

Structured Perception Representations for Off-road Dynamics Learning

Micah Nye¹, Samuel Triest¹, Matthew Sivaprakasam¹,
Isaiah Adu^{1,2}, Sebastian Scherer¹, Wenshan Wang¹

¹The Robotics Institute, Carnegie Mellon University, ²Pennsylvania State University

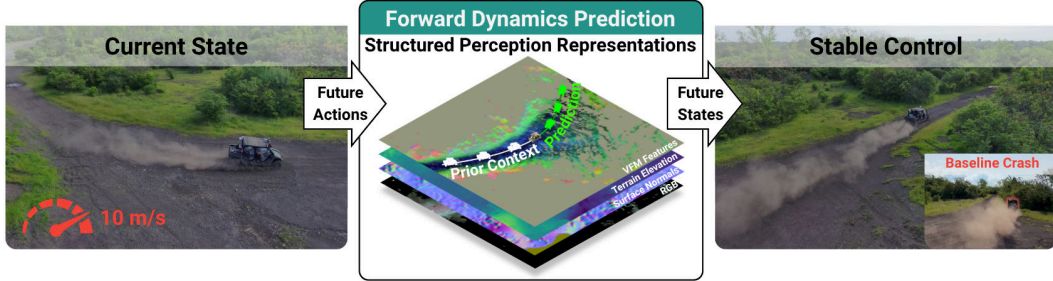


Figure 1: A learned dynamics model, grounded by structured perception representations. By leveraging the spatial structure of bird’s-eye-view maps, we are able to decrease worst-case trajectory prediction error by 17.1% from learned and classical baselines at speeds in excess of 10 m/s.

Abstract: Autonomously navigating off-road terrain at high speed demands accurate control, which often relies on dynamics models that capture complex interactions between the robot and terrain. In off-road regimes, terrain interaction is a major component of the vehicle’s dynamics. Prior work attempts to capture this effect by training learned models of the vehicle’s dynamics, conditioned on its perception of the local environment. However, said conditioning is often overly simplistic, ignoring the spatial structure of the dynamics problem. Our approach, SpatialBEV, exploits this spatial structure. At each step, we query the bird’s-eye-view (BEV) features around the predicted state, conditioning the next prediction on current geometric and semantic cues. We additionally present a benchmarking approach that mines a dataset’s most difficult scenarios. On datasets collected across seasons, SpatialBEV reduces prediction error by 17.1%, compared to physics-based and learned baselines. We further evaluate on hardware on high-speed ($\geq 10\text{m/s}$) and geometrically challenging courses with a 25.3% reduction in maximum cross-track error. More details can be found at <https://theairlab.org/spatial-bev-dynamics>.

Keywords: Learning for Control, Dynamics Modeling, Perception Representations, Off-road Navigation

1 Introduction

Autonomous navigation in unstructured, off-road environments at high speeds remains a significant challenge for model-predictive control (MPC) frameworks. MPC is a well-established method in control theory; however, its success depends on accurate dynamics models for forward prediction. Traditional methods rely on simplified physics models [1, 2, 3, 4, 5] that struggle to capture complex terradynamics induced by challenging terrains like loose gravel, deep snow, or standing water. High speeds exacerbate these unmodeled interactions which may lead to task failure.

While learned dynamics models have emerged as a potential solution due to their expressivity, they often neglect environment observation [6, 7]. When a robot commands identical actions under iden-

tical initial conditions on different surfaces with differing geometry or friction, the resulting trajectory varies significantly [8, 9]. In state-only models, this variance is captured as an unmodeled disturbance term or stochastic noise term. However, this uncertainty is largely epistemic; it is a direct consequence of the model’s inability to observe the environmental physics imposed on the robot during interaction.

Motivated by this shortcoming, we propose a dynamics representation that conditions its prediction on environment geometry and visual semantics. We use birds-eye-view (BEV) representations which provide a spatially-aligned structure for environment attributes such as terrain elevation and Visual Foundation Model (VFM) features. While prior works encode BEV features as a fixed contextual prior at the beginning of a prediction pass [10, 11, 12], this leaves the spatial representation underutilized. Instead of treating the BEV information as a single prior, we make full use of its spatial structure by treating it as a continuous source for incremental feedback and state correction at every prediction step. This explicit architecture ensures the most relevant BEV features are queried at each step. This is especially important at high speed, where a robot traverses large distances and fixed-prior approaches condition every prediction step on the same initial encoding, rather than the features local to the robot’s current position.

We demonstrate the efficacy of this representation through extensive evaluation. We also introduce a systematic testing methodology that clusters similar actions to isolate high-variance regimes, specifically targeting the scenarios where current models are likely to fail and where environment context is necessary for accurate dynamics prediction. Our results span both offline evaluation on aggressive driving data around multiple test sites as well as online hardware deployment on high speed ($\geq 10\text{m/s}$) and geometrically challenging courses. Through all experiments, we show that our spatial representation for incremental conditioning reduces forward prediction error and improves closed-loop tracking. Specifically our work provides three key contributions:

1. A new method for leveraging structured perception representations for environment context in dynamics modeling
2. A benchmarking approach to mine the most challenging sequences of dynamics datasets
3. Extensive offline experiments and online hardware experiments at high speeds ($\geq 10\text{m/s}$), demonstrating the efficacy of our approach

2 Related Works

Both the learning objective and representations for the dynamics model are critical for robust control on varying terrains. Djeumou et al. [13] choose to learn the parameters of a tire model, which grants a structured learning paradigm with natural saturation in physically infeasible regimes, but also hinges on the accuracy of this physical assumption. Such models struggle on off-road terrains, with difficult-to-model nonlinearities imposed by undulated terrain surfaces, which Gibson et al. [14] address by learning the force residual to a tire model. Dynamics residual learning supplements the physics model by learning any unmodeled disturbances and is used widely in literature for off-road platforms [4, 12, 11, 15, 6, 16], on-road platforms [17, 18], and aerial vehicles [19, 20], among other embodiments. Yet, these approaches depend on the physics model accuracy, require ample time to tune the physics model, and restrict the model to an explicit physical form factor.

Thus, much like model-based reinforcement learning [21, 22, 23], it is common to learn the forward states directly, without any physics-informed prior [24, 25, 26, 27, 28, 29, 30, 31]. These state representations can be further formulated probabilistically [32, 7, 33, 34, 2, 35, 36, 37]. To improve terrain awareness, dynamics models can be augmented with geometric and semantic inputs using perception representations. Prior work conditions dynamics models on first-person-view (FPV) images [22, 38, 39, 40], point clouds [41], or BEV representations, specifically elevation maps [10, 14, 11], RGB maps [42, 16], or VFM feature maps [12, 14]. However, most prior work treats perception as a fixed context, encoded into the latent state once at rollout start [10, 12]. Lee et al. [11] reduce the encoder load via a BEV crop around a conservative feasible motion extent. Most similar to our

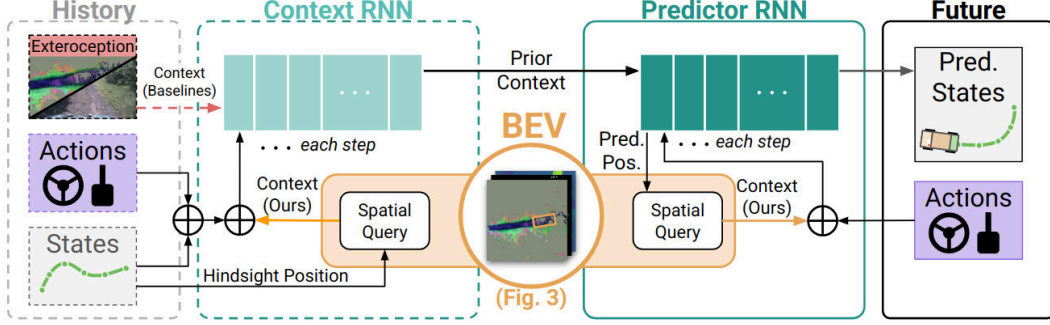


Figure 2: An overview of our dynamics model. We query each prior or predicted position in the static, structured BEV map at each step. This provides geometric and semantic context to our model for more accurate dynamics rollouts. Figure 3 details the structured representation formally.

approach, Gibson et al. [14] query features at tire points along the rollout; however, they use this context as a physics residual and no further. In contrast, our approach takes full advantage of the context by using it for both state prediction *and correction*, which contributes substantially to prediction accuracy. Beyond the architecture, we formalize this spatially-grounded representation and isolate its benefit through extensive ablations and quantitative & qualitative evaluations.

3 Method

We aim to formulate a dynamics model that leverages perception to more accurately predict forward motion over challenging terrain before traversing it. We first detail the proprioceptive forward dynamics backbone (Sec. 3.1), then describe our method for leveraging BEV representations that better exploit spatial structure (Sec. 3.2). Figures 2 and 3 illustrate our approach.

Preliminaries Given a current state $\mathbf{x}_t \in \mathcal{X}$ and action $\mathbf{u}_t \in \mathcal{U}$ the next state $\mathbf{x}_{t+1}$ is governed by the deterministic robot dynamics f_θ such that $\mathbf{x}_{t+1} = f_\theta(\mathbf{x}_t, \mathbf{u}_t)$. For a sequence of T actions forward in time $\mathbf{u}_{t:t+T}$, f_θ predicts the future states $\mathbf{x}_{t+1:t+T+1}$. We augment the model with proprioceptive context of K prior timesteps of state and action history $c_t = [\mathbf{x}_{t-K:t}, \mathbf{u}_{t-K:t-1}]$. Additionally, to capture terrain and environment conditions not encoded in state alone, the robot receives a local observation $o_t \in \mathcal{O}$ from onboard sensors, such as a BEV map. Together, the robot dynamics follow $\mathbf{x}_{t+1} = f_\theta(\mathbf{x}_t, \mathbf{u}_t, c_t, o_t)$.

3.1 Proprioceptive Dynamics Model

We architect the core dynamics model as a Recurrent Neural Network [43]. Similar to Gibson et al. [4], we compose it of two components: 1) the predictor network composed of RNN cells Φ_P which takes an action sequence with prior context to predict the future states and 2) a context network composed of RNN cells Φ_C which initializes the prior context for the predictor network through state history, action history, and current observations. We choose to use an RNN rather than a Transformer [44] because its sequential recurrence lets us insert our structured spatial representation between cells, while keeping the memory footprint small.

The predictor cell for time t takes action $\mathbf{u}_t$, encoded as $\mathbf{z}_t$, as input with prior latent state $\mathbf{h}_t$ to predict the next hidden state and recurrently produces this output for $\mathbf{u}_{t:t+T}$ (eq. 1). Like Nagabandi et al. [23], we avoid predicting state derivatives for numerical integration. Instead, we predict the state differences $\Delta \mathbf{x}_t$ over a time step, so the dynamics function follows $\mathbf{x}_{t+1} = \mathbf{x}_t + f_\theta(\mathbf{x}_t, \mathbf{u}_t, c_t, o_t)$ (eq. 1). This localizes the target near zero for a better learnable signal.

We provide proprioceptive context to the predictor through the context network cells Φ_C . Taking the state and action history $\mathbf{x}_{t-K:t}$ and $\mathbf{u}_{t-K:t-1}$, the context network initializes the predictor's hidden state $\mathbf{h}_0$ (eq. 2) and, importantly, trains the shared state and action encoders. While purely

proprioceptive architectures like this suffice in structured domains with benign terrain, off-road environments with undulating terrain and variable ground cover need perception for greater information on the terrain ahead.

$$\mathbf{z}_t = \phi^u(\mathbf{u}_t), \quad \mathbf{h}_{t+1} = \Phi_P(\mathbf{h}_t, \mathbf{z}_t, c_t), \quad \text{for } t = 0, \dots, T, \quad \mathbf{x}_{t+1} = \mathbf{x}_t + \phi^x(\mathbf{h}_{t+1}) \quad (1)$$

$$\mathbf{h}_{k+1} = \Phi_C(\mathbf{h}_k, \mathbf{z}_k, c_k), \quad \text{for } k = -K, \dots, -1, \quad \mathbf{h}_{-K} \triangleq \vec{0} \quad (2)$$

3.2 Structured Perception Representations for Environment Context

Perception remains an underutilized modality for dynamics prediction, despite its ability to expose otherwise unobserved semantic and geometric priors that affect future robot locomotion [9]. Prior work [10, 16, 11, 12] has incorporated perceptual conditioning through inputs such as FPV images or BEV maps as fixed context signals. These approaches do not properly exploit the spatial structure of the BEV representation; we can leverage the spatial organization to get more informative, localized context at each prediction step. Through structured BEV representations, we embed geometric cues in the form of terrain height estimation and normal layers, alongside visual semantic features from VFMs. We construct these representations through a streamlined perception pipeline relying on LiDAR pointclouds for geometric features and map registration and FPV imagery for semantic features, detailed in Section C.1.1.

Recurrent BEV Cells Prior work concatenates a flattened BEV latent to the initial RNN hidden state. While this improves performance, it forces the network to learn spatial indexing from an unstructured vector. It discards the spatial prior that the most relevant BEV features at step $t + 1$ are those the robot is about to physically interact with, from step t . We instead intercept the connection between consecutive RNN memory cells with a new recurrent BEV cell that injects this structured prior (Figure 3).

Concretely, let $\mathbf{B} \in \mathbb{R}^{W \times H \times F}$ denote the BEV feature map and $\hat{\mathbf{h}}_{t+1}$ the hidden state at step $t + 1$ from cell t . We decode the hidden state into a state difference $\Delta \mathbf{x}_t$ such that $\hat{\mathbf{x}}_{t+1} = \mathbf{x}_t + \phi^x(\hat{\mathbf{h}}_{t+1})$. We define a set of tire track coordinates for our wheeled platform as the BEV footprint $\mathcal{F}(\hat{\mathbf{x}}_{t+1}) \subset \mathbb{R}^2$ (Sec. D.2) and use it to sample over $\mathbf{B}$ to obtain the BEV feature context vector in eq. 3 where $\mathbf{B}[\cdot]$ denotes bilinear interpolation of the features in the BEV grid at the footprint points and ϕ^b is a learned encoder. This context is used in two key manners. Most importantly, it applies a corrective update to the hidden state $\mathbf{h}_{t+1} = \hat{\mathbf{h}}_{t+1} + \mathbf{b}_{t+1}$, such that each state prediction is informed by its current context vector at t alongside its predicted context vector at $t + 1$. Second, it augments the RNN cell’s input latent action $\mathbf{z}_{t+1}$, so the next state has its current BEV context. For Φ_C , we forgo the correction step and use the history of known states $\mathbf{x}_{t-K:t}$ for query. Formally, our structured perception representation is defined as

$$\mathbf{b}_{t+1} \triangleq \phi^b(\text{vec}(\mathbf{B}[\mathcal{F}(\hat{\mathbf{x}}_{t+1})])), \quad (3)$$

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \phi^x(\Phi_P(\mathbf{h}_t, \mathbf{z}_t, \mathbf{b}_t) + \mathbf{b}_{t+1}) \quad (4)$$

This construction can be viewed as a variant of spatial attention, differing in how much of the sampling location is fixed by geometry rather than learned. Standard attention over the BEV map [45] assigns learned weights to every spatial location and recovers footprint locality through training. Learned offset approaches such as deformable attention [46] learn to predict where to sample. Our formulation reflects that of Jaderberg et al. [47], with a sampling grid fixed by construction. While our footprint corresponds to a wheeled platform, the sampling grid is the only platform-specific component; replacing $\mathcal{F}$ with the contact pattern of a quadrupedal, bipedal, or tracked system leaves the rest of the architecture unchanged.

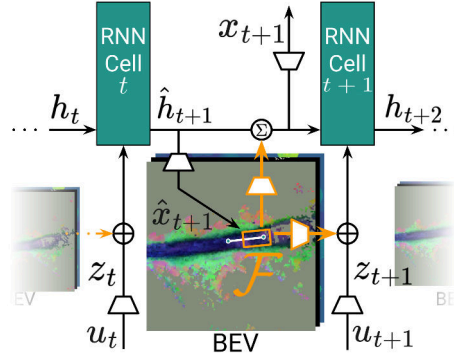


Figure 3: Recurrent BEV Cells. At each step, the next predicted state determines a footprint over the BEV map. Sampled features are passed into the subsequent cell.

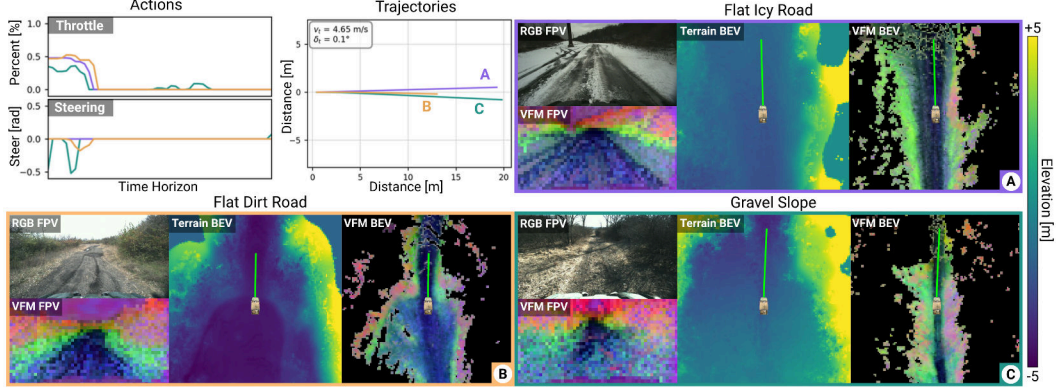


Figure 4: Similarly clustered actions yield different trajectories. Perception highlights how terradynamics are causing this difference. (C) travels further, going down a hill, as seen by the terrain map, (B) moves the least on flat dirt, while (A) travels further on flat, low-friction ice.

4 Benchmarking

Dynamics models must be rigorously evaluated before deployment in the field. Aggregate metrics over large datasets $\mathcal{D}$ are dominated by easy, benign interactions and mask long-tail failure modes that determine real-world safety. We therefore mine the dataset for its hardest samples, enabling targeted evaluation on the state sequences models are most likely to mispredict. Intuitively, an identical action sequence, executed from an identical initial condition in an identical environment, will yield an identical trajectory. In reality however, environments are rarely identical, which motivates this work to learn environment terradynamics so as not to defer to epistemic uncertainty. Terradynamics are most challenging to model for off-road driving [48, 2, 3], so our goal is to formulate a dataset $\mathcal{D}_{\text{hard}} \subset \mathcal{D}$ that maximizes trajectory variance under similar actions, attempting to highlight where terradynamics are most prevalent.

We define similarity between two candidates by the $\mathcal{L}_2$ distance between their normalized action sequences and initial conditions, connecting pairs that fall within a threshold and partitioning the resulting graph into cliques. The threshold is selected by maximizing the mean silhouette coefficient [49]. For a given group $\mathcal{G}$ of N samples, the mean pairwise Hausdorff distance $\bar{d}_H$ is computed across all trajectories in the group as a measure of trajectory variance and difficulty score (eq. 5). Figure 4 shows an example group of similar candidates, where a difference in trajectories under similar actions can be attributed to downward slopes and ground friction of ice.

Sorting groups by this score ranks the dataset by difficulty. The benchmark defines $\mathcal{D}_{\text{hard}}$ as the union of the top p percent of ranked groups, where p controls how aggressively the long tail is targeted. The error metric itself is unchanged; only the distribution it is taken over differs (eq. 6). A model may match its baselines on $\text{err}_{\mathcal{D}}$ while err_{hard} diverges, revealing the long-tail failures that aggregate evaluation conflates with dominant benign data.

$$\bar{d}_H(\mathcal{G}) = \frac{1}{N(N-1)} \sum_{i=1}^N \sum_{j=1}^N d_H(\tau^i, \tau^j), \quad j \neq i \quad (5)$$

$$\text{err}_{\mathcal{D}} = \mathbb{E}_{\tau \sim \mathcal{D}} [\text{err}(\hat{\tau}, \tau)], \quad \text{err}_{\text{hard}} = \mathbb{E}_{\tau \sim \mathcal{D}_{\text{hard}}} [\text{err}(\hat{\tau}, \tau)] \quad (6)$$

5 Experiments

Using our benchmarking approach, we perform a series of online and offline evaluations across our dataset and real-world closed-loop control trials. We investigate our approach, following these key research questions:

RQ1: Do spatial perception representations and leveraging the spatial structure of the problem improve dynamics predictions?

RQ2: Can our benchmarking approach mine and expose long-tail failure cases?

RQ3: Does open-loop prediction error correlate to closed-loop control performance?

Training & Evaluation We collect data at two test sites: Gascola and Renegade. The approach is trained on 79.8k samples (2.2 hr) of autonomous and teleoperated navigation around Gascola. Importantly, this dataset is only collected in the spring, fall, and summer, with no snow data available. We present all results on a held-out dataset of 75.3k samples (2.1 hr) of autonomous and teleoperated navigation around Gascola and Renegade. Renegade includes extreme hills not seen in the training set, and 34% of the test set is composed of winter data with 6-9" deep snow and thick ice which introduces additional tire forces and reduced friction.

Our approach uses a Long Short-Term Memory (LSTM) [50], trained to predict 5 s into the future with a Δt of 0.1 s, given a BEV map with geometry and visual semantic features, alongside 5 s of prior historical state-action pairs. We train to minimize the Mean Squared Error (MSE) loss using the Adam optimizer [51]. Our state reflects an extended kinematic state and is comprised of pose in $SE(2)$, speed, current steer, and pitch: $\mathbf{x} = [x \ y \ \psi \ v \ \delta \ \phi]$.

Map Inputs Our BEV maps are 100×100 m at 40 cm resolution. We train and evaluate with either sparse maps, which only use aggregate lidar mapping (Sec. C.1.1), or dense maps, which use a learned map inpainting network from Triest et al. [52]. Using both sparse and dense maps highlights how different BEV-conditioned methods leverage the BEV features.

