Terrain-Aware Dynamics Models for High-Speed Off-Road Navigation

Micah Nye

CMU-RI-TR-26-101

August, 2026

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Wenshan Wang, *co-chair*
Sebastian Scherer, *co-chair*
Aaron M. Johnson
Anoushka Alavilli

Submitted in partial fulfillment of the requirements for the degree of Master's in Robotics.

Copyright © 2026 Micah Nye. All rights reserved.

To Nidhi

Abstract

High-speed autonomy in the real world requires accurate control, which often relies on dynamics models that capture the complex interaction between a robot and its environment. In off-road regimes, this terrain interaction dominates the robot's dynamics, driven by challenging characteristics such as diverse surface properties, complex geometries, environmental diversity, and high-speed instability. Physics-based dynamics models lack the expressivity for these terradynamics, and existing learned models do not fully exploit available perception. This thesis investigates how terrain-aware perception can improve learned dynamics modeling and control at high speed and how such models can be rigorously evaluated before deployment.

First, we demonstrate how perception representations can make dynamics models terrain-aware, capturing the geometric and semantic detail that drives terrain interaction. We formulate a representation that queries terrain features along the robot's predicted motion, yielding higher prediction accuracy. Second, we introduce a rigorous evaluation method to expose failure modes before real-world deployment. We collect a challenging, multi-season dataset at speeds up to 13 m/s and mine the most difficult evaluation samples using our benchmarking method. We observe that, while models appear accurate on average, our benchmark surfaces the long-tail failure cases where prior models fail catastrophically.

Together with our verified, terrain-aware model, we decrease the worst-case prediction error by 23.8%, compared to physics-based and learned baselines. We further evaluate on a full-scale ATV platform across high-speed ($\geq 10\,\mathrm{m/s}$) and geometrically challenging courses with a 34.9% reduction in maximum cross-track error. These results demonstrate the importance of embedding environment context for locomotion-related tasks.

Acknowledgments

First and foremost, thank you to my advisors, Wenshan Wang and Sebastian (Basti) Scherer. You gave me complete freedom to chase my ideas and never said no to a wild one. Wenshan, you taught me to find the key technical nuggets in my work and to communicate them in brief or detail. Your kindness and support boosted my confidence and helped me re-strategize after setbacks. Basti, you welcomed me as an undergrad who didn't know the first thing about research and helped me understand what robotics research means. You constantly remind me to think big picture and how to best communicate that to my audience. I sometimes forget your hardware and software mastery until we're in the field for a sponsor demo and you snap into action when things go awry. To the rest of my committee—Aaron Johnson and Anoushka Alavilli—thank you. Aaron, thank you for the support over the last few months. I have enjoyed our discussions and appreciate your encouragement to use the future works section as a springboard for my PhD directions. Anoushka, thank you for being a great collaborator—whether on your work or mine—and friend. It has been a delight chatting about dynamics & control together and in our reading group.

Thank you to more key players along my journey—Samuel Triest, Rachel Burcin, Andrew Saba, Brady Moon—who showed me how to do research, design systems, communicate, and support my community. Sam, thank you for your mentorship, friendship, and commiseration of the last two years. From the highs of high-speed autonomy to the lows of sub-zero tests when nothing worked, I've enjoyed every step alongside you. You cemented my writing with formalisms and helped me refine my PhD applications. And you have made me a better researcher and engineer, encouraging me to think about modularity and simplicity with the shortest path to a result. Rachel, thank you for RISS and for your dedication to me as an individual. Your passion for supporting young students knows no bounds and has changed hundreds of lives. You showed me what community engagement means, and you backed me as a leader and a teacher for the roboticists who come next. Andrew, I would not be writing this if it were not for your relentless dedication to my future. The moment you showed me the RI high bay, my eyes were awaken; I never would have dreamed a few years later, you'd have mentored me to lead the team at LVMS. And, Brady, whenever I make *any* media, I think "What would Brady say?" and am likewise always giddy to show you my latest creations. Your excellence in communication is something I will always strive towards.

Thank you to the TartanDriver team—Sam Triest, Matthew Sivaprakasam, Yifei Liu, Geethika Hemkumar, Saksham Sharma, Zhipeng Zhao—and all the off-road folks—Mateo Guaman Castro, Chase Noren, Isaiah Adu, Alina Wang, Flo Heisen, Anas Atmani, Parv Maheshwari, Shubhra Aich, Anton Yanovich, Shibo Zhao, Anthony Brogni,

Anoushka Alavilli, Neelay Joglekar[†]. We bonded through the hardships of hardware, and I'm grateful for the incredible feats, the unhinged photos, and the iron particles we have collectively ingested. Matthew, you are as calm, honest, and smart as they get—honest enough that we both know you're not fit for the military, and smart enough to drive a pick-up and place nametags. Thanks for keeping me level through my weekly "I'm dying" spiral, and for being a great co-lead on all our projects and proposals. Yifei, you knew the grind like you know how to win 1st at go-karting. Thank you for the checking in on me during my grind and for all our tea sessions. Geethika, our field tests enabled this work and earned you "wettest member of the team" and "most navigationally challenged" awards in the process. Thank you, and sorry. Saksham, thank you for making me feel like I was born in 1955, for keeping me current on all things tech, and for handling those autonomous high-speed corners that briefly separated you (and me) from our souls.

To the rest of my lab (Neelay, Jack, Cherie, Nick, Jay, James, Lorenzo, Yaoyu, Kabir, Winston, Viktor, and so many more): thank you for lending an ear, for conversations both deeply intellectual and deeply not, and for making the space one I wanted to be in. To Steve and Bob, thank you for your support with operations and facilities. To all my friends in RI, RISS, and CMU—Anoushka, Krishna, Billy, Juan, Nate L., Tushaar, Hayden, Nate O., Hilton, Christa, Maxwell, Abraham, Lameck, Eric, Erick, Natural, so many more—thank you for the chats, the dinners, and everything in between. And to my robots Vicky, Betty, Roley, Lester, and Spotless, thanks for all the pain and joy you've caused me.

My Rales cohort and its leadership are my second academic family. Amy Burkert and Moi Padilla, thank you for the tenacity it took to build this program; you have already handed many of us opportunities we would not have dreamed of. Amy, you made time for me despite everything else CMU asks of you. Our conversations helped me decide on my commitment to PhD. Moi, you radiate positivity no matter the circumstance; there is no better pilot to fly this plane while building it. I will always remember your words of encouragement, "Everybody poops." And to Jamie, Britnai, Erin, and Markita—thank you for all your advice and support.

From my humble beginnings as a roboticist, thank you to MPRW and the people at Pitt who made it happen—Andrew, Ro, Josh, Matthew, Nayana, Prem—along with Eidan, Joey, Alex, Will, Thomas, Aragya, Robbie, and many more. Sicko mode will live on forever, second to the skillet pancakes in the pit lane. Thank you, Kirby, bill, Nick, Matthew, and Jenna for our time at Estat. Kirby, our coffee chats left my stomach full and career bright—thank you for all your grad school support. And to my friends outside RI—Will, Nick C., Alex, Jenny, Jake, Jacob, Ian, Nick G., Jayden,

[†]Honorary.

Liam—thank you for only partially questioning my madness in pursuing a PhD.

My biggest thanks goes to my family. Mom and Dad, you supported me unconditionally as I chased the unknown of higher education. As if undergraduate was not enough, I dove deeper with a Master's and now toward a PhD. Thank you for instilling in me curiosity and kindness needed for this venture. Bethany and Joelle, you have been with me through thick and thin. You relentlessly gaslight me into feeling smart and on top of things, even though I most certainly do not; however, my domination in our game nights do remind me of what I am capable of ;). Chris, I hope my research reminds you of Aperture Science and Vault-Tec (Volunteer for testing today)—just kidding. Mabel, maybe one day you'll read your uncle's thesis. If so, hi! Thanks to the rest of my family: Anjali, Grandma, Megan, Archana, Mahendra, and many more.

And Nidhi, this achievement is yours as much as it is mine. You are the one who likes rollercoasters but somehow I took you for a ride worse than my ATV demonstration data. From late night consoling to the ultimate hype engine, you have always been in my corner. Thank you for your patience and positive energy, even when they were at their limit. It is the movie nights, the blanket forts, the Lego creations, and our Pittsburgh adventures that have kept me sane. I treasure the memories we collect and our collections of treasure, expanding one movie, book, record, and game at a time. I cannot wait to move our trove into our new home and continue our life together.

Funding

The work was supported by the CMU Rales Fellows Program. The author was a member of the inaugural 2024 cohort. 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.

Contents

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Challenges . . . 2
- 1.3 Research Questions . . . 4
- 1.4 Thesis Statement and Contributions . . . 5

2 Background **7**
- 2.1 Preliminaries . . . 7
- 2.2 Related Work . . . 7
  - 2.2.1 Dynamics Models for Mobile Robots . . . 7
  - 2.2.2 Learning Physics-informed Dynamics Models . . . 8
  - 2.2.3 Proprioceptive Learned Dynamics Models . . . 8
  - 2.2.4 Exteroceptive Learned Dynamics Models . . . 9
- 2.3 Model-Predictive Control . . . 10
  - 2.3.1 Sampling-based Model-Predictive Control . . . 11
- 2.4 Dynamics Modeling . . . 13
  - 2.4.1 Physics Modeling . . . 14

3 Terrain-Aware Dynamics Models **17**
- 3.1 Introduction . . . 17
- 3.2 Perception Pipeline . . . 18
  - 3.2.1 Feature Mapping . . . 19
  - 3.2.2 Colorizing . . . 19
  - 3.2.3 Voxelizing and Registration . . . 20
  - 3.2.4 BEV Projection . . . 20
  - 3.2.5 BEV Preprocessing . . . 21
- 3.3 Proprioceptive Modeling . . . 21
- 3.4 Structured Perception Representations for Environment Context . . . 23
  - 3.4.1 Recurrent Spatial BEV Cells . . . 24
  - 3.4.2 Correction through Observation . . . 26
  - 3.4.3 Intuition on Structured Representations . . . 27
- 3.5 Improving Terrain-Aware Dynamics Models with Inpainted Features . 30

4 Rigorous Evaluation **33**

4.1 Introduction . . . 33
4.2 Challenging Data Collection . . . 34
4.2.1 Platform . . . 35
4.2.2 Dataset . . . 37
4.3 Benchmark Design . . . 37
4.3.1 Data Mining Intuition . . . 39
4.3.2 Clustering Approach . . . 39
4.3.3 Benchmark Metric . . . 40

5 Experiments **43**
5.1 Shared Setup . . . 44
5.1.1 Baselines . . . 44
5.1.2 Training Dataset . . . 44
5.1.3 Model Implementation . . . 45
5.2 Offline Experiments . . . 45
5.2.1 Test Datasets . . . 45
5.2.2 Quantitative Evaluation . . . 46
5.2.3 Qualitative Evaluation . . . 50
5.3 Benchmark Efficacy . . . 52
5.4 Online Evaluations . . . 52
5.4.1 Line-following Control . . . 52

6 Conclusion **59**
6.1 Limitations & Future Work . . . 60
6.1.1 Terrain-Aware Dynamics Modeling . . . 60
6.1.2 Worst-Case Benchmarking . . . 62
6.1.3 Closed-Loop Evaluation . . . 63

A Implementation & Analysis **65**
A.1 Perception Pipeline . . . 65
A.1.1 Mapping Resolution Analysis . . . 65
A.2 MPPI Implementation . . . 65
A.2.1 Cost Function . . . 65
A.2.2 MPPI Failure Analysis . . . 68
A.3 Learned Dynamics Model Implementation . . . 69
A.3.1 Network Architecture . . . 69
A.3.2 BEV Footprint . . . 70
A.3.3 Basestation . . . 71
A.3.4 Training Machines . . . 71

Bibliography **73**

When this thesis is viewed as a PDF, the page header is a link to this Table of Contents.

List of Figures

1.1 Robots with different dynamics in off-road environments, which were worked on during the course of this thesis. 2
1.2 Challenging characteristics of off-road driving. 3
1.3 The key thesis research questions and contributions in the scope of a full off-road navigation autonomy stack. 5

3.1 Terrain-aware dynamics models through perception conditioning. Leveraging the structure of the representation improves prediction accuracy, enabling stable control high-speed. 17
3.2 Different dynamics modeling formulations, ultimately leading to expressive and terrain-aware approaches through structured perception representations. 18
3.3 Perception and mapping flow diagram. 19
3.4 A simple proprioceptive dynamics model, to serve as the base architecture for perception-conditioned approaches. Also shown is an optional perception conditioning approach that baselines will use. 21
3.5 A simple proprioceptive predictor RNN cell. 23
3.6 An overview of our terrain-aware 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 3.7 details the structured representation formally. 24
3.7 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. 25
3.8 Comparing key difference between Vanilla BEV and our structured, spatial approach. We incrementally query a footprint around where we predict the robot to be, whereas Vanilla BEV encodes the entire map at once without leveraging the spatial structure of BEV maps. . 26
3.9 Vanilla BEV activations and our queries with reference BEV map and ground truth trajectory (white). Vanilla BEV's activations fall on irrelevant map regions; our queries fall along our prediction, which closely follows the robot's true path. 28

3.10 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) should various environments and motions. 29
3.11 The inpainting architecture of Triest et al. [81] creates fully filled BEV maps, predicting trails and shrubbery out of view. 30
3.12 When presented with denser, inpainted BEV maps, Vanilla BEV leverages even less relevant features, when compared to the robot's motion. On the contrary, our approach queries only where we predict to be and leverages the denser features completely. 31

4.1 Rigorous evaluation requires challenging data and a benchmark design to create challenging datasets. 33
4.2 Our dataset provides broad environment and season diversity. 34
4.3 Autonomous ATV platform, retrofitted with LiDAR, RGB stereo, thermal stereo, GNSS with RTK corrections, actuation, and wheel telemetry. 35
4.4 Renegade Ridge offers more geometric complexity, compared to Gascola, with a higher roll and pitch variance and steeper slopes with higher max pitch but lower top speeds. 38
4.5 Our dataset features environments encountered in all seasons. The same location in different seasons highlights the stark differences in visual and geometric appearance, as well as the terrain surface. 38
4.6 Similarly clustered actions yield different trajectories. Perception highlights how terradynamics are causing this difference. (A) travels further, going down a hill, as seen by the terrain map, (B) moves the least on flat dirt, while (C) travels further on flat, low-friction ice. . . 41

5.1 Comparing test and train dataset speed distributions, the test dataset has a higher density of higher speed. 46
5.2 Offline full state prediction error distributions across 3 datasets. The snow dataset proves most challenging for all methods with a high average prediction error. The speed dataset yields the most variance with the KBM performing poorly. Across all datasets, our method outperforms the baselines. 48
5.3 When using inpainted BEV inputs, our approach improves significantly, outperforming all baselines on all datasets on average. Inpainted BEV inputs improve our approach on all datasets on average. 49

5.4 Offline qualitative results highlighting interesting prediction examples and environments, including high-speed turns and deep snow traversal. Provided are the rollout predictions for all methods, an FPV image showing the environment, and scalar statistics to characterize the motion profile of the run, including inital velocity, average steer, and max pitch. 51
5.5 Evaluation on the highest to lowest variance deciles of the clustered benchmark. Model performance increases with lower variance deciles. 53
5.6 High-speed hardware experiments. Significant understeer can be seen by the KBM Baseline leading to numerous interventions (A,D). Additionally all learned methods, including ours, experience cases of understeer like in the tight turn in C. 55
5.7 Challenging geometry hardware experiments. Significant understeer can be seen by the KBM Baseline leading to numerous interventions (E) and additional understeer which did not lead to intervention (F,G). Baselines struggled to follow a tight turn (G), while our approach could follow it. 57

A.1 A comparison of BEV resolutions, zoomed in around a 10×10 m section of a BEV map. We use the 40 cm resolution. While the 10 cm resolution map increases sharpness, it is too sharp, and the LiDAR scan sparsity becomes apparent through the empty regions where no points were. 66
A.2 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. 68
A.3 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. 71

List of Tables

3.1 Ablation on spatial representations and why the correction step is a key component of our approach. 27

4.1 Dataset statistics 37

5.1 Offline position prediction results across 3 datasets, alongside the full test set. 48
5.2 When given inpainted BEV maps, indicated by red text, our approach has lowest average and worst-case prediction error across all datasets. 49
5.3 Hardware deployment results on the High Speed Course. 55
5.4 Hardware deployment results on the Geometry Course. 57
5.5 Offline position prediction results on closed-loop experiment data . . 58

A.1 Action Sampling Parameters 68
A.2 MPPI cost function parameters 68
A.3 Hyperparameters for the LSTM architecture and perception encoders. 70

Chapter 1

Introduction

1.1 Motivation

Autonomous robotic systems navigate the world to assist humans in the dull, dirty, and dangerous tasks of the transportation, construction, logistics, space, and defense sectors and beyond [34, 38, 66, 71, 82, 91]. As robots move through these varied environments, they need to reason about the consequences of their actions before taking them. Dynamics models provide exactly this capability—predicting how a robot's state evolves in response to its actions—and lay the foundation for robotic decision-making. They are prevalent among control approaches to predict and correct a robot's trajectory [66, 77] and in motion planners to determine physically feasible paths [30, 47]. They are common in model-based reinforcement learning (MBRL), where dynamics functions help policies evaluate actions [36, 72], as well as other paradigms that leverage dynamics to augment training data or simulate outcomes without costly real-world trials [11, 23, 45]. Across all of these fields, the quality of the dynamics model influences the quality of the decisions built on top of it.

In the context of control, particularly model-predictive control (MPC) [13, 22, 25, 63], a dynamics model predicts where a robot would move given a sequence of actions. This prediction can be used to enforce motion constraints or optimize over a cost function designed to accomplish a task, where modeling inaccuracy leads to suboptimal task completion or failure. One common source of inaccuracy is in incorrectly modeling a robot's interaction with its environment. For example, a

Figure 1.1: Robots with different dynamics in off-road environments, which were worked on during the course of this thesis.

wheeled robot's behavior depends on how its tires interact with the ground surface, while an aerial robot's depends on aerodynamic effects at its propellers. A model that fails to account for this environment interaction can diverge significantly from reality, but this interaction is difficult to model [3, 7, 43, 59].

While dynamics modeling spans many applications, each with its own challenges, this thesis focuses on aggressive off-road navigation for ground robots, as seen in Figure 1.1, as the core operational domain.

1.2 Challenges

Challenges in Off-Road Navigation

Off-road navigation presents many unique challenges because of the immense diversity and lack of structure in the operating environment. These challenges, shown in Figure 1.2, are stratified by having:

1. **Diverse Surface Cover**: a diverse taxonomy of ground cover on the terrain surface, including grass, mud, gravel, and standing water
2. **Geometric Complexity**: complex geometric features on the terrain surface, including steep slopes, embankments, ditches, ruts, lips, and berms
3. **Environment Diversity**: environmental variation across different regions and seasons, including inclement conditions, tight corridors, and open pastures
4. **High-Speed Operation**: demanding speed requirements, operating near the platform's limits to complete tasks quickly and efficiently

Traditional physics models, such as the kinematic bicycle model (KBM), fail to overcome these challenges because they lack expressivity to capture complex ter-

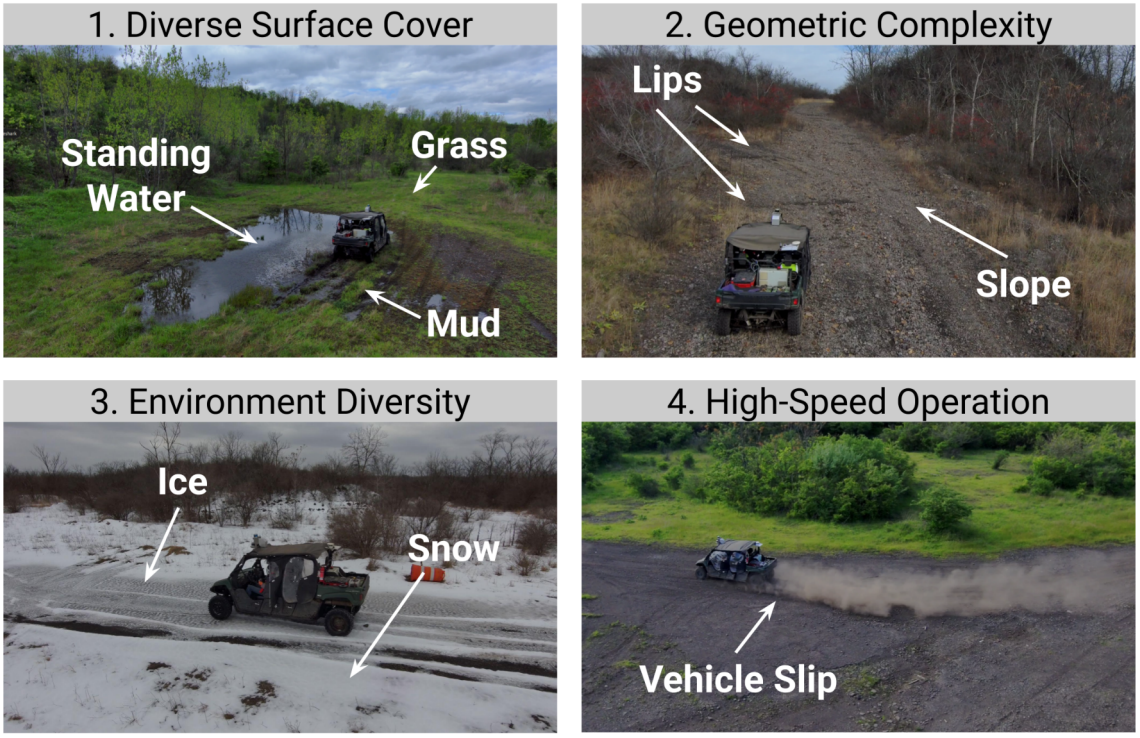


Figure 1.2: Challenging characteristics of off-road driving.

radynamics induced by the challenging terrains [25, 30, 37, 73]. Learned models conditioned on internal measurement, i.e. proprioception, have emerged as a potential solution with more expressivity, but they neglect environment observations and lack terrain-awareness [11, 19, 58, 92, 97]. Perception conditioning addresses this gap by supplying geometric and semantic information through birds-eye-view (BEV) or first-person-view (FPV) image representations, allowing the model to ground its motion predictions in what it observes of the surrounding terrain. BