Online Adaptation of Terrain-Aware Dynamics for Planning in Unstructured Environments

William Ward, Sarah Etter, Tyler Ingebrand, Christian Ellis, Adam J. Thorpe, Ufuk Topcu

***Abstract*—Autonomous mobile robots operating in remote, unstructured environments must adapt to new, unpredictable terrains that can change rapidly during operation. In such scenarios, a critical challenge becomes estimating the robot's dynamics on changing terrain in order to enable reliable, accurate navigation and planning. We present a novel online adaptation approach for terrain-aware dynamics modeling and planning using function encoders. Our approach efficiently adapts to new terrains at runtime using limited online data without retraining or fine-tuning. By learning a set of neural network basis functions that span the robot dynamics on diverse terrains, we enable rapid online adaptation to new, unseen terrains and environments as a simple least-squares calculation. We demonstrate our approach for terrain adaptation in a Unity-based robotics simulator and show that the downstream controller has better empirical performance due to higher accuracy of the learned model. This leads to fewer collisions with obstacles while navigating in cluttered environments as compared to a neural ODE baseline.**

I. Introduction

Rapid adaptation to unknown environments and terrain is critical for autonomous mobile robots. In off-road navigation, unpredictable terrain features such as rocky paths, forest floors, and wet fields can cause skidding, tripping, or immobilization, jeopardizing the robot's ability to reach its objective. Autonomous ground vehicles must therefore dynamically adjust their behavior to terrain-specific conditions. This adaptation is challenging because terrain variations directly alter system dynamics. For example, tire response to acceleration depends on surface friction. As a result, accurate terrain-aware dynamics models are critical for control, particularly in planning-based control frameworks such as model predictive control (MPC) or model predictive path integral control (MPPI), which depend on accurate forward predictions of the robot's dynamics [18, 47]. Estimating the robot's dynamics online in uncertain environments is a particularly challenging problem because it typically must be performed without prior data or knowledge of the environment. This challenge is especially apparent for mobile robots operating in locations where human intervention is impractical or impossible, such as extraterrestrial exploration or hazardous disaster response. Consequently, learned models for dynamics prediction or estimation must be able to rapidly adapt to new environments and terrain at runtime.

W. Ward, T. Ingebrand, C. Ellis, A. Thorpe, U. Topcu are with the Oden Institute for Computational Engineering & Science, University of Texas at Austin. S. Etter is with the Department of Computer Science, University of Texas at Austin. C. Ellis is also with the DEVCOM Army Research Laboratory. Email: wward@utexas.edu, etter@utexas.edu, tyleringebrand@utexas.edu, christian.ellis@austin.utexas.edu, adam.thorpe@autin.utexas.edu, utopcu@utexas.edu.
Corresponding author: A. Thorpe

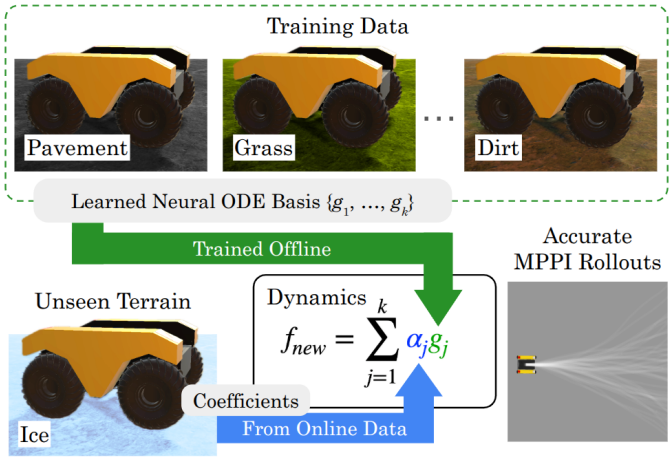

Fig. 1: We rapidly identify the robot dynamics on a new, unseen terrain online using a set of learned neural ODE basis functions and a small amount of online data, leading to accurate, terrain-aware MPPI dynamics rollouts.

We present a method for modeling system dynamics that enables adaptation to new environments and conditions at runtime using limited online data. We focus on terrain adaptation for autonomous ground vehicles, though we note that our approach is broadly applicable to various systems and environment interactions. Our approach uses the theory of function encoders, which have been applied in reinforcement learning [20], operator learning for PDE modeling [21, 44], and zero-shot dynamics estimation using neural ordinary differential equation (ODE) basis functions [19]. Rather than training a separate dynamics model for each terrain, we use function encoders to learn a set of neural network basis functions that spans a space of robot dynamics. At runtime, the dynamics are represented as a linear combination of the learned basis functions, thereby reducing the problem of dynamics estimation to that of identifying a vector of coefficients. We use this adapted model within a model-predictive controller to plan trajectories on unseen terrain, enabling closed-loop control without retraining.

State-of-the-art autonomous off-road navigation systems use a complex technology stack involving image segmentation [45], object identification, terrain estimation [23, 31], robot planning [50], and control [6, 15, 32]. However, many approaches utilize simplified kinematic models, and neglect

terrain-induced dynamics effects that are important for accurate MPC-based planning. For instance, a ground robot equipped with a model tuned for driving on dry asphalt will experience significant tracking errors or loss of control when encountering slick, icy surfaces due to unmodeled changes in tire friction.

A common strategy for terrain-aware modeling is to classify the terrain using semantic segmentation or discrete labels [8, 39, 46], and then either switch to a corresponding dynamics model or adapt known models via parameter estimation. However, these approaches are only effective if the unknown terrain corresponds to a known model. In practice, this assumption fails in remote, unstructured environments, where terrain may be ambiguous or variable. For example, robots may encounter mixed terrain, such as patches of grass interspersed with mud, or micro-terrain features, such as loose gravel on top of hard soil, that does not fit neatly into predefined categories.

Our main contributions are: (1) a method for terrain adaptation that combines function encoders, neural ODEs, and MPPI control, and (2) an implementation of this approach in a realistic robotic simulator that demonstrates accurate closed-loop control on varied terrains. While prior research has introduced function encoders for zero-shot prediction [19], we extend this approach to closed-loop control via MPPI. Specifically, we integrate the adapted dynamics model into an MPC pipeline to support real-time trajectory optimization on previously unseen terrain. The function encoder learns a set of neural ODE basis functions that span a space of vehicle dynamics, trained on trajectories from known terrains. In contrast to existing approaches, our approach collects limited data from a new terrain at runtime and rapidly adapts its coefficients online without retraining or fine-tuning. We implement and evaluate our approach in a high-fidelity Unity-based robotics simulator using a Clearpath Warthog robot. Figure 1 shows a visual representation of the proposed method.

A. Related Work

Learning-based estimation and prediction: Accurately predicting the dynamics of robotic systems operating in diverse, unstructured environments remains a critical yet challenging problem. Traditional approaches often rely on analytical models derived from first principles. However, the complexity and the difficulty of parameterizing real-world interactions, especially with varying terrain, significantly limit their applicability and accuracy in off-road settings. Data-driven methods, leveraging neural networks to learn dynamics from experience, have shown promise in specific scenarios, including robotic manipulation [3, 16], autonomous driving on structured roads [33, 40], and even highly dynamic maneuvers on controlled surfaces [9]. These learned models, however, typically struggle to generalize to novel terrains not encountered during training, which can lead to significant errors in trajectory prediction and control. The assumption of consistent ground conditions, inherent in many existing models, breaks down in off-road or unstructured environments where terrain properties change rapidly and unpredictably.

