Conference on Robot Learning*, 2024.

[20] Elena Sorina Lupu, Fengze Xie, James A Preiss, Jedidiah Alindogan, Matthew Anderson, and Soon-Jo Chung. MAGIC-VFM-meta-learning adaptation for ground interaction control with visual foundation models. *IEEE Transactions on Robotics*, 2024.

[21] Xiangyun Meng, Nathan Hatch, Alexander Lambert, Anqi Li, Nolan Wagener, Matthew Schmittle, JoonHo Lee, Wentao Yuan, Zoey Chen, Samuel Deng, et al. Terrainnet: Visual modeling of complex terrain for high-speed, off-road navigation. *arXiv preprint arXiv:2303.15771*, 2023.

[22] Tomáš Nagy, Ahmad Amine, Truong X Nghiem, Ugo Rosolia, Zirui Zang, and Rahul Mangharam. Ensemble gaussian processes for adaptive autonomous driving on multi-friction surfaces. *IFAC-PapersOnLine*, 56(2):494–500, 2023.

[23] Issa AD Nesnas, Lorraine M Fesq, and Richard A Volpe. Autonomy for space robots: Past, present, and future. *Current Robotics Reports*, 2(3):251–263, 2021.

[24] Julian Nubert, Shehryar Khattak, and Marco Hutter. Graph-based multi-sensor fusion for consistent localization of autonomous construction robots. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 10048–10054. IEEE, 2022.

[25] Forrest Rogers-Marcovitz, Neal Seegmiller, and Alonzo Kelly. Continuous vehicle slip model identification on changing terrains. In *RSS 2012 Workshop on Long-term Operation of Autonomous Robotic Systems in Changing Environments*, 2012.

[26] Vandi Verma, Mark W Maimone, Daniel M Gaines, Raymond Francis, Tara A Estlin, Stephen R Kuhn, Gregg R Rabideau, Steve A Chien, Michael M McHenry, Evan J Graser, et al. Autonomous robotics is driving perseverance rover's progress on mars. *Science Robotics*, 8(80):eadi3099, 2023.

[27] Bogdan Vlahov, Jason Gibson, David D Fan, Patrick Spieler, Ali-akbar Agha-mohammadi, and Evangelos A Theodorou. Low frequency sampling in model predictive path integral control. *IEEE Robotics and Automation Letters*, 2024.

[28] Grady Williams, Paul Drews, Brian Goldfain, James M Rehg, and Evangelos A Theodorou. Aggressive driving with model predictive path integral control. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1433–1440. IEEE, 2016.