Baselines We evaluate our approach against 4 baselines: 1) a single-track extended kinematic bicycle model with actuator modeling and parameters fit by least squares to ground-truth trajectories; 2) a state-only LSTM (State), with state history context, as a faithful recreation of Gibson et al. [4] as a non-hybrid approach; 3) a FPV-conditioned LSTM (FPV), in which the FPV is encoded once at the start of propagation and combined with the state-history context; 4) a BEV-conditioned LSTM (Vanilla BEV), with the same, aforementioned initial encoding, similar to Roth et al. [10].

5.1 Effect of Structured Perception Representations

To answer **RQ1** and determine the benefit of structured perception representations, we evaluate our approach on the open-loop prediction training task, provide a qualitative analysis on the benefit of our representation, and ablate our architecture design.

Open-loop Prediction We first evaluate our approach on the training task of open-loop prediction with no closed-loop actuation using MSE on

the full and position state spaces. We evaluate on three sub-datasets: Snow, Slopes, and Speed, defined in Section B. The benefit of our structured representation is evident in Figure 6 and Table 1, where our approach with dense maps reduces prediction error relative to the strongest baseline of each sub-dataset by 17.1% on Snow, 8.5% on Slopes, 19.9% on Speed, and 17.1% on Full. Even with sparse maps, our approach outperforms baselines on Speed and Full datasets; however, FPV has lower error on Snow and Slopes. This is expected, since LiDAR coverage degrades on slopes, leaving sparse BEV maps least informative. Finally, comparing against our Vanilla BEV counterpart isolates the effect of our representation. Holding the map input fixed, we improve by

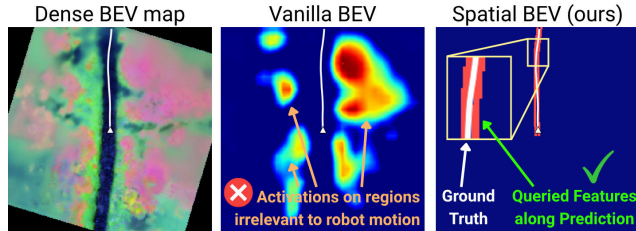


Figure 5: Vanilla BEV activations and our queries with reference BEV map and ground truth trajectory (white). Vanilla BEV’s activations fall on irrelevant regions, contrary to our queries along our prediction near the ground truth.

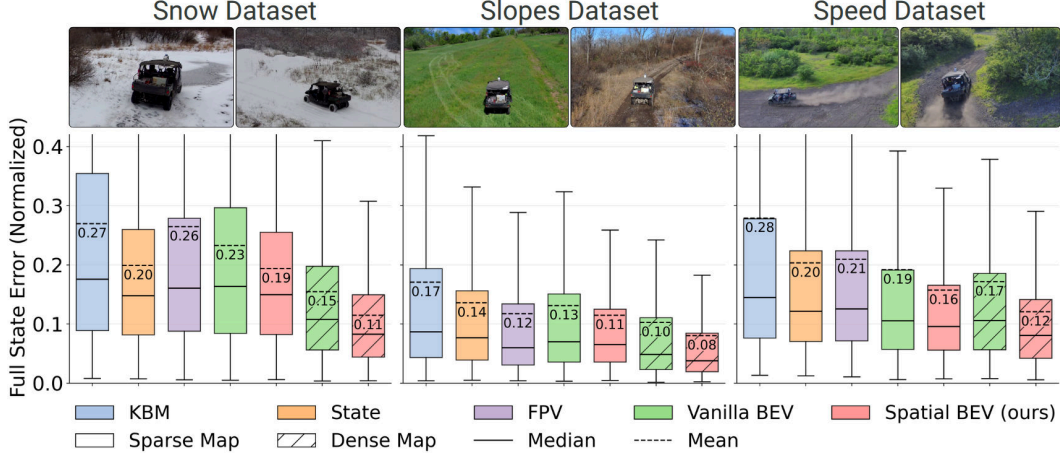


Figure 6: Offline full state prediction error distributions across 3 datasets. The Snow dataset is most challenging for all methods. Dense maps reduce error mean and distribution variance. Across all datasets, our method outperforms the baselines.

Method	Map	Snow		Slopes		Speed		Full	
		MSE [m ²] ↓	P95	MSE [m ²] ↓	P95	MSE [m ²] ↓	P95	MSE [m ²] ↓	P95
KBM	–	2.68 ± 4.23	8.92	2.31 ± 4.71	8.00	2.83 ± 6.52	10.24	2.10 ± 4.24	7.20
State	–	2.54 ± 2.52	7.62	2.20 ± 3.15	7.48	1.91 ± 3.69	7.39	1.99 ± 2.75	6.71
FPV	–	2.34 ± 2.37	6.93	1.69 ± 2.95	<u>5.82</u>	2.12 ± 4.55	7.82	1.77 ± 2.83	6.00
Vanilla BEV	□	2.49 ± 2.89	8.12	1.98 ± 3.39	7.20	1.59 ± 3.47	6.33	1.73 ± 2.87	6.66
Vanilla BEV	▨	2.10 ± 2.44	<u>6.76</u>	1.97 ± 3.28	7.70	1.41 ± 2.93	<u>5.17</u>	1.46 ± 2.50	<u>5.63</u>
Spatial BEV (ours)	□	2.45 ± 2.66	8.06	1.78 ± 2.51	6.87	1.50 ± 3.90	5.42	1.70 ± 2.68	6.26
Spatial BEV (ours)	▨	1.74 ± 1.98	5.55	1.51 ± 2.44	5.26	1.13 ± 2.95	3.58	1.21 ± 2.14	4.41

Table 1: Offline position prediction MSE across three datasets, alongside the full test set. BEV maps are sparse □ or dense ▨. Our spatial approach outperforms its Vanilla BEV counterpart with sparse and dense maps. With dense maps, our approach outperforms all baselines.

4.6% with sparse maps and 19.4% with dense maps on average, highlighting the benefit of our representation and addressing **RQ1**. Qualitative model rollouts can be found in Section A.1.

Qualitative Representation Analysis Intuitively, our structured representation queries local environment features which improves predictions, but we seek to explain *why* this intuition holds. We visualize the activation maps of Vanilla BEV and our explicit approach in Figure 5 and see Vanilla BEV’s activations fall on irrelevant map regions, whereas our queries fall along our prediction, which closely follows the robot’s true path. This affirms our intuition and illustrates the benefit of our representation. More examples and activation map computation details in Section D.3.

Architectural Decisions We motivate our architecture’s correction step and footprint via ablations in Table 2. The recurrent BEV cell correction reduces error on the full dataset by 22.4% on sparse maps and 4.7% on dense maps, with up to 30.8% and 12.1% on snow, and reduces error in all but one setting of dataset and map input. We attribute the smaller dense map gain to the richer input leaving less to correct. Next, we compare footprints described in Section D.2.1. The Tracks footprint reduces error on the full dataset by 14.2% over the next best footprint and is the best across all sub-datasets. Compared to 4-point, Tracks densely cover the tire

Ablations	Map	MSE [m ²] ↓			
		Snow	Slopes	Speed	Full
BEV Cell	Non-corr. □	3.54	2.18	1.89	2.19
	Corr. □	2.45	1.78	1.50	1.70
	Non-corr. ▨	1.98	1.56	1.11	1.27
	Corr. ▨	1.74	1.51	<u>1.13</u>	1.21
Footprint	4-point ▨	<u>1.79</u>	1.88	1.49	<u>1.41</u>
	Tracks ▨	1.74	1.51	1.13	1.21
	Full body ▨	2.05	1.68	1.35	1.50
	Lookahead ▨	1.87	2.03	1.54	1.55

Table 2: Ablations of the correction step (top) and footprint (bottom) on all datasets.

path, which matters most at speed (24.2% improvement), where the robot moves further than the map resolution per step. However, the coverage must be balanced; the Full body and Lookahead footprints query cells away from the tire path which dilutes the relevant information.

5.2 Mining Long-tail Failures

We want to validate our benchmarking method by confirming if we are able to mine the most challenging data $\mathcal{D}_{\text{hard}}$ from a broader dataset $\mathcal{D}$, answering **RQ2**. To do this, we must determine if we can identify the most challenging data. We partition our dataset sorted by trajectory variance into 10 equal partitions or deciles, such that the 1st decile should have the highest variance and 10th decile has lowest. We evaluate all methods on each decile in Figure 7 and find that all models’ worst-case prediction error is highest on the highest variance decile and lowest on the lowest variance decile with a clear downwards trend. This demonstrates how our benchmark can identify the highest variance runs and how this correlates to dynamics modeling difficulty. Moreover, on the most challenging partition, we see the FPV and State baselines have much higher error, highlighting the importance of evaluating on the long-tail.

5.3 Closed-loop Control on Hardware

We further evaluate our approach on hardware and investigate how prediction error is correlated with closed-loop performance to answer **RQ3**. We deploy our solutions on a full-sized Yamaha Viking ATV, retrofitted with autonomy sensors and actuators [53, 54].

For closed-loop evaluation, we use Model Predictive Path Integral (MPPI) control [55, 56] with 825 particles at 10 Hz, whose actions are executed by a 30 Hz low-level controller. MPPI follows a reference line subject to lateral acceleration constraints (Sec. C.1.2). We measure cross-track error (CTE) of all methods and number of interventions as a light heuristic. We were unable to run BEV-based methods with dense maps. We evaluate on two courses at Gascola, shown in Figure 8: 1) A high speed, 739 m course over flat gravel and dirt terrain with a target speed of 10 m/s (1 trial); 2) A 524 m course over challenging terrain, such as steep slopes and banked turns, with a target speed of 8 m/s (4 trials).

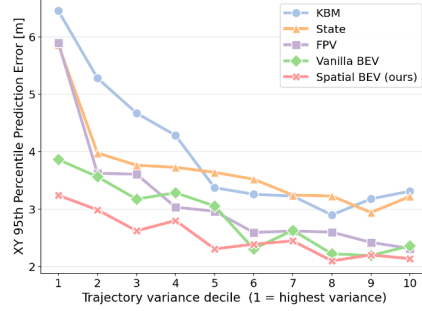


Figure 7: Evaluation on the highest to lowest variance deciles of the clustered benchmark. Model performance increases with lower variance deciles.

Method	Map	MSE [m ²] ↓	P95 ↓
KBM	–	2.91 ± 3.18	10.20
State	–	1.60 ± 1.52	5.00
FPV	–	1.78 ± 1.61	5.38
Vanilla BEV	□	1.35 ± 1.53	4.86
Vanilla BEV	⊠	1.26 ± 1.24	3.97
Spatial BEV	□	1.57 ± 1.34	4.43
Spatial BEV	⊠	1.05 ± 0.83	2.82

Table 3: Offline prediction results on closed-loop experiment data

the experiment data in Table 3. We find while our CTE is closer to the baseline performance with the closed-loop control, our prediction error is still 16.6% better than the best baseline on average. While one intervention did happen for our method, this was diagnosed in Section C.3 as an MPPI-tracking issue instead of a dynamics modeling inaccuracy, but it is listed here for transparency. This answers **RQ3** and shows open-loop and closed-loop performance are not always correlated due to the entanglement of the controller.

While all models can track straight lines well at speed, the turns prove to be the most challenging to track. The KBM baseline consistently understeers in low and high curvature turns, resulting in at least 3.00 average interventions per trial across both courses. From Table 4, our approach matches closed-loop performance of the other learned baselines, even with sparse maps. While we demonstrate competitive average CTE on the high speed and geometry courses, our approach has the lowest worst-case CTE for the geometry course, aligning with the open-loop results of reducing worst-case performance. To help isolate the effect of the controller, we evaluate prediction error on

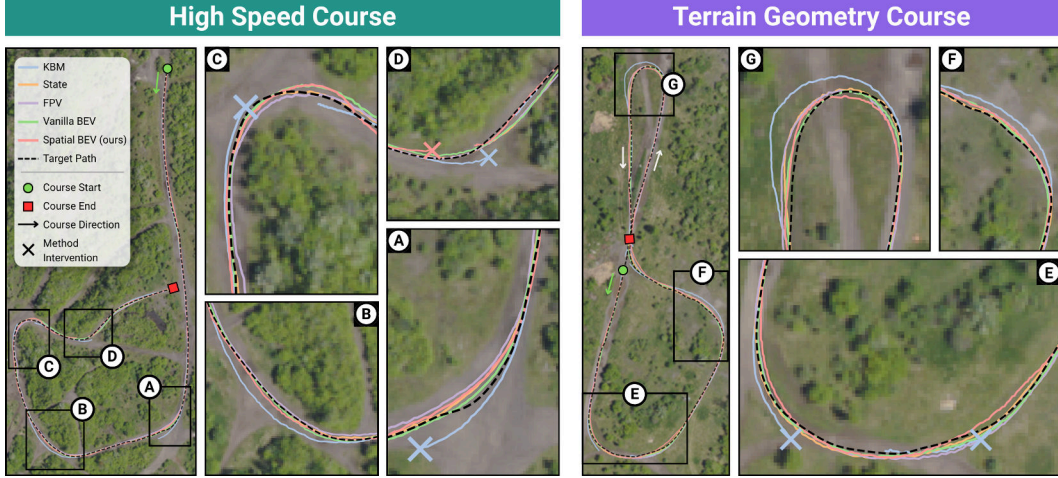


Figure 8: Experiments for hardware evaluation on two challenging courses. Significant understeer can be seen by the KBM baseline leading to numerous interventions (A,D,E). Additionally all learned methods, including ours, experience cases of understeer like in the tight turn in C or G.

Method	High Speed Course (1 trial)					Geometry Course (4 trials)				
	# Intv avg	vel. avg	vel. max	CTE avg	CTE max	# Intv avg	vel. avg	vel. max	CTE avg	CTE max
KBM	3.00	7.75	11.20	0.73	5.04	3.75	6.13	9.01	0.81	5.51
State	0.00	8.39	10.96	0.42	2.29	0.00	6.80	8.68	0.45	3.35
FPV	0.00	7.85	10.98	0.80	3.41	0.75	6.50	8.90	0.65	4.90
Vanilla BEV	0.00	8.44	10.94	0.47	3.26	0.00	6.94	8.60	0.43	3.67
Spatial BEV (ours)	1.00	8.02	10.41	0.51	2.62	0.00	6.58	8.66	0.60	2.74

Table 4: Hardware deployment results on two challenging courses.

6 Limitations

First, although our approach performs well on held-out datasets and environments, online adaptation could further improve robustness and generalization [12, 36, 20, 57, 58]. Second, training on real-world data avoids a sim-to-real gap but is expensive, sample-inefficient, and vulnerable to hardware failures; combining simulation and real data is a natural next step [59, 36]. Third, map querying and interpolation add computational cost. Vanilla BEV rolls out 825 particles in 12.9 ms versus 28.3 ms for ours, rising to 40 ms and 82 ms, respectively, with all components running. While sufficient for 10 Hz control, this leaves little room for more samples or a longer horizon. Finally, a better line tracking algorithm could be used for future testing, given the false intervention of our approach.

7 Conclusion

This paper presents structured perception representations for learned dynamics models for off-road terrain. We demonstrate how off-road terrain, represented through geometric or visual semantics, should not be left as unmodeled epistemic uncertainty but rather a core component to the dynamics model. Through extensive offline evaluation and online hardware results, we demonstrate the necessity of including perception representations in dynamics models and how additional spatial structure further improves dynamics modeling on high speed ($\geq 10\text{m/s}$) and geometrically challenging courses.

Acknowledgments

We thank Geethika Hemkumar and Saksham Sharma for their extensive support with field testing for the submission and rebuttal. We also thank Jason Gibson and Anoushka Alavilli for insightful discussions during the beginning of this work. Micah Nye is supported by the Rales Fellowship. This work was supported by DSO National Laboratories. This work used Bridges-2 at PSC through allocation cis220039p from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program which is supported by NSF grants #2138259, #2138286, #2138307, #2137603, and #213296.

References

- [1] T. Han, A. Liu, A. Li, A. Spitzer, G. Shi, and B. Boots. Model Predictive Control for Aggressive Driving Over Uneven Terrain. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi:[10.15607/RSS.2024.XX.022](https://doi.org/10.15607/RSS.2024.XX.022).
- [2] N. Seegmiller, F. Rogers-Marcovitz, G. Miller, and A. Kelly. Vehicle model identification by integrated prediction error minimization. *The International Journal of Robotics Research*, 32(8):912–931, 2013.
- [3] T. M. Howard and A. Kelly. Optimal rough terrain trajectory generation for wheeled mobile robots. *The International Journal of Robotics Research*, 26(2):141–166, 2007.
- [4] J. Gibson, B. Vlahov, D. Fan, P. Spieler, D. Pastor, A.-a. Agha-mohammadi, and E. 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, 2023. doi:[10.1109/ICRA48891.2023.10161330](https://doi.org/10.1109/ICRA48891.2023.10161330).
- [5] A. Kelly and N. Seegmiller. A Vector Algebra Formulation of Mobile Robot Velocity Kinematics. In K. Yoshida and S. Tadokoro, editors, *Field and Service Robotics: Results of the 8th International Conference*, pages 613–627. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014. ISBN 978-3-642-40686-7. doi:[10.1007/978-3-642-40686-7_41](https://doi.org/10.1007/978-3-642-40686-7_41). URL https://doi.org/10.1007/978-3-642-40686-7_41.
- [6] C. J. Ostafew, A. P. Schoellig, and T. D. Barfoot. Learning-based nonlinear model predictive control to improve vision-based mobile robot path-tracking in challenging outdoor environments. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4029–4036, 2014. doi:[10.1109/ICRA.2014.6907444](https://doi.org/10.1109/ICRA.2014.6907444).
- [7] K. Chua, R. Calandra, R. McAllister, and S. Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31, 2018.
- [8] P. E. Dupont. The effect of friction on the forward dynamics problem. *The International Journal of Robotics Research*, 12(2):164–179, 1993. doi:[10.1177/027836499301200205](https://doi.org/10.1177/027836499301200205). URL <https://doi.org/10.1177/027836499301200205>.
- [9] M. Bode. Learning the Forward Predictive Model for an Off-Road Skid-Steer Vehicle.
- [10] P. Roth, J. Frey, C. Cadena, and M. Hutter. Learned Perceptive Forward Dynamics Model for Safe and Platform-aware Robotic Navigation. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025. doi:[10.15607/RSS.2025.XXI.001](https://doi.org/10.15607/RSS.2025.XXI.001).
- [11] H. Lee, T. Kim, J. Mun, and W. Lee. Learning terrain-aware kinodynamic model for autonomous off-road rally driving with model predictive path integral control. *IEEE Robotics and Automation Letters*, 8(11):7663–7670, 2023. doi:[10.1109/LRA.2023.3318190](https://doi.org/10.1109/LRA.2023.3318190).

- [12] E. S. Lupu, F. Xie, J. A. Preiss, J. Alindogan, M. Anderson, and S.-J. Chung. Magicvfm-meta-learning adaptation for ground interaction control with visual foundation models. *IEEE Transactions on Robotics*, 41:180–199, 2025. doi:10.1109/TRO.2024.3475212.
- [13] F. Djeumou, J. Y. Goh, U. Topcu, and A. Balachandran. Autonomous drifting with 3 minutes of data via learned tire models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 968–974, 2023. doi:10.1109/ICRA48891.2023.10161370.
- [14] J. Gibson, A. Alavilli, E. Tevere, E. A. Theodorou, and P. Spieler. Dynamics modeling using visual terrain features for high-speed autonomous off-road driving. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9809–9815, 2025. doi:10.1109/ICRA55743.2025.11128031.
- [15] J. Levy, J. Gibson, B. Vlahov, E. Tevere, E. Theodorou, D. Fridovich-Keil, and P. Spieler. Meta-learning online dynamics model adaptation in off-road autonomous driving. In *RSS 2025 Workshop on Resilient Off-road Autonomous Robotics*, 2025. URL <https://openreview.net/forum?id=jnKYdHfFWA>.
- [16] P. Maheshwari, W. Wang, S. Triest, M. Sivaprakasam, S. Aich, J. G. Rogers, J. M. Gregory, and S. Scherer. PIAug – Physics Informed Augmentation for Learning Vehicle Dynamics for Off-Road Navigation, Nov. 2023. URL <http://arxiv.org/abs/2311.00815>. arXiv:2311.00815 [cs].
- [17] H. Xue, E. L. Zhu, J. M. Dolan, and F. Borrelli. Learning model predictive control with error dynamics regression for autonomous racing. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13250–13256, 2024. doi:10.1109/ICRA57147.2024.10611628.
- [18] F. Djeumou, T. J. Lew, N. DING, M. Thompson, M. Suminaka, M. Greiff, and J. 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=0gDbEtVrd>.
- [19] N. Mohajerin, M. Mozifian, and S. Waslander. Deep learning a quadrotor dynamic model for multi-step prediction. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2454–2459, 2018. doi:10.1109/ICRA.2018.8460840.
- [20] M. O’Connell, G. Shi, X. Shi, K. Azizzadenesheli, A. Anandkumar, Y. Yue, and S.-J. Chung. Neural-fly enables rapid learning for agile flight in strong winds. *Science Robotics*, 7(66): eabm6597, 2022. doi:10.1126/scirobotics.abm6597. URL <https://www.science.org/doi/abs/10.1126/scirobotics.abm6597>.
- [21] S. J. Wang, S. Triest, W. Wang, S. Scherer, and A. Johnson. Rough Terrain Navigation Using Divergence Constrained Model-Based Reinforcement Learning. In *Proceedings of the 5th Conference on Robot Learning*, pages 224–233. PMLR, Jan. 2022. URL <https://proceedings.mlr.press/v164/wang22c.html>. ISSN: 2640-3498.
- [22] G. Kahn, P. Abbeel, and S. Levine. Badgr: An autonomous self-supervised learning-based navigation system. *IEEE Robotics and Automation Letters*, 6(2):1312–1319, 2021. doi:10.1109/LRA.2021.3057023.
- [23] A. Nagabandi, G. Kahn, R. S. Fearing, and S. Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7559–7566, 2018. doi:10.1109/ICRA.2018.8463189.
- [24] S. Triest, M. Sivaprakasam, S. J. Wang, W. Wang, A. M. Johnson, and S. Scherer. Tartandrive: A large-scale dataset for learning off-road dynamics models. In *2022 International Conference*

- on *Robotics and Automation (ICRA)*, pages 2546–2552, 2022. doi:[10.1109/ICRA46639.2022.9811648](https://doi.org/10.1109/ICRA46639.2022.9811648).
- [25] A. Kumar, Z. Fu, D. Pathak, and J. Malik. RMA: Rapid Motor Adaptation for Legged Robots. In *Proceedings of Robotics: Science and Systems*, Virtual, July 2021. doi:[10.15607/RSS.2021.XVII.011](https://doi.org/10.15607/RSS.2021.XVII.011).
 - [26] W. Ward, S. Etter, T. Ingebrand, C. Ellis, A. Thorpe, and ufuk topcu. Online adaptation of terrain-aware dynamics for planning in unstructured environments. In *RSS 2025 Workshop on Resilient Off-road Autonomous Robotics*, 2025. URL <https://openreview.net/forum?id=T7KHLw3031>.
 - [27] T. Ingebrand, A. J. Thorpe, and U. Topcu. Zero-shot transfer of neural odes. *Advances in Neural Information Processing Systems*, 37:67604–67626, 2024.
 - [28] P. Atreya, H. Karnan, K. S. Sikand, X. Xiao, S. Rabiee, and J. Biswas. High-speed accurate robot control using learned forward kinodynamics and non-linear least squares optimization. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11789–11795, 2022. doi:[10.1109/IROS47612.2022.9981259](https://doi.org/10.1109/IROS47612.2022.9981259).
 - [29] I. Radosavovic, B. Zhang, B. Shi, J. Rajasegaran, S. Kamat, T. Darrell, K. Sreenath, and J. Malik. Humanoid locomotion as next token prediction. *Advances in neural information processing systems*, 37:79307–79324, 2024.
 - [30] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter. Learning agile and dynamic motor skills for legged robots. *Science Robotics*, 4(26): eaau5872, Jan. 2019. ISSN 2470-9476. doi:[10.1126/scirobotics.aau5872](https://doi.org/10.1126/scirobotics.aau5872). URL <https://www.science.org/doi/10.1126/scirobotics.aau5872>.
 - [31] J.-F. Tremblay, T. Manderson, A. Noca, G. Dudek, and D. Meger. Multimodal dynamics modeling for off-road autonomous vehicles. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1796–1802, 2021. doi:[10.1109/ICRA48506.2021.9561910](https://doi.org/10.1109/ICRA48506.2021.9561910).
 - [32] D. D. Fan, A.-a. Agha-mohammadi, and E. A. Theodorou. Deep learning tubes for tube mpc. *Robotics: Science and Systems XVI*, 2020.
 - [33] Y. Zhang, W. Wang, R. Bonatti, D. Maturana, and S. Scherer. Integrating kinematics and environment context into deep inverse reinforcement learning for predicting off-road vehicle trajectories. In A. Billard, A. Dragan, J. Peters, and J. Morimoto, editors, *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 894–905. PMLR, 29–31 Oct 2018. URL <https://proceedings.mlr.press/v87/zhang18a.html>.
 - [34] A. Koppel, J. Fink, G. Warnell, E. Stump, and A. Ribeiro. Online learning for characterizing unknown environments in ground robotic vehicle models. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 626–633, Oct. 2016. doi:[10.1109/IROS.2016.7759118](https://doi.org/10.1109/IROS.2016.7759118). URL <https://ieeexplore.ieee.org/document/7759118/>. ISSN: 2153-0866.
 - [35] W. D. Compton, N. Csomay-Shanklin, C. Johnson, and A. D. Ames. Dynamic tube mpc: Learning tube dynamics with massively parallel simulation for robust safety in practice. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2613–2619, 2025. doi:[10.1109/ICRA55743.2025.11127279](https://doi.org/10.1109/ICRA55743.2025.11127279).
 - [36] J. Levy, T. Westenbroek, and D. Fridovich-Keil. Learning to walk from three minutes of real-world data with semi-structured dynamics models. In P. Agrawal, O. Kroemer, and W. Burgard, editors, *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pages 2061–2079. PMLR, 06–09 Nov 2025. URL <https://proceedings.mlr.press/v270/levy25a.html>.

- [37] A. Doerr, C. Daniel, M. Schiegg, N.-T. Duy, S. Schaal, M. Toussaint, and T. Sebastian. Probabilistic recurrent state-space models. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1280–1289. PMLR, 10–15 Jul 2018. URL <https://proceedings.mlr.press/v80/doerr18a.html>.
- [38] A. Bar, G. Zhou, D. Tran, T. Darrell, and Y. LeCun. Navigation world models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15791–15801, June 2025.
- [39] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020. URL <https://openreview.net/forum?id=S1l0TC4tDS>.
- [40] J. Chen, J. Frey, R. Zhou, T. Miki, G. Martius, and M. Hutter. Identifying terrain physical parameters from vision - towards physical-parameter-aware locomotion and navigation. *IEEE Robotics and Automation Letters*, 9(11):9279–9286, 2024. doi:10.1109/LRA.2024.3455788.
- [41] Y. Kim, C. Kim, and J. Hwangbo. Learning Forward Dynamics Model and Informed Trajectory Sampler for Safe Quadruped Navigation. In *Proceedings of Robotics: Science and Systems*, New York City, NY, USA, June 2022. doi:10.15607/RSS.2022.XVIII.069.
- [42] A. Pokhrel, M. Nazeri, A. Datar, and X. Xiao. Cahsor: Competence-aware high-speed off-road ground navigation in $\mathbb{SE}(3)$. *IEEE Robotics and Automation Letters*, 9(11):9653–9660, 2024. doi:10.1109/LRA.2024.3457369.
- [43] D. E. Rumelhart and J. L. McClelland. *Learning Internal Representations by Error Propagation*, pages 318–362. 1987.
- [44] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS’17*, page 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.
- [45] Z. Li, W. Wang, H. Li, E. Xie, C. Sima, T. Lu, Y. Qiao, and J. Dai. Bevformer: Learning bird’s-eye-view representation from multi-camera images via spatiotemporal transformers. In *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IX*, page 1–18, Berlin, Heidelberg, 2022. Springer-Verlag. ISBN 978-3-031-20076-2. doi:10.1007/978-3-031-20077-9_1. URL https://doi.org/10.1007/978-3-031-20077-9_1.
- [46] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai. Deformable {detr}: Deformable transformers for end-to-end object detection. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=gZ9hCDWe6ke>.
- [47] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. *Advances in neural information processing systems*, 28, 2015.
- [48] F. Rogers-Marcovitz, N. Seegmiller, and A. 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.
- [49] P. J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987. ISSN 0377-0427. doi:[https://doi.org/10.1016/0377-0427\(87\)90125-7](https://doi.org/10.1016/0377-0427(87)90125-7). URL <https://www.sciencedirect.com/science/article/pii/0377042787901257>.

- [50] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8): 1735–1780, Nov. 1997. ISSN 0899-7667. doi:10.1162/neco.1997.9.8.1735. URL <https://doi.org/10.1162/neco.1997.9.8.1735>.
- [51] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2017. URL <https://arxiv.org/abs/1412.6980>.
- [52] S. Triest, A. Shaban, D. D. Fan, W. Wang, and S. Scherer. Travsuite: Traversability via self-supervised, uncertainty-aware irl and terrain estimation. *Robotics: Science and Systems 2026*, 2026.
- [53] J. Mai. System design, modelling, and control for an off-road autonomous ground vehicle. Master’s thesis, Carnegie Mellon University, Pittsburgh, PA, July 2020.
- [54] M. Sivaprakasam, P. Maheshwari, M. G. Castro, S. Triest, M. Nye, S. Willits, A. Saba, W. Wang, and S. Scherer. Tartandrive 2.0: More modalities and better infrastructure to further self-supervised learning research in off-road driving tasks. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12606–12606, 2024. doi:10.1109/ICRA57147.2024.10611265.
- [55] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou. Aggressive driving with model predictive path integral control. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1433–1440, 2016. doi:10.1109/ICRA.2016.7487277.
- [56] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou. Information-theoretic model predictive control: Theory and applications to autonomous driving. *IEEE Transactions on Robotics*, 34(6):1603–1622, 2018. doi:10.1109/TRO.2018.2865891.
- [57] M. Sivaprakasam, S. Triest, C. Ho, S. Aich, J. Lew, I. Adu, W. Wang, and S. Scherer. Salon: Self-supervised adaptive learning for off-road navigation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 16999–17006, 2025. doi:10.1109/ICRA55743.2025.11128268.
- [58] M. Liu, D. Pathak, and A. Agarwal. Locoformer: Generalist locomotion via long-context adaptation. In *9th Annual Conference on Robot Learning*, 2025.
- [59] W. Xiao, H. Xue, T. Tao, D. Kalaria, J. M. Dolan, and G. Shi. Anycar to anywhere: Learning universal dynamics model for agile and adaptive mobility. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8819–8825, 2025. doi:10.1109/ICRA55743.2025.11128396.
- [60] M. Ranzinger, G. Heinrich, J. Kautz, and P. Molchanov. Am-radio: Agglomerative vision foundation model reduce all domains into one. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12490–12500, 2024. doi:10.1109/CVPR52733.2024.01187.
- [61] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11975–11986, 2023.
- [62] S. Hajimiri, I. B. Ayed, and J. Dolz. Pay attention to your neighbours: Training-free open-vocabulary semantic segmentation. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 5061–5071. IEEE, 2025.
- [63] O. Alama, A. Bhattacharya, H. He, S. Kim, Y. Qiu, W. Wang, C. Ho, N. Keetha, and S. Scherer. Rayfronts: Open-set semantic ray frontiers for online scene understanding and exploration. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5930–5937, 2025. doi:10.1109/IROS60139.2025.11245813.

- [64] S. Zhao, H. Zhang, P. Wang, L. Nogueira, and S. Scherer. Super odometry: Imu-centric lidar-visual-inertial estimator for challenging environments. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8729–8736. IEEE, 2021.
- [65] C. Wellington, A. C. Courville, and A. Stentz. Interacting markov random fields for simultaneous terrain modeling and obstacle detection. In *Robotics: Science and Systems*, volume 6, pages 1–8, 2005.
- [66] D. Maturana. *Semantic Mapping for Autonomous Navigation and Exploration*. PhD thesis, Carnegie Mellon University, 2022.
- [67] M. Bhardwaj, B. Sundaralingam, A. Mousavian, N. D. Ratliff, D. Fox, F. Ramos, and B. Boots. Storm: An integrated framework for fast joint-space model-predictive control for reactive manipulation. In *Conference on Robot Learning*, pages 750–759. PMLR, 2022.
- [68] B. Vlahov, J. Gibson, D. D. Fan, P. Spieler, A.-a. Agha-mohammadi, and E. A. Theodorou. Low frequency sampling in model predictive path integral control. *IEEE Robotics and Automation Letters*, 9(5):4543–4550, 2024.
- [69] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 618–626, 2017. doi:[10.1109/ICCV.2017.74](https://doi.org/10.1109/ICCV.2017.74).

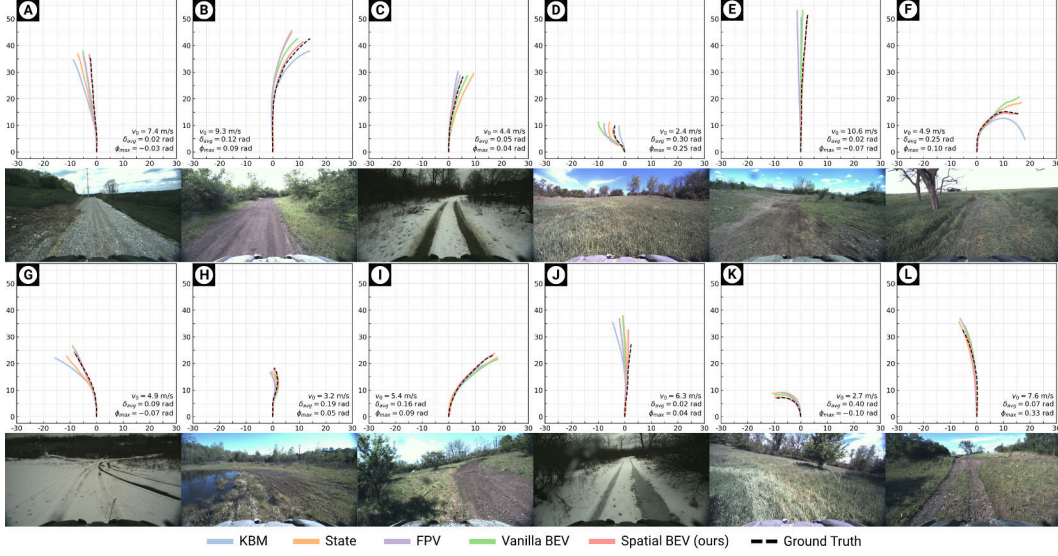


Figure 9: Offline qualitative results highlighting interesting prediction examples and environments.

A Additional Experiments

We further evaluate our approach qualitatively and quantitatively. Specifically, we investigate the qualitative offline results of our approach, compared to other methods and provide more insight on how our method’s proper use of structured perception representations improve its forward dynamics prediction (Sec. A.1). Next, we evaluate our approach against more methods, which do not use the RNN-based dynamics backbone and instead use Transformers (Sec. A.2). Finally, we provide insight to our perception representation via seed ablations (Sec. A.3).

A.1 Open-loop Qualitative Analysis

We qualitatively evaluate the success and failure cases of our approach by analyzing the prediction results in Figure 9. Across a diverse set of environments and motion profiles, including snow (C), ice (J), marshes (H), uphill grades (D,K,L), our method consistently produces accurate trajectory predictions at low, medium, and high speeds, spanning straight segments (A,E), slight and large turns (G,I), sharp turns (F,K), and slalom maneuvers (D,H). Notably in (G), all perception-based methods perform comparably well on a slight snowy turn, highlighting conditions where appearance-based cues alone provide enough context.

In challenging uphill scenarios (L), all methods overshoot due to poor terrain understanding ahead, while our approach remains closest to the ground truth with only a slight undershoot, reflecting our model’s improved terrain awareness. The most difficult case arises in (J), where ice on the right breaks under the robot’s weight, causing the vehicle to slow and its tires to get lodged in a newly formed deep rut. All methods overshoot, predicting trajectories to the left of the actual path, since the rut constrains the robot’s motion. No method is able to predict this phenomenon, regardless of representation.

A.2 Extended Open-loop Results

While we restricted our evaluation to an LSTM-based ablation for brevity, we provide further offline quantitative results on popular alternatives in sequential prediction, such as Transformer-based architectures [44]. We choose to evaluate against the following baselines:

Method	Map	Snow		Slopes		Speed		Full	
		MSE ↓	P95 ↓	MSE ↓	P95 ↓	MSE ↓	P95 ↓	MSE ↓	P95 ↓
KBM	–	2.68	8.92	2.31	8.00	2.83	10.24	2.10	7.20
State	–	2.54	7.62	2.20	7.48	1.91	7.39	1.99	6.71
FPV	–	2.34	6.93	<u>1.69</u>	<u>5.82</u>	2.12	7.82	1.77	6.00
Vanilla BEV	□	2.49	8.12	<u>1.98</u>	<u>7.20</u>	1.59	6.33	1.73	6.66
Vanilla BEV	⊠	<u>2.10</u>	<u>6.76</u>	1.97	7.70	<u>1.41</u>	<u>5.17</u>	<u>1.46</u>	<u>5.63</u>
State TF	–	2.40	7.53	<u>1.69</u>	6.34	1.83	7.24	1.75	6.21
BEV TF	□	3.61	10.29	2.06	7.83	3.46	12.54	2.37	8.56
AnyCar [59]	–	2.42	7.92	1.92	7.89	1.66	6.40	1.70	6.51
Spatial BEV (ours)	□	2.45	8.06	1.78	6.87	1.50	5.42	1.70	6.26
Spatial BEV (ours)	⊠	1.74	5.55	1.51	5.26	1.13	3.58	1.21	4.41

Table 5: Extended offline position prediction results, including Transformer-based evaluations, evaluated across 3 datasets, alongside the full test set. Standard deviation removed for width-fitting.

1. State-only Transformer (State TF), which tokenizes the history of state-action pairs alongside future actions. We architect it as a Transformer encoder, with a causal attention mask on the state-action pairs.
2. BEV Transformer (BEV TF), which additionally tokenizes a BEV map for perceptual conditioning. We tokenize the BEV map by encoding each 5×5 cell patch into one token. We architect this as a Transformer decoder, with a causal attention mask on the state-action pairs and full-attention between BEV and state-action pairs, so each state or action token is able to attend to the most relevant BEV patches.
3. AnyCar [59]: a state-of-the-art, state-based Transformer for dynamics prediction. We train from scratch on the same data all LSTM’s were trained on in Section 5. Notably, we train it to predict 5s sequences at 10Hz, whereas, Xiao et al. [59] predict 1s in advance at 50Hz. Likewise the history differs in this manner.

Table 5 extends the main results to include these Transformer baselines. The Transformer baselines achieve low prediction error on the Snow and Slopes datasets but degrade on the Speed dataset. The BEV Transformer performs worst across nearly all methods and datasets, suggesting that explicit sampling grids are a more effective spatial attention mechanism. Granted, its architecture is relatively simple and could be improved in future work.

The advantage of our structured representation is most apparent on the Speed dataset, where our method achieves the best average and worst-case performance on trajectories with large displacements from the starting position. This supports our conjecture that explicit spatial structure helps the dynamics model attend to the most relevant environmental features.

A.3 Seed Ablations

While offline results and ablations show success of the architecture, we perform a seed ablation. Over four random seeds, we see a prediction error standard deviation of 0.07 m^2 .

B Dataset Details

We collect our dataset using human expert demonstrations or prior autonomous navigation on an off-road all-terrain vehicle in the real world across two test sites in all four seasons. The entire dataset encompasses 16.3 hr of driving data, prior to post-process filtering and sub-dataset partitioning. This dataset has top speeds of 13 m/s and max pitch of 19.1° (32.2% grade) with further dataset statistics detailed in Table 6. The dataset includes terrain ground cover such as grass, loose gravel, snow, ice, mud, and standing water, as well as geometry like steep slopes, banks, ruts, and undulating terrain. These features provide a diversity in geometric complexity, friction properties, and terrain deformability.

	Mean	Std	Min	5%	Median	95%	Max
Forward vel. (m/s)	4.66	1.87	-0.12	1.92	4.40	8.27	13.01
Lateral vel. (m/s)	0.02	0.27	-1.61	-0.37	0.01	0.44	3.89
Slip (deg)	0.27	4.41	-38.43	-6.10	0.13	7.24	51.23
Yaw Rate (rad/s)	-0.00	0.14	-0.89	-0.24	-0.00	0.24	0.84
Pitch (deg)	-0.07	3.84	-18.92	-7.34	0.12	6.71	19.10
Roll (deg)	0.22	3.31	-18.85	-5.28	0.37	5.22	20.07

Table 6: Statistics of entire dataset, encompassing test and train datasets

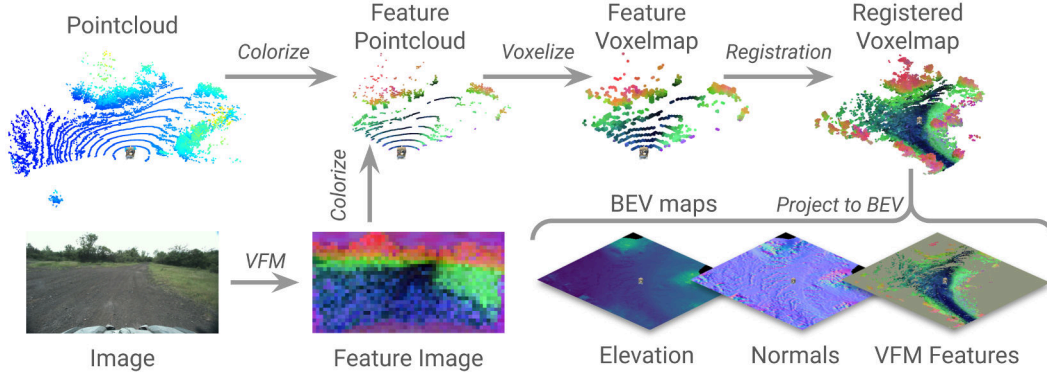


Figure 10: Perception and mapping flow diagram

Using this large dataset, we split it into our train and test datasets as mentioned in Section 5. Our test dataset is broken up into Snow, Slopes, and Speed sub-datasets for evaluation. These sub-datasets are defined in the following ways:

1. Snow: Our dataset was manually filtered using run log metadata to find the runs with snow. Having snow was defined as having snow completely cover the ground.
2. Slopes: We filtered our dataset to ensure, for a given trajectory τ , the maximum pitch experienced is at least 0.1 radians (5.8°).
3. Speed: We filtered out data to ensure, for a given trajectory τ , the average speed is at least 6.0 m/s.

Post-process filtering filters negative speed to remove reverse motions, samples with large odometry covariance, and samples with braking, since brake is not in the robot’s action space (Sec. C.2).

C System Implementation & Analysis

C.1 Software

C.1.1 Perception

In order to obtain inputs for our learned model, we leverage a GPU-accelerated local mapping package that produces geometric-semantic BEV maps from camera and LiDAR data. This process is shown in Figure 10 and accomplished via the following procedure:

1. Salient visual features are extracted from the front-facing camera images using c-RadioV3 [60] with a SIGLIP2 adaptor [61] and NACLIP attention [62] and are then passed through a domain-specific principal component analysis (in a similar fashion to RayFronts [63]). This results in a $53 \times 28 \times 32$ “feature image”.
2. This feature image is then lifted into 3D via pointcloud colorization, in which the LiDAR point cloud is projected into the feature image frame. Points that lie within the image

Sampler	σ	K
Gaussian Walk 1	[0.001, 0.001]	128
Gaussian Walk 2	[0.0005, 0.0005]	128
Gaussian Walk 3	[0.000025, 0.00025]	128
Action Library	–	441

Table 7: Action Sampling Parameters

Parameter	Value
$a_{\max} (m/s^2)$	5.0
$v_{\max} (m/s)$	8.0 or 10.0
K_{track}	10.0
K_{progress}	1.0

Table 8: MPPI cost function parameters

frame inherit the feature from the pixel that they fall into (Eq 7), resulting in a “feature point cloud”.

3. The feature point cloud is then transformed into a fixed frame (via an on-board LiDAR-inertial odometry algorithm [64]), voxelized, and aggregated into a scrolling voxel grid centered at the robot’s current position.
4. Finally, we use the voxel grid to estimate the underlying terrain surface of the robot’s local environment using a Markov random field [65, 66]. The geometric features in the input BEV map such as step height and slope, are computed using the voxel grid geometry and the terrain surface. VFM features are computed as the average feature of all voxels within a threshold (roughly the ATV height = 1.7 m) of the terrain surface.

$$[u', v', w]^T = \mathbf{K}[\mathbf{R}|\mathbf{t}][x, y, z, 1]^T, \quad u = \frac{u'}{w}, \quad v = \frac{v'}{w} \quad (7)$$

This mapping system is run using the on-board GPU (Sec. C.2) at 5 Hz to produce BEV maps for training and inference. We apply minor post-processing on the BEV maps before inference:

1. To avoid the need for the network to learn BEV orientation and feature rasterizing, we transform the BEV map into an ego-aligned frame such that forward is always facing right. We do this by creating a grid of ego-frame coordinates, centered at (0, 0) in ego space. Next, we rotate this grid into odom frame. We then sample the odom BEV map at the ego coordinates in odom frame.
2. We modify the features to limit local terrain elevation from [-10, 10 m] and we rotate geometric data, such as surface normals, to be aligned in ego frame.

C.1.2 Control

Our path tracking results used MPPI [55, 56] as a model-predictive controller. We follow the standard formulation of MPPI, which requires us to specify 1) a dynamics model, 2) an action sampler, and 3) a cost function.

Our learned models serve as the dynamics model that MPPI optimizes over. Following recent advances in action-sampling techniques for MPPI [67, 68], we generate candidate action sequences via additive noise generated from a set of Gaussian walks with varying σ . We also add a fixed set of uniform action sequences to enable the robot to more effectively space potential local minima in the cost function. We grid sample these sequences uniformly in [throttle, steer] space, resulting in 441 uniform sequences that are evaluated at every MPPI iteration.

Our cost function for MPPI uses a combination of path-tracking costs and physics-derived safety limits (Eq. 8). In a nutshell, this cost function encourages the robot to make forward progress along the reference path τ_{ref} while adhering to its shape (Eqs. 11, 12), subject to speed and lateral acceleration constraints (Eqs. 9, 10). Path progress can be interpreted as the amount of reference path (in m) traveled from the candidate trajectory’s initial state x_t to terminal state x_{t+T} . Path track error can be interpreted as the average distance of the states in the trajectory to the reference path. The tradeoff between progress and accuracy is governed by the weights K_{track} and K_{progress} . K_{track} is set significantly higher than K_{progress} to disincentivize cutting corners; however, as witnessed in Section 5.3,

more tuning could be done to avoid MPPI-related failures and expose only dynamics modeling failures. Paths are represented as a dense sequence of SE(2) poses and all path computation occurs in 2D (Eqs. 11, 12). Hyperparameters are presented in Table 8.

$$J(\tau) = J_{\text{speed}}(\tau, v_{\max}) + J_{a_y}(\tau, a_{\max}) + K_{\text{track}} J_{\text{track}}(\tau, \tau_{\text{ref}}) + K_{\text{progress}} J_{\text{progress}}(\tau, \tau_{\text{ref}}) \quad (8)$$

$$J_{\text{speed}}(\tau, v_{\max}) = \sum_{x \in \tau} \left[\begin{cases} \infty & \text{if } x_v > v_{\max} \\ 0 & \text{otherwise} \end{cases} \right] \quad (9)$$

$$J_{a_y} = \sum_{x \in \tau} \left[\begin{cases} \infty & \text{if } \frac{x_v^2 \tan(\delta)}{L} > a_{\max} \\ 0 & \text{otherwise} \end{cases} \right] \quad (10)$$

$$J_{\text{progress}}(\tau, \tau_{\text{ref}}) = \sum_{i=i_{\text{start}}}^{i_{\text{end}}-1} \|\tau_{\text{ref}}^{(i+1)} - \tau_{\text{ref}}^{(i)}\|_2 \quad (11)$$

$$i_{\text{start}} = \underset{i}{\operatorname{argmin}} \left[\|\tau_{\text{ref}}^{(i)} - x_t\|_2 \right]$$

$$i_{\text{end}} = \underset{i}{\operatorname{argmin}} \left[\|\tau_{\text{ref}}^{(i)} - x_{t+T}\|_2 \right]$$

$$J_{\text{track}}(\tau, \tau_{\text{ref}}) = \frac{1}{T} \sum_{x \in \tau} \left[\min_{x_{\text{ref}} \in \tau_{\text{ref}}} [\|x_{\text{ref}} - x\|_2] \right] \quad (12)$$

C.2 Hardware

Actuators The vehicle’s action space covers steering and throttle, with no brake actuators available. It is important to note while autonomous action is relatively smooth, teleoperation actions are large discrete, because operators are controlling a gamepad controller with low-friction joysticks.

Additionally, due to the resistance of off-road terrain and limitations of the steering DC motor, there is steering actuation delay, with ~ 5 s of delay from commanding max steer to reaching max steer. The dynamics model implicitly learns this steering delay. There are also standard throttle dynamics, as for any vehicle.

Sensors For this work, the robot is equipped with a Velodyne VLP-32 LiDAR and a Multisense S21 RGB stereo camera. It is also has a MicroStrain 3DMGQ7-GNSS for GPS localization with RTK corrections. The LiDAR and camera run at 10Hz, whereas the RTK runs at 25Hz.

Basestation We deployed our perception and control algorithms on a laptop, equipped with an Intel Core Ultra 9 275HX CPU with 24 cores and 32GB of RAM and a laptop-grade Nvidia GeForce RTX 5080 with 16GB of VRAM.

C.3 MPPI Failure Analysis

We detail why our model intervention from Figure 8 is not a modeling failure but instead a control failure due to the cost function tuning. Consider the scene in Figure 11 where each frame progresses after 400 ms and the red MPPI solution is a new solution computed online. (A) shows the robot is following the target path at speed while closely tracking the reference line. Next, 400 ms later in (B), while still closely following the line, MPPI determines its optimal solution undercuts the reference line, which begins the cause for our intervention. In (C-E), MPPI follows through with its undercut of the reference line, and our accurate dynamics yield little deviation. After 2000 ms of following these undercutting solutions in (F), the human must intervene as the robot nears too dense of vegetation.

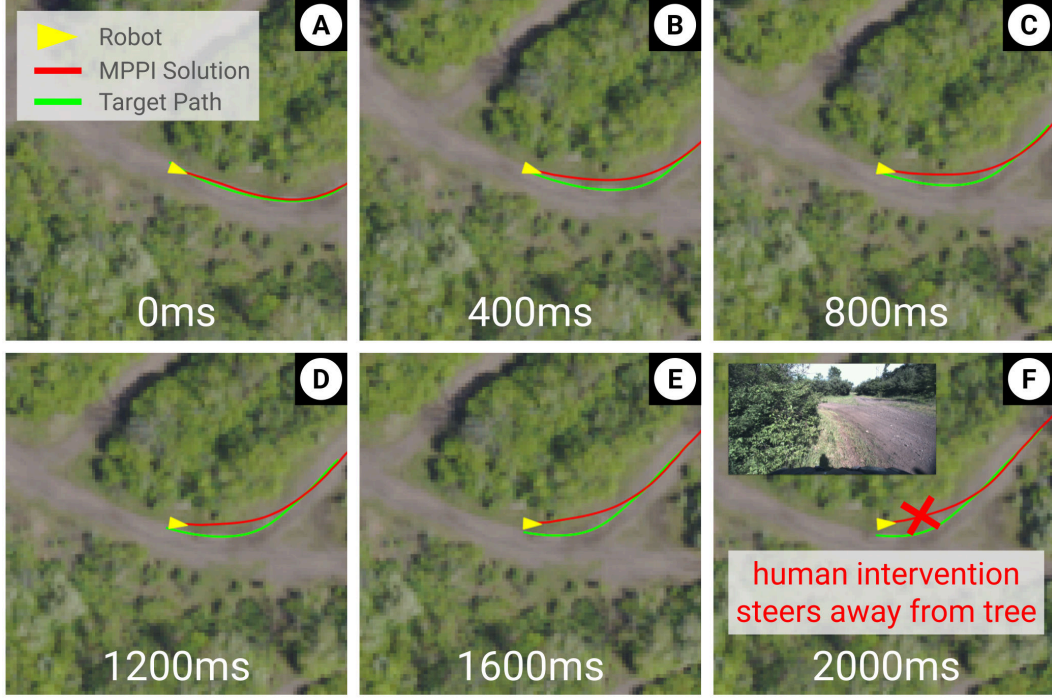


Figure 11: A robot following a target path with its optimal MPPI solution visualized. It is closely following the line in (A), highlighting accurate dynamics and control, but from (B) onwards, MPPI determines the optimal path undercuts the target path, nearly resulting in collision with dense vegetation.

This particular intervention is most likely a cause of the tracking cost term weight, K_{track} , being too low and also due to sampling sparsity, where there was no action sequence that yielded a sample close enough to the reference path. This case study highlights how not all failures are dynamics modeling failures and how an MPC objective function can be very sensitive to its tunable parameters.

D Learned Dynamics Model Implementation

D.1 Network Architecture

The base LSTM architecture is the same for all LSTM methods. The base architecture, composed of the context and predictor LSTM both have a hidden size d_h of 128. All methods use the same MLP state encoder and decoder, encoding to and from d_h . All methods use the same MLP input encoders. For the predictor LSTM, the input encoder is purely an action encoder, encoding to d_z^p of 16; however, for the context LSTM, the input encoder encodes state-action pairs, encoding to d_z^c of 144. The context LSTM receives the encoded last state of the history buffer for hidden state initialization. Hyperparameters are in Table 9.

Perception encoding The vanilla perception methods encode the perception once into a hidden state of dimension d_h to initialize the context LSTM. This latent gets concatenated with the state latent to form the initialization. The FPV LSTM uses a CNN encoder with $3\times$ downsampling blocks. The BEV LSTM uses a CNN encoder with $4\times$ downsampling blocks. These blocks apply the convolution, apply batch norm layers, use ReLU activation, and perform global max pooling.

Our spatial method queries the BEV map at the current predicted state. It queries 40 points, representing tire track points. This tensor of features $40 \times F$ gets encoded with an MLP into the bev context latent size d_b of 32. This gets concatenated with the latent input to condition the next state

Hyperparameter	Variable	Value
LSTM hidden size	d_h	128
Predictor input encoding dim	d_z^p	16
Context input encoding dim	d_z^c	144
BEV context latent size	d_b	32
Spatial query points (tire track)	$\mathcal{F}$	40
FPN CNN Downsample block	–	$3\times$
Vanilla BEV CNN Downsample block	–	$4\times$
BEV Features	F	36
Visual BEV Features	F_{VFM}	32
Terrain BEV Features	$F_{Terrain}$	4

Table 9: Hyperparameters for the LSTM architecture and perception encoders.

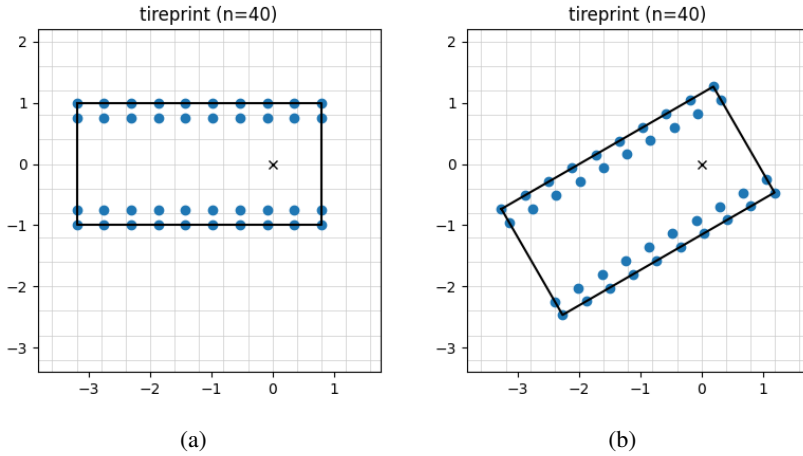


Figure 12: Tire track footprint visualization over a BEV map of the $100 \times 100 \times 0.4$ m resolution aligned with the map (a) and rotated (b) to show how many new cells are queried. Vehicle outline is in black.

prediction, and it gets concatenated with the hidden input and then added to apply the correction step. Hyperparameters are in Table 9.

D.2 BEV Footprint

Our footprint is two “tracks,” separated 1 m laterally from vehicle frame and uniformly spanning 4×0.42 m with 10×2 points. With a 0.4 m BEV resolution, most points query unique cells, and we bi-linearly interpolate to prevent duplicate queries. Figure 12 shows our footprint over a BEV map aligned with the map and rotated. We do not condition on velocity or steer—our footprint aligns with our predicted position and orientation—nor suspension or roll, which follow from the terrain estimate. Slip and weight transfer are present in our training data and are represented implicitly in the recurrent latent, not as explicit state, since MPPI does not require them as planning states.

D.2.1 Ablation Footprints

For our ablation in Section 5.1, we compare against these three other footprints, including our tracks footprint, shown in Figure 13. 4-point is the simplest footprint and has been used by Gibson et al. [14]. Full Body is a naive footprint, without considering the tire location. And Lookahead is a footprint used by Lee et al. [11] as a means to remove the prediction component of the perception querying by encompassing a larger area of where the robot can feasibly go.

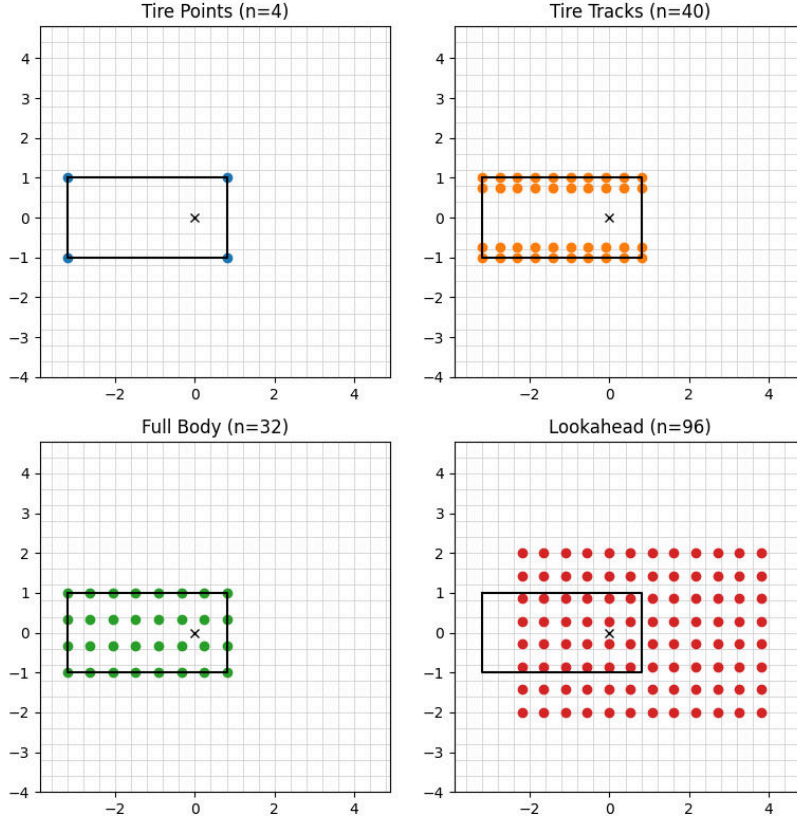


Figure 13: Footprints used for footprint ablation. Vehicle outline is in black.

D.3 Activation Maps

We create these activation maps using GradCAM [69], where we compute the activation needed to yield a single predicted scalar. In our case, for a given sample, we compute our prediction through the forward dynamics RNN. Then, we take the first predicted state’s x position, and back-propagate this result back to the last layer of our Vanilla BEV CNN encoder. At this layer, we compute the global importance of each channel by global-average-pooling its gradients, yielding a per-channel weight. We then compute a weighted vote by taking a per-cell weighted sum of the activations across channels, using these weights. To handle polarity, we apply a ReLU (or, alternately, an absolute value) to the weighted activation, retaining only features that positively (or, in the latter case, either positively or negatively) influence the predicted output. Finally, we upsample this coarse activation map to the input resolution to produce the final attention map seen in Figure 5. We choose to back-propagate from the x position because we saw, qualitatively, the activations were similar between state fields and along the rollout. More examples shown in Figure 14.

D.4 Training

Training Data We train using state estimates from SuperOdometry [64], a LiDAR-Inertial Odometry solution, but our models use RTK during real-world deployment. RTK was not available throughout the collection of this dataset, hence the mismatch. This does highlight performance robustness to state estimation solution.

Training Machines We trained our state-based methods on a single V100-32 GPU. Our BEV-based methods were trained on a single H100 GPU. While they could be trained on V100’s, we used H100’s for their superior VRAM, allowing us to have larger batches and more efficient training. It

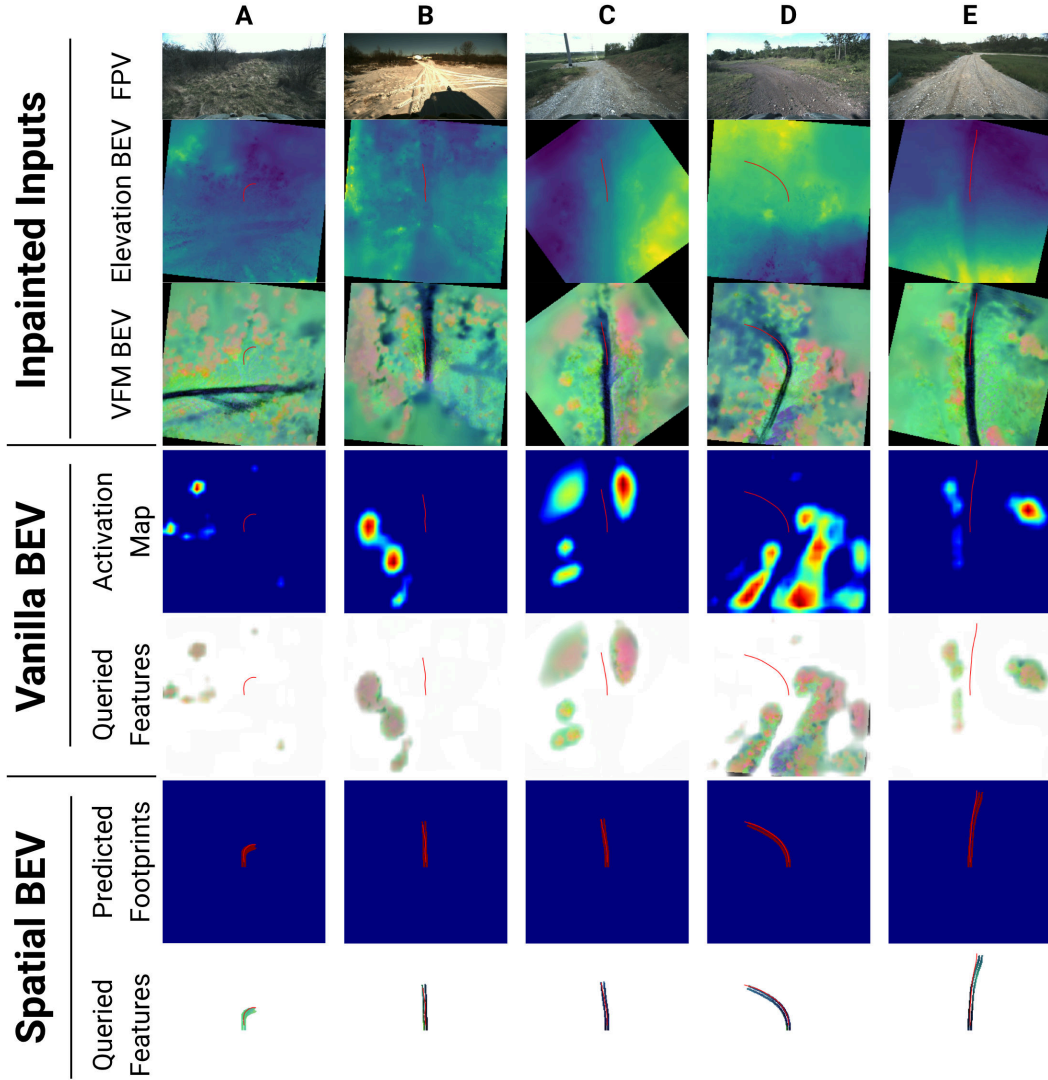


Figure 14: Regions of interest for Vanilla BEV and our Spatial BEV approach on five different samples. Vanilla BEV attends to non-sensical regions for predicting motion, whereas our approach queries exactly where it predicts the robot to be, yielding a more informed environment context. The red line is the ground truth trajectory on all BEV plots. (A) through (E) show various environments and motions.

takes ~ 30 m to train the state-based methods and ~ 6 hr to train the BEV-based RNNs. The BEV-based transformer took about ~ 24 hr to train.