Transfer learning & domain adaption: Pre-trained dynamics models, while effective in their training domains, often exhibit poor performance when deployed in novel environments. Domain adaptation techniques aim to mitigate this by fine-tuning models using data collected in the new environment. In the context of off-road robotics, significant research has focused on capturing terrain-specific properties like friction, slip, deformability, and sinkage through parameter estimation [7, 13, 26, 36, 37, 38]. These methods typically involve robots estimating unknown terrain parameters from onboard sensor data at runtime. However, a critical limitation of this approach is the reliance on a predefined underlying dynamics model. If the true terrain dynamics deviate significantly from this assumed structure, parameter estimation will fail to provide accurate predictions.

Meta-learning approaches [14, 49] expand these capabilities by training models specifically to adapt quickly to new tasks or environments with minimal data. These techniques relate closely to lifelong learning [27, 28], where systems continuously incorporate new knowledge while maintaining performance on previously learned tasks. Despite their advantages, meta-learning approaches ultimately still rely on some form of fine-tuning, which presents challenges in time-critical or resource-constrained scenarios. What is needed are models capable of online adaptation without retraining or fine-tuning. Our work addresses this by developing a dynamics estimation approach that adapts to novel terrains in real-time without explicit retraining. Related online policy adaptation methods use a two-stage training procedure to train a base policy and an adaptation module that predicts a latent vector that captures the effects of different terrains [24], such that the robot can adapt to changing terrains online without meta-learning.

Foundation models have demonstrated exceptional performance and adaptability across unstructured domains. In robotics, researchers have created diverse datasets [35] and developed generalizable models for various morphologies [1, 2, 12, 41, 42], primarily using transformer architectures and leveraging behavior cloning. Yet they lack built-in mechanisms for adaptation. While our work focuses specifically on ground robot dynamics estimation across diverse terrains, we take a similar approach in principle and pre-train a function encoder model using diverse data before adapting predictions at runtime from observations.

Off-road driving at the limits: Off-road autonomy presents unique challenges due to highly variable terrain conditions. Significant research has focused on estimating terrain traversability and hazard detection to identify safe paths [17, 29, 31], yet comparatively less attention has been given to directly estimating dynamics and terrain interactions. Terrain classification and semantic segmentation represent common adaptation approaches, where robots use sensor measurements to identify current terrain types [34] and distinguish between traversable and non-traversable regions [4, 23, 25, 30, 31]. Typically, neural network classification and segmentation models are trained offline using data from various environments, then used at runtime to generate cost maps that improve

local planning decisions. However, this approach has important limitations. First, terrain classification is only useful when the robot already knows the associated dynamics. Otherwise, the robot must approximate using the closest available model, potentially leading to dangerous situations like attempting to use a pavement model on ice. Second, classification and segmentation primarily inform cost functions rather than dynamics functions, helping robots avoid dangerous areas but not necessarily navigate them when unavoidable.

Planning and control for autonomous navigation: Adaptation strategies for control systems typically operate through either cost function or dynamics model adjustments. Local planners such as MPPI control use cost functions to evaluate planned trajectories and improve future control decisions. Prior work demonstrates that terrain-informed cost functions help robots avoid dangerous terrain [4, 23, 25, 31]. However, when robots must traverse challenging terrain, such as large patches of ice, they require terrain-informed dynamics that enable local planners to determine how to navigate such terrain, beyond simply identifying areas to avoid. Dynamics-based adaptation directly models how the robot interacts with different terrains. This approach allows control systems to anticipate and compensate for terrain-specific behaviors such as wheel slip or reduced traction [10, 11]. While cost function adaptation primarily influences path selection, dynamics adaptation affects both planning and execution by providing more accurate predictions of how control inputs will translate to robot motion across varying terrain types. Our work focuses on dynamics adaptation as it provides a more complete solution for autonomous navigation in challenging environments.

II. Preliminaries & Problem Formulation

We seek to compute an estimate of robot dynamics online without prior data or knowledge of the environment. This is particularly challenging for mobile robots navigating unstructured environments. Specifically, we seek a model that is compatible with the Model Predictive Path Integral (MPPI) control framework [48]. MPPI is a sampling-based optimization algorithm that computes an approximate solution to a optimal control problem at each control step. Below, we summarize its formulation and refer the reader to [48] for additional details.

Let $w \in \mathcal{W}$ denote the world state, which parameterizes the terrain effects on the dynamics. The discrete-time system dynamics are given by,

$$x_{t+1} = F^w(x_t, v_t), \tag{1}$$

where $x_t \in \mathbb{R}^n$ is the state of the system at time t and $v_t \in \mathbb{R}^m$ is the control input. Note that the world state w is unobserved, meaning the true dynamics F^w are unknown. We assume that the actual control inputs applied to the system $V = \{v_0, v_1, \ldots, v_{T-1}\}$, over the horizon $T \in \mathbb{N}$, are not directly controllable. Instead, they are affected by noise such that $v_t \sim \mathcal{N}(u_t, \Sigma)$. The goal of MPPI is to compute a mean control sequence $U = \{u_0, u_1, \ldots, u_{T-1}\}$.

Given a terminal cost function $\phi(x_T)$ and a running cost function $\mathcal{L}(x_t, u_t)$, the discrete-time optimal control problem is defined as,

$$\min_{U \in \mathcal{U}} \mathbb{E}\left[\phi(x_T) + \sum_{t=0}^{T-1} \mathcal{L}(x_t, u_t)\right], \tag{2}$$

where $\mathcal{U}$ is the set of admissible control sequences and the expectation is taken with respect to the distribution over the control inputs V. We presume that the running cost $\mathcal{L}(x_t, u_t)$ can be split into a state-dependent cost term $c(x_t)$ and a control cost that is a quadratic function of u_t,

$$\mathcal{L}(x_t, u_t) = c(x_t) + \frac{\lambda}{2}(u_t^\top \Sigma^{-1} u_t + \beta_t^\top u_t). \tag{3}$$

From [48], the affine term β allows the location of the minimum control cost to be shifted away from zero.

Using a learned estimate $\hat{F}^w$ of the dynamics F^w in (1), the control algorithm computes r rollouts, or state trajectories. From a given initial condition x_0 and control sequence V, a rollout is the predicted state trajectory $\{x_1, \ldots, x_T\}$ such that $x_{t+1} = \hat{F}^w(x_t, v_t)$. The state cost of a rollout is given by,

$$S(V, x_0) = \phi(x_T) + \sum_{t=0}^{T-1} c(x_t). \tag{4}$$

Each rollout is generated by randomly sampling sequences of control perturbations $V_1, \ldots, V_r$, which are random draws from the distribution over V. Then, we compute the state costs $S(V_i, x_0)$, $i = 1, \ldots, r$, for each rollout.

After we obtain the costs, we compute probability weights and update the control sequence $\{u_0, \ldots, u_{T-1}\}$ by taking a probability-weighted average followed by a Savitsky-Galoy smoothing filter. We then send the first control input u_0 to the robot, and the remaining sequence is used to initialize the optimization algorithm at the subsequent time instant. See [48] for more details.

Importantly, the system model in (1) is effectively a black box. In MPPI, the control updates and optimization procedure are decoupled from the rollout generation process. This decoupling allows learning-based models to serve as drop-in replacements for analytical models. However, the performance of MPPI is highly sensitive to model fidelity. Model predictive control—and MPPI in particular—relies heavily on accurate forward models to generate rollouts. When there is mismatch between the learned model and the true system, the optimizer evaluates candidate control sequences under incorrect assumptions about how the system will evolve. As a result, it may select actions that appear optimal under the model but lead to poor or unsafe behavior in the real system. These errors are especially problematic in off-road environments, where unmodeled nonlinearities, contact dynamics, and terrain effects introduce significant model mismatch. Moreover, because learned models are typically trained in fixed conditions, they often fail to generalize to new environments without mechanisms for online adaptation.

We consider the problem of learning a model $\hat{F}^w$ of the system dynamics in (1). The key challenge in the off-road

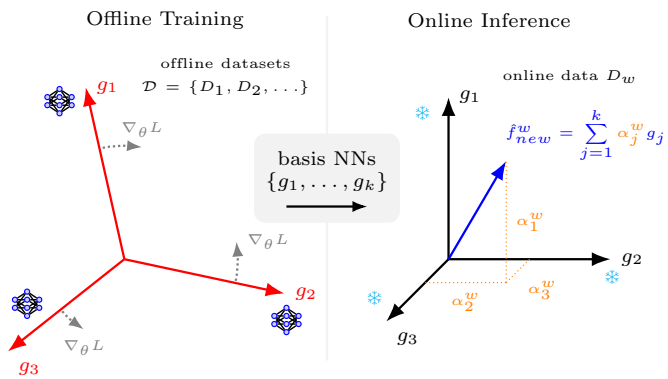

Fig. 2: Function encoders consist of two steps: a training step to learn the neural network basis functions using offline datasets and an inference step using online data.

setting is that the world state w is unobserved. To adapt to changing conditions and terrains, we seek a learned model that is both terrain-aware and computationally efficient for model-predictive control, suitable for deployment on hardware. During training, we presume access to historical trajectories of the robot operating on diverse terrains. From the trajectory data, we construct a set of datasets $\{D^{w_1}, \ldots, D^{w_\ell}\}$. Each dataset D^{w_i} consists of states and actions collected from a single dynamics function F^{w_i},

$$D^{w_i} = \{(x_j, v_j, x_{j+1})\}_{j=0}^{m}, \tag{5}$$

where $x_{j+1} = F^{w_i}(x_j, v_j)$. Such a dataset may be available, for instance, from historical trajectories observed during robot operation.

III. Method

We seek to learn a model of the system dynamics that can rapidly adapt to new terrains at runtime. The evolution of the system can be expressed in terms of the equation,

$$x_{t+1} = F^w(x_t, v_t) = x_t + \int_t^{t+1} f^w(x(\tau), v_t) d\tau, \tag{6}$$

where f^w is the underlying vector field, and we integrate f^w holding v_t constant over the interval from t to $t+1$.

Neural ordinary differential equations [5] are a common approach for learning dynamical system models. These models approximate the vector field f of a dynamical system with a neural network f_θ. Unlike discrete-time neural network models which map directly to the future states in a single forward pass, neural ODEs integrate over time in order to compute the solutions to initial value problems. Because neural ODEs compute the vector field of the system, they demonstrate excellent prediction accuracy without suffering from compounding errors that are inherent to discrete-time models. Like standard Euler integration, discrete-time models accumulate prediction errors over long time horizons, which compound over time and can be detrimental for model predictive control that requires accurate rollouts. However, neural ODEs do not have a built-in mechanism for online adaptation.

To learn a model that can rapidly adapt to new terrains at runtime without retraining, we use function encoders with neural ODE basis functions [19]. Our work advances the existing function encoder approach by adapting the formulation with neural ODE basis functions to use least squares coefficient calculations and integrating it within MPPI control. In section IV, we demonstrate its effectiveness for ground robots navigating unstructured environments.

A. Learning Neural ODE Basis Functions

Let $\mathcal{F} = \{f^w \mid w \in \mathcal{W}\}$ denote the space of all dynamics functions parameterized by the world state $w \in \mathcal{W}$. The space $\mathcal{F}$ can be viewed as the space of all possible dynamics of the robot on different terrains, where the world state w determines the terrain. We assume that $\mathcal{F}$ is a Hilbert space. Note that this assumption is generally mild, since it imposes only mild regularity or smoothness conditions on the dynamics. This assumption is key to our approach, since it allows us to learn a representation for $\mathcal{F}$.

We employ function encoders as in [19] to estimate the dynamics. Figure 2 shows a visual representation of function encoders. Function encoders learn a set of neural network basis functions $\{g_1, \ldots, g_k\}$ to span the Hilbert space $\mathcal{F}$. We can compute an estimate $\hat{f}^w$ of any function $f^w \in \mathcal{F}$ as a linear combination of the learned basis functions,

$$\hat{f}^w = \sum_{j=1}^{k} \alpha^w_j g_j \tag{7}$$

where $\alpha^w \in \mathbb{R}^k$ are real coefficients corresponding to f^w. Intuitively, we can view the model as a linear combination of neural ODE basis functions,

$$x_{t+1} - x_t = \sum_{j=1}^{k} \alpha^w_j \int_t^{t+1} g_j(x(\tau), v_t) d\tau \tag{8}$$

$$= \sum_{j=1}^{k} \alpha^w_j G_j(x_t, v_t) \tag{9}$$

where $G_j(x_t, v_t) := \int_t^{t+1} g_j(x(\tau), v_t) d\tau$ is a neural ODE. Note that the basis functions output the *change* in state. This means that in practice, we generally need to account for this during integration by adding the state x_t back at each integration step. The future state of the system x_{t+1} can be computed as in (6) by integrating $\hat{f}^w$ as in (9).

Importantly, we can quickly compute an estimate $\hat{f}^w$ of any dynamics function $f^w \in \mathcal{F}$ by identifying the coefficients $\alpha^w \in \mathbb{R}^k$. The coefficients α^w can be computed efficiently in closed form as the solution to a least squares problem via the normal equation as,

$$\alpha^w = \begin{bmatrix} \langle G_1, G_1 \rangle & \cdots & \langle G_1, G_k \rangle \\ \vdots & \ddots & \vdots \\ \langle G_k, G_1 \rangle & \cdots & \langle G_k, G_k \rangle \end{bmatrix}^{-1} \begin{bmatrix} \langle F, G_1 \rangle \\ \vdots \\ \langle F, G_k \rangle \end{bmatrix}. \tag{10}$$

This requires us to compute the inverse of a $k \times k$ Gram matrix, which is generally $\mathcal{O}(k^3)$. However, k is generally chosen to

be small, on the order of 100 or fewer, meaning the inverse can be computed efficiently. Additionally, we note that this computation can be performed once offline after training, and that we do not need to recompute the inverse during inference. Since the basis functions output the change in state $x_{t+1} - x_t$, the inner product between functions F and G is defined as

$$\langle F, G \rangle = \int_{\mathcal{X} \times \mathcal{U}} F(x,u)^\top G(x,u) d(x,u). \tag{11}$$

We approximate the inner product using Monte Carlo integration [19, 20] using a small amount of input-output data of the form (x_t, v_t) and $x_{t+1} - x_t$ collected online.

Offline Training: For each function f^{w_i} and the corresponding dataset D^{w_i}, we first compute the coefficients via (10) and then compute an empirical estimate $\hat{f}^{w_i}$ of f^{w_i} as in (7). We then compute the MSE of the estimate using the norm induced by the inner product of $\mathcal{F}$. The loss is the sum of the errors for all $\hat{f}^{w_i}$, which is minimized via gradient descent. After training, the basis functions $\{g_1, \ldots, g_k\}$ are fixed. We denote the low-dimensional representation of $\mathcal{F}$ as

$$\hat{\mathcal{F}} := \overline{\text{span}}\{g_1, \ldots, g_k\}. \tag{12}$$

Online Inference: At inference time, the basis functions do not require retraining or fine-tuning. For any new function $f^w_{new} \in \mathcal{F}$, we simply need to identify the coefficients of the dynamics representation $\hat{f}^w_{new}$ in the learned function space $\hat{\mathcal{F}}$ in order to estimate the function via (7). This representation is central to our approach, since it offers a means to quickly identify and estimate the dynamics. In practice, we typically require only a small amount of online data to compute α^w. Generally, we can identify α^w using approximately 100 data points, which corresponds to a few seconds of online data.

As a remark, it is important to clarify the coefficients α^w correspond to a *single* terrain. However, the terrain may change during off-road navigation, e.g. from dirt to sand to mud. We note that our proposed approach can likely be adapted to streaming data collected online via recursive least squares, which would offer a significant advantage over existing adaptation approaches such as meta-learning that require network fine-tuning, since we can use update our learned model via a simple coefficient update.

B. Online Terrain Adaptation with MPPI

At runtime, we collect a small sample of online data and solve the least squares problem (10) to estimate an approximate dynamics model $\hat{f}^w$ for f^w as a linear combination of the learned neural ODE basis functions (9). We then use $\hat{f}^w$ in place of the dynamics model in (1) to compute state trajectory rollouts from the robot's current state and compute the control inputs from MPPI as normal. Using the r sampled control sequences $V_1, \ldots, V_r$, we generate predicted state trajectories using (9) by integrating the model over the planning horizon T. Then, for each rollout, we compute the state costs $S(V_i, x_0)$ as in (4) and compute the MPPI probability weights as normal.

By incorporating terrain-dependent dynamics, we obtain more accurate trajectory predictions and corresponding state costs (4). As a result, MPPI generates more accurate trajectories that guide it toward better candidate solutions of the optimal control problem (2). Moreover, the rollouts for MPPI can be computed efficiently since they can be computed in batch, where each single-step prediction incurs only a cheap forward pass through $\hat{f}^w$. Most importantly, the function encoder model gives us terrain-aware predictions. Using function encoders with neural ODE basis functions to compute the rollouts for MPPI yields a computationally efficient online adaptation scheme that does not rely on expensive gradient updates or neural network fine-tuning.

IV. Results

We demonstrate our proposed approach for terrain-aware planning and control in Phoenix, a Unity-based, high-fidelity robotic simulation environment designed for realistic outdoor navigation experiments. Phoenix provides accurate dynamics modeling for robotic vehicles and supports obstacle-aware planning. In this work, we evaluate our approach using a simulated Clearpath Warthog robot, a wheeled ground rover employing skid-steer drive.

The robot's tires in Phoenix feature adjustable friction parameters, enabling simulation of various terrains such as pavement, mud, and ice. Phoenix defines each tire's forward and lateral friction behavior using five parameters: extremum slip, extremum value, asymptote slip, asymptote value, and stiffness [43]. Here, slip quantifies the relative rotation of the tire with respect to the ground before friction thresholds are reached, while value denotes the frictional force at these thresholds. The extremum represents the point of maximum frictional force, and the asymptote describes the steady-state friction force after surpassing the extremum slip. Adjusting these parameters modulates the robot's traction, effectively simulating diverse terrain interactions and driving dynamics.

We leverage the simulator's adjustable friction parameters to replicate a variety of terrains. Specifically, we define two extreme parameter sets: a "normal friction" set representing pavement conditions, a, and a "low friction" set representing slick, icy conditions b. To systematically generate intermediate terrains, we select 6 parameters in-between these two values a and b using a convex combination $\theta a + (1-\theta) b$ of the parameter vector with $\theta \in [0, 0.25, 0.5, 0.75, 0.812, 0.875, 0.939, 1]$ to create a total of 8 different terrains. Each resulting terrain corresponds uniquely to a different world state w.

A. Dataset Collection and Pre-processing

For each simulated terrain, we collect approximately 15 minutes of driving data using the simulator's integrated MPPI controller and analytical model guided by manually specified waypoints. The dataset comprises odometry measurements and control commands sent to the robot. Specifically, odometry data includes the robot's inertial frame coordinates x, y, heading angle ψ, linear velocities v_x, v_y, and angular velocity ω_z, with velocities expressed in the robot's local (body) coordinate frame. Additionally, we record the commanded forward and angular velocities throughout each run.

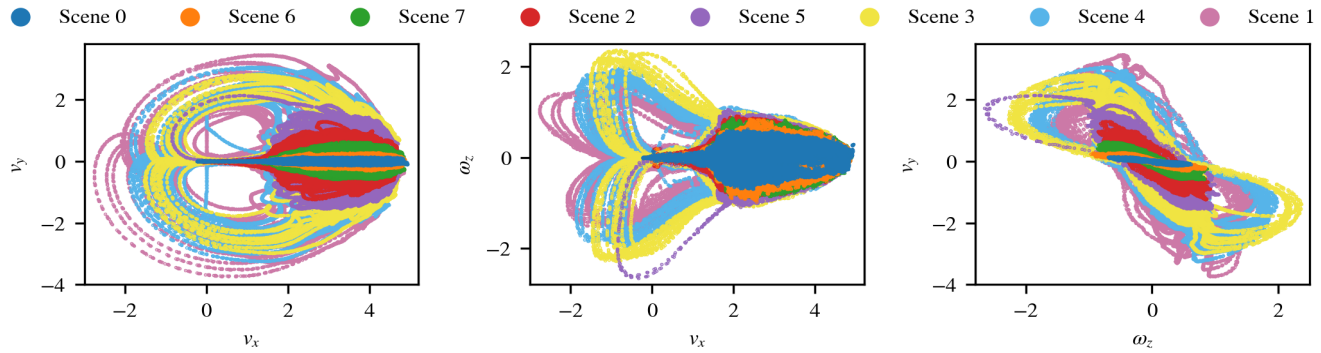

Fig. 3: We collect data corresponding to diverse terrains that induce significant variation in the dynamics. Several terrains have low friction, which causes high lateral velocity.

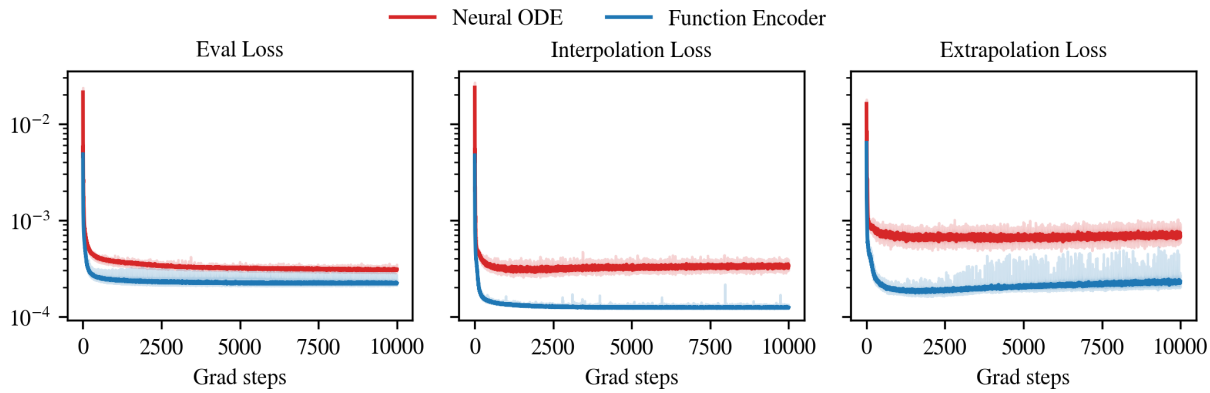

Fig. 4: (Left) MSE of the function encoder versus the neural ODE model on unseen data from in-distribution scenes. (Middle) MSE on an out-of-distribution scene whose dynamics is an interpolation of two in-distribution scenes. (Right) MSE on an out-of-distribution scene whose dynamics are more slippery than anything in the training sets, i.e., extrapolation. We evaluate over 10 random seeds. Each plot shows the median bounded by the minimum and maximum values (shaded regions).

The collected dataset comprises measurements expressed in both inertial and body reference frames. Because the system dynamics are invariant under translation and rotation, we apply a coordinate transformation to express the inertial states (x, y, ψ) within the robot's local (body) frame. Since the robot's dynamics remain consistent irrespective of absolute position or orientation, converting global states into the body frame significantly improves data efficiency and model learning. In the inertial frame, odometry data collected from the simulator tends to be sparse and broadly distributed across a large range, leading to limited overlap among trajectories. This sparsity can degrade model accuracy, motivating our transformation to the robot's local frame. However, we acknowledge that such a coordinate transformation may not be feasible or practical in all real-world scenarios.

Figure 3 shows the distribution of velocities recorded on each terrain. On high-friction terrain, the robot exhibits minimal slip, and its y-velocity remains tightly bounded. As terrain friction decreases, slip increases, particularly during acceleration. In low-friction conditions, the robot frequently loses traction, resulting in significant lateral drift, spinouts, and occasional backward sliding.

From each terrain, we generate a dataset D^{w_i} from the collected trajectory $\{x_0, v_0, \ldots, x_{T-1}, v_{T-1}, x_T\}$. From the trajectory, we compute the time difference Δt between consecutive states, as well as the state difference $\Delta x_t = x_{t+1} - x_t$. We then construct a dataset of input-output pairs $(x_t, v_t, \Delta t, \Delta x_t)$ for training the model as described in Equation (9).

B. Model & Implementation Details

Both the neural ODE and function encoder (FE-NODE) models are trained on identical datasets. Training was performed on a desktop equipped with an NVIDIA RTX 4060 GPU, Intel Core i7-14700 CPU, and 32GB RAM. The function encoder model required approximately 1 hour and 15 minutes of training, while the neural ODE model took approximately 21 minutes. For inference, both models used a Runge-Kutta 4 (RK4) numerical integrator to compute forward passes. In our implementation, the hyperparameter k denoting the number of basis functions was set to $k = 8$. The number of basis functions required depends on the system. However, this is

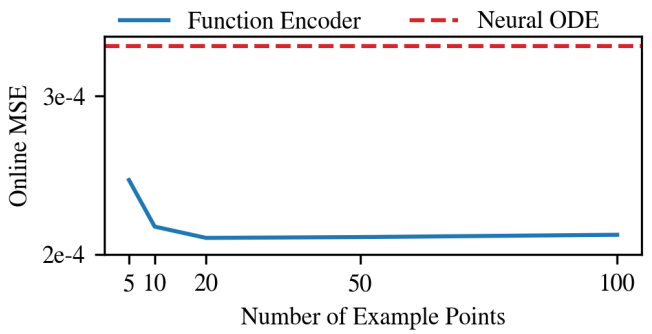

Fig. 5: We evaluate the performance of the function encoder in an online setting. At each timestep, we use the most recent n data points to compute the coefficients. This figure shows that the function encoder only requires a small amount of online data to get accurate 1-step predictions.

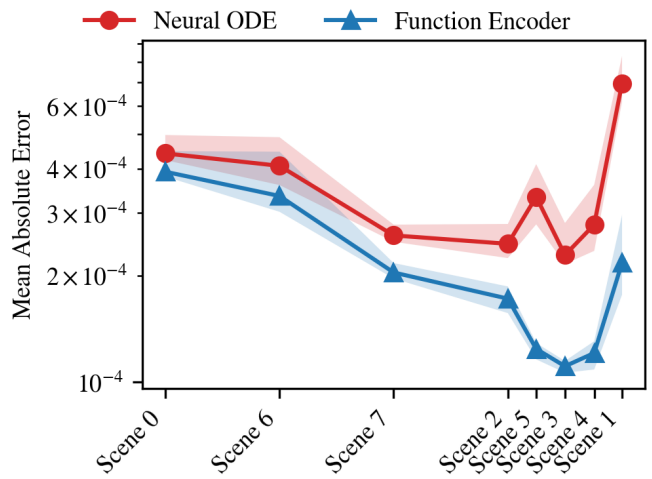

Fig. 6: Error per-scene of our proposed approach compared to a standard neural ODE. Our approach achieves low error across all scenes, and does not suffer an increase in error for the interpolation dataset (scene 5). In contrast, neural ODEs perform much worse at this unseen, interpolation dataset. We evaluate over 10 random seeds. The median is bounded by the minimum and maximum values (shaded regions).

generally unknown without prior knowledge. Therefore, we can overestimate with a larger number of basis functions than needed. For an ablation analysis of increasing the number of basis functions, see [19, 22]. The MPPI controller used in our experiments samples $1,000$ rollouts at each control iteration. Each rollout was simulated over a time horizon of 10 seconds with a sampling rate of 10 Hz, resulting in predictions at 0.1-second intervals.

The data processing pipeline for both models was identical. As input, we use real-time odometry data, including linear and angular velocities, along with the commanded velocities. Using this input, we compute forward passes through the respective learned dynamics models. These predictions are then used to generate trajectory rollouts within the MPPI-based model predictive control algorithm. Finally, control inputs are computed from the rollouts following the standard MPPI procedure.

C. Training Results

We plot the training curves in Figure 4. The results indicate that the function encoder slightly outperforms neural ODEs on in-distribution scenes. However, for an out-of-distribution scene that is an interpolation of two of the training scenes, the neural ODE's performance degrades significantly. This is because the neural ODE is overfitting to the training scenes, whereas the function encoder adapts to this new scene via its coefficient calculation. This effect is even more pronounced for the held-out, extrapolation scene, where the neural ODE's performance suffers. Additionally, the training curve on the extrapolation dataset is indicative of overfitting for both algorithms.

D. Generalization and Online Adaptation Across Scenes

To highlight our approach's ability to adapt to new settings, we mimic an online setting and measure the MSE of the next state predictions with only recent data to compute the coefficients. The results are shown in Figure 5. We iterate over the data for each scene, and at any given timestep, we use the previous n data points to compute the coefficients. The resulting loss for the function encoder is the same as during training, suggesting that the function encoder is successfully adapting from the most recent n data points. Furthermore, this curve ablates the amount of data needed to compute the coefficients. We find that 5 data points, which is approximately half a second, is enough for the function encoder to outperform neural ODEs. The function encoder's performance is stable as the amount of example data increases, suggesting that the coefficient estimate's accuracy has saturated.

In Figure 6, we plot the error per scene. We see that the function encoder slightly outperforms neural ODEs on all scenes, which corresponds with the training curves. Furthermore, we see that neural ODEs suffer an increase in error for scene 5, which is the held-out, interpolation scene. This is because neural ODEs are overfitting to the training scenes, whereas the function encoder uses a small dataset to calibrate to this new scene. Both approaches have worse performance on the extrapolation scene (Scene 1), although the function encoder outperforms neural ODEs on the validation set.

In Figures 7 and 8, we evaluate the accumulated MSE of each model's predictions over time. To simulate online trajectory rollouts as in MPPI, we choose random initial states from the scene data, and the models propagate the state forward using the recorded control inputs. Then, we evaluate the MSE between the true and predicted state after each prediction. Figure 7 shows the results on the interpolation scene (Scene 5), and figure 8 shows the results on the extrapolation scene (Scene 1). In both plots, the neural

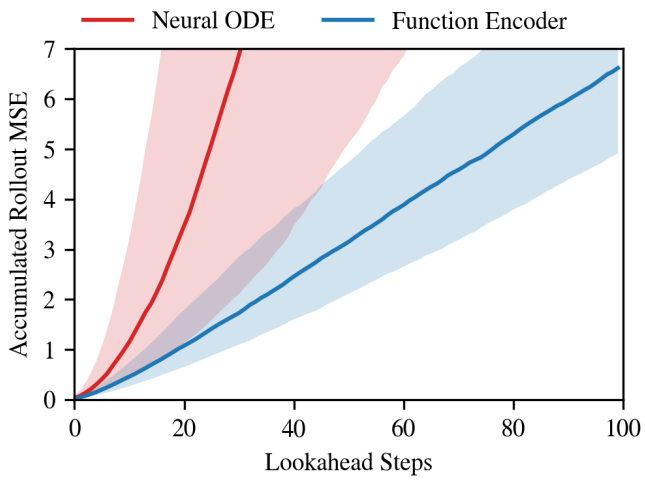

Fig. 7: Accumulated MSE over time for scene 5. Our approach achieves more accurate rollouts over time on the interpolation scene (Scene 5). We evaluate over 10 random seeds and 100 rollouts per seed. The median error values are bounded by the 90% and 10% quartiles (shaded regions).

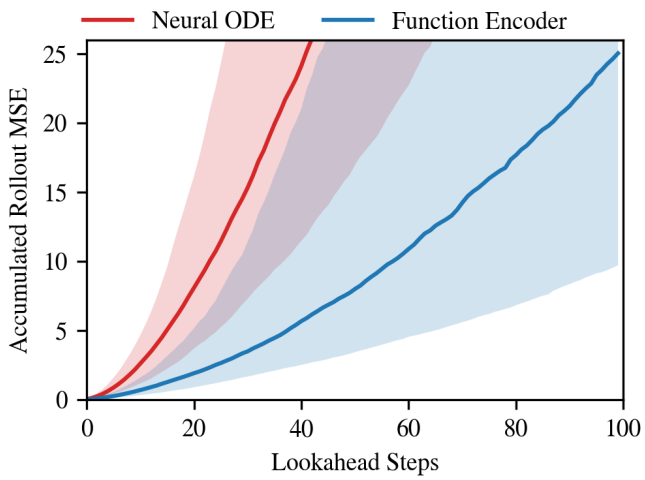

Fig. 8: Accumulated MSE over time for Scene 1 (icy).

ODE accumulates prediction error at a much faster rate than the function encoder. This indicates that the function encoder produces more accurate MPPI rollouts over time.

E. Qualitative Results

We deploy the trained neural ODE and function encoder models on the simulated Warthog ground robot in Phoenix. The robot must navigate through a wooded area to reach a series of waypoints without colliding with trees. MPPI controls the robot during waypoint navigation using a learned model to calculate trajectory rollouts. Figures 9 and 10 compare how the neural ODE and the function encoder perform in this navigation experiment while driving on the unknown ice terrain (Scene 1 in Figure 6). The blue circle shows the

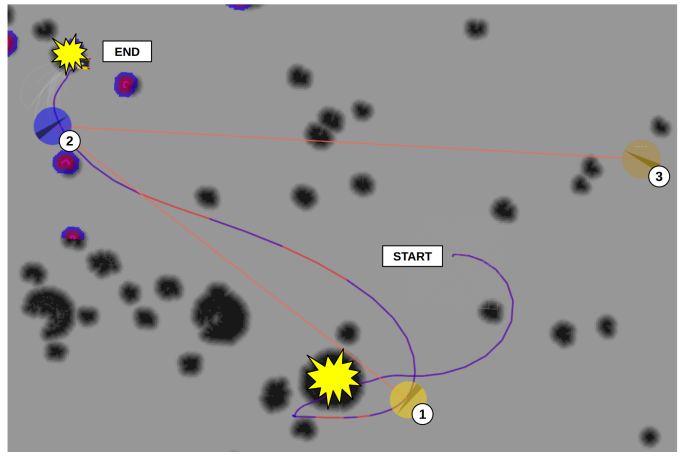

Fig. 9: The neural ODE struggles to extrapolate to an unknown icy terrain. Due to poor rollout predictions, the robot collides with two trees and fails the mission.

proximity radius of each waypoint, and the thin purple line shows the path along which the robot traveled. As expected, the robot using the neural ODE model in Figure 9 fails to extrapolate to the unknown ice terrain, where the surface friction in Phoenix is set extremely low. The robot only reached two waypoints because it slides out of control, collides with a tree, and was unable to recover. On the other hand, the robot using the function encoder model in Figure 10 reaches all three waypoint goals without colliding with any obstacles. Over four trials, the function encoder had no collisions, while the neural ODE had 11 total.

Figure 11 illustrates each model's prediction accuracy in a timelapse. MPPI attempts to control the robot through an arc maneuver while traveling at high speeds on a slippery terrain. This is very challenging due to low friction between the tires and the ground that causes the robot to slide through turns. The function encoder better predicts how the robot slides while turning and maintains control of the vehicle. On the other hand, the neural ODE predicts with less accuracy, and MPPI loses control.

V. Limitations

The main limitations of our approach are related to the data and the limitation of our validation results to a simulated robot environment.

Sensor Modality: The current implementation relies exclusively on proprioceptive measurements (e.g. odometry and commanded velocities). Consequently, adaptation is inherently reactive, meaning it occurs only after the robot has physically encountered the terrain. This limits the ability to anticipate upcoming terrain transitions and can result in control degradation, particularly during abrupt surface changes. Nevertheless, we anticipate that our approach could likely be adapted to streaming data via recursive least squares or Bayesian approaches to update the coefficients in response to new data.

Model Expressiveness and Excitation: The accuracy of the adapted model is constrained by the span of the learned

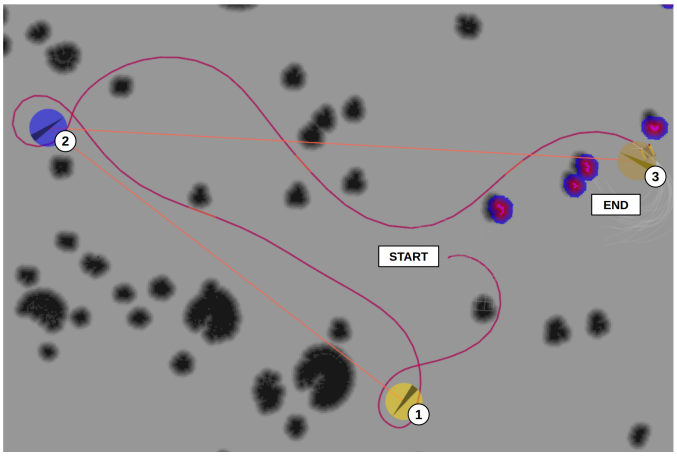

Fig. 10: The function encoder successfully extrapolates to an unknown icy terrain, avoids all obstacles, and completes the mission.

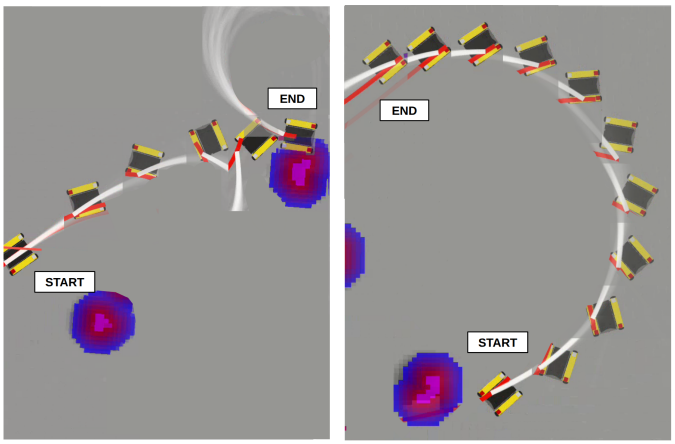

Fig. 11: Timelapse illustration of model predictions. The neural ODE trajectory predictions break down during a high speed turn (left). As a result, MPPI loses control of the vehicle and collides with an obstacle. The function encoder accurately predicts the robot's trajectory during a similar maneuver (right) and maintains control.

function space. If terrain-induced dynamics fall outside this span—e.g. due to highly nonlinear effects like slippage or discontinuous contacts—the model may fail to generalize, limiting performance in extreme conditions.

At runtime, the method requires only a small number of transitions (e.g. $\sim$100) to estimate terrain-specific coefficients via least-squares. However, this estimation assumes persistently exciting control inputs; under low-excitation conditions, such as steady, straight-line driving, the resulting regression problem may be ill-conditioned, reducing adaptation quality.

Simulation-Only Validation: All experiments are conducted in high-fidelity simulation using a Unity-based physics engine with realistic terrain modeling. While these results are promising, real-world validation on hardware is needed to assess performance and robustness under real-world conditions.

VI. Conclusion & Future Work

This work introduces a method for online adaptation of terrain-aware dynamics using function encoders, which represent robot dynamics as a linear combination of basis functions learned from diverse terrain data. By projecting dynamics onto a compact function space representation and estimating terrain-specific coefficients online via least squares regression, the method achieves fast and data-efficient adaptation without retraining or fine-tuning. Integrated into a model predictive path integral (MPPI) controller, the adapted model enables accurate trajectory prediction and robust closed-loop performance on previously unseen terrain. Experiments in a high-fidelity Unity simulation with a Clearpath Warthog robot show that our approach reduces model prediction error and results in fewer collisions compared to a neural ODE baseline, demonstrating its effectiveness for terrain-aware planning in unstructured environments.

Future work aims to incorporate visual perception into the adaptation process, enabling anticipatory, zero-shot dynamics estimation directly from visual terrain features. Specifically, we plan to map embeddings from onboard RGB cameras to dynamics representations in the learned function space, enabling the robot to infer terrain dynamics before contact. We also plan to extend this work to other types of terrain variations. This work focuses on varying friction parameters, but future work seeks to evaluate our method on uneven, bumpy surfaces that are common in real-world, off-road navigation.

Acknowledgments

This work was supported by the National Science Foundation under NSF Grant Numbers 2214939 and 2409535. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views the National Science Foundation.

This work was also supported by the Air Force Office of Scientific Research under award FA9550-19-1-0005, as well as the Office of Naval Research under award ONR N00014-20-1-2115. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views the U.S. Department of Defense.

Research reported in this paper was sponsored in part by the DEVCOM Army Research Laboratory under Cooperative Agreement W911NF23-2-0211. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the DEVCOM Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

References

[1] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy

Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[2] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[3] Arunkumar Byravan, Felix Leeb, Franziska Meier, and Dieter Fox. Se3-pose-nets: Structured deep dynamics models for visuomotor planning and control. *arXiv preprint arXiv:1710.00489*, 2017.

[4] Mateo Guaman Castro, Samuel Triest, Wenshan Wang, Jason M Gregory, Felix Sanchez, John G Rogers, and Sebastian Scherer. How does it feel? self-supervised costmap learning for off-road vehicle traversability. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 931–938. IEEE, 2023.

[5] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

[6] Calvin Chanyoung Chung, Andrea Finazzi, Hyunki Seong, Daegyu Lee, Seungwook Lee, Bosung Kim, Gyuri Gang, and David Hyunchul Shim. Autonomous system for head-to-head race: Design, implementation and analysis; team kaist at the indy autonomous challenge. *Transactions on Field Robotics*, pages 1–1, 2024. doi: 10.1109/TFR.2024.3497922.

[7] James Dallas, Kshitij Jain, Zheng Dong, Leonid Sapronov, Michael P Cole, Paramsothy Jayakumar, and Tulga Ersal. Online terrain estimation for autonomous vehicles on deformable terrains. *Journal of Terramechanics*, 91:11–22, 2020.

[8] Mauro Dimastrogiovanni, Florian Cordes, and Giulio Reina. Terrain estimation for planetary exploration robots. *Applied Sciences*, 10(17), 2020. ISSN 2076-3417. doi: 10.3390/app10176044. URL https://www.mdpi.com/2076-3417/10/17/6044.

[9] 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.

[10] Franck Djeumou, Cyrus Neary, and Ufuk Topcu. How to learn and generalize from three minutes of data: Physics-constrained and uncertainty-aware neural stochastic differential equations, 2023. URL https://arxiv.org/abs/2306.06335.

[11] Franck Djeumou, Thomas Jonathan Lew, Nan Ding, Michael Thompson, Makoto Suminaka, Marcus Greiff, and John Subosits. One model to drift them all: Physics-informed conditional diffusion model for driving at the limits. In *8th Annual Conference on Robot Learning*, 2024. URL https://openreview.net/forum?id=0gDbaEtVrd.

[12] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

[13] Albert A Espinoza, Jorge L Torres-Filomeno, Karla M Montañez-Sánchez, and Ángel J Ortiz-Andújar. Vehicle-terrain parameter estimation for small-scale unmanned tracked vehicles. In *2019 IEEE International Symposium on Measurement and Control in Robotics (ISMCR)*, pages A3–5. IEEE, 2019.

[14] 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.

[15] Jonas Frey, Manthan Patel, Deegan Atha, Julian Nubert, David Fan, Ali Agha, Curtis Padgett, Patrick Spieler, Marco Hutter, and Shehryar Khattak. RoadRunner—learning traversability estimation for autonomous off-road driving. *Transactions on Field Robotics*, 1:192–212, 2024. doi: 10.1109/TFR.2024.3464369.

[16] Morgan T Gillespie, Charles M Best, Eric C Townsend, David Wingate, and Marc D Killpack. Learning nonlinear dynamic models of soft robots for model predictive control with neural networks. In *2018 IEEE International Conference on Soft Robotics (RoboSoft)*, pages 39–45. IEEE, 2018.

[17] Mateo Guaman Castro, Samuel Triest, Wenshan Wang, Jason M. Gregory, Felix Sanchez, John G. Rogers III, and Sebastian Scherer. How does it feel? self-supervised costmap learning for off-road vehicle traversability. In *International Conference on Robotics and Automation*. IEEE, 2023.

[18] Tyler Han, Alex Liu, Anqi Li, Alex Spitzer, Guanya Shi, and Byron Boots. Model predictive control for aggressive driving over uneven terrain. In *Robotics: Science and Systems Conference*, 2024.

[19] Tyler Ingebrand, Adam J. Thorpe, and Ufuk Topcu. Zero-shot transfer of neural ODEs. In *Advances in Neural Information Processing Systems*, volume 37, pages 67604–67626. Curran Associates, Inc., 2024.

[20] Tyler Ingebrand, Amy Zhang, and Ufuk Topcu. Zero-shot reinforcement learning via function encoders. In *International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 21007–21019. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/ingebrand24a.html.

[21] Tyler Ingebrand, Adam J. Thorpe, Somdatta Goswami, Krishna Kumar, and Ufuk Topcu. Basis-to-basis operator learning using function encoders. *Computer Methods in Applied Mechanics and Engineering*, 435:117646, 2025. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2024.117646. URL https://www.sciencedirect.com/science/article/pii/S0045782524009009.

[22] Tyler Ingebrand, Adam J. Thorpe, and Ufuk Topcu. Function encoders: A principled approach to transfer

learning in hilbert spaces. *ICML*, 2025.

[23] Sanghun Jung, Joonho Lee, Xiangyun Meng, Byron Boots, and Alexander Lambert. V-STRONG: visual self-supervised traversability learning for off-road navigation. In *IEEE International Conference on Robotics and Automation*, pages 1766–1773. IEEE, 2024.

[24] Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. RMA: rapid motor adaptation for legged robots. In Dylan A. Shell, Marc Toussaint, and M. Ani Hsieh, editors, *Robotics: Science and Systems*, 2021. doi: 10.15607/RSS.2021.XVII.011.

[25] Jean-François Lalonde, Nicolas Vandapel, Daniel F Huber, and Martial Hebert. Natural terrain classification using three-dimensional ladar data for ground robot mobility. *Journal of field robotics*, 23(10):839–861, 2006.

[26] Yuankai Li, Liang Ding, Zhizhong Zheng, Qizhi Yang, Xingang Zhao, and Guangjun Liu. A multi-mode real-time terrain parameter estimation method for wheeled motion control of mobile robots. *Mechanical Systems and Signal Processing*, 104:758–775, 2018.

[27] Bing Liu. Learning on the job: Online lifelong and continual learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13544–13549, 2020.

[28] Bo Liu, Xuesu Xiao, and Peter Stone. A lifelong learning approach to mobile robot navigation. *IEEE Robotics and Automation Letters*, 6(2):1090–1096, 2021.

[29] Daniel Maturana, Po-Wei Chou, Masashi Uenoyama, and Sebastian Scherer. Real-time semantic mapping for autonomous off-road navigation. In *International Conference on Field and Service Robotics*, pages 335 – 350, September 2017.

[30] Dave Mechergui and Paramsothy Jayakumar. Efficient generation of accurate mobility maps using machine learning algorithms. *Journal of Terramechanics*, 88:53–63, 2020.

[31] Xiangyun Meng, Nathan Hatch, Alexander Lambert, Anqi Li, Nolan Wagener, Matthew Schmittle, Joonho Lee, Wentao Yuan, Zoey Qiuyu Chen, Samuel Deng, Greg Okopal, Dieter Fox, Byron Boots, and Amirreza Shaban. Terrainnet: Visual modeling of complex terrain for high-speed, off-road navigation. In *Robotics: Science and Systems XIX*, 2023.

[32] Benjamin Morrell, Rohan Thakker, Àngel Santamaria Navarro, Amanda Bouman, Xiaoming Lei, Jeffrey Edlund, Torkom Pailevanian, Tiago Stegun Vaquero, Yun Lin Chang, Thomas Touma, et al. NeBula: Team CoSTAR's robotic autonomy solution that won phase ii of DARPA subterranean challenge. *Field robotics*, 2: 1432–1506, 2022.

[33] Xiaobo Nie, Chuan Min, Yongjun Pan, Ke Li, and Zhixiong Li. Deep-neural-network-based modelling of longitudinal-lateral dynamics to predict the vehicle states for autonomous driving. *Sensors*, 22(5):2013, 2022.

[34] Lauro Ojeda, Johann Borenstein, Gary Witus, and Robert Karlsen. Terrain characterization and classification with a mobile robot. *Journal of field robotics*, 23(2):103–122, 2006.

[35] Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

[36] Chandramouli Padmanabhan, Sayan Gupta, Annadurai Mylswamy, et al. Estimation of terramechanics parameters of wheel-soil interaction model using particle filtering. *Journal of Terramechanics*, 79:79–95, 2018.

[37] Jesse Pentzer, Sean Brennan, and Karl Reichard. Online estimation of vehicle motion and power model parameters for skid-steer robot energy use prediction. In *2014 American Control Conference*, pages 2786–2791, 2014.

[38] Giulio Reina, Antonio Leanza, and Arcangelo Messina. Terrain estimation via vehicle vibration measurement and cubature kalman filtering. *Journal of Vibration and Control*, 26(11-12):885–898, 2020.

[39] Amirreza Shaban, Xiangyun Meng, JoonHo Lee, Byron Boots, and Dieter Fox. Semantic terrain classification for off-road autonomous driving. In Aleksandra Faust, David Hsu, and Gerhard Neumann, editors, *Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 619–629. PMLR, 08–11 Nov 2022. URL https://proceedings.mlr.press/v164/shaban22a.html.

[40] Nathan A Spielberg, Matthew Brown, Nitin R Kapania, John C Kegelman, and J Christian Gerdes. Neural network vehicle models for high-performance automated driving. *Science robotics*, 4(28):eaaw1975, 2019.

[41] Austin Stone, Ted Xiao, Yao Lu, Keerthana Gopalakrishnan, Kuang-Huei Lee, Quan Vuong, Paul Wohlhart, Sean Kirmani, Brianna Zitkovich, Fei Xia, et al. Open-world object manipulation using pre-trained vision-language models. *arXiv preprint arXiv:2303.00905*, 2023.

[42] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

[43] Unity Technologies. Wheel collider friction. https://docs.unity3d.com/6000.0/Documentation/Manual/wheel-colliders-friction.html, 2025. Accessed 21 Apr. 2025.

[44] Adam J. Thorpe, Tyler Ingebrand, Somdatta Goswami, Krishna Kumar, and Ufuk Topcu. Basis-to-basis operator learning: A paradigm for scalable and interpretable operator learning on hilbert spaces. In *2025 SIAM Conference on Computational Science and Engineering*, 2025.

[45] Kasi Viswanath, Kartikeya Singh, Peng Jiang, PB Sujit, and Srikanth Saripalli. Offseg: A semantic segmentation framework for off-road driving. In *International Conference on Automation Science and Engineering*, pages 354–359. IEEE, 2021.

[46] Maggie Wigness, Sungmin Eum, John G Rogers, David Han, and Heesung Kwon. A RUGD dataset for autonomous navigation and visual perception in unstructured outdoor environments. In *International Conference on Intelligent Robots and Systems*, pages 5000–5007. IEEE, 2019.

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

[48] Grady Williams, Nolan Wagener, Brian Goldfain, Paul Drews, James M. Rehg, Byron Boots, and Evangelos A. Theodorou. Information theoretic mpc for model-based reinforcement learning. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1714–1721, Singapore, Singapore, May 2017. IEEE. doi: 10.1109/ICRA.2017.7989202.

[49] Zhou Xian, Shamit Lal, Hsiao-Yu Tung, Emmanouil Antonios Platanios, and Katerina Fragkiadaki. Hyperdynamics: Meta-learning object and agent dynamics with hypernetworks. In *International Conference on Learning Representations*. OpenReview.net, 2021.

[50] Xuesu Xiao, Bo Liu, Garrett Warnell, and Peter Stone. Motion planning and control for mobile robot navigation using machine learning: a survey. *Autonomous Robots*, 46(5):569–597, 2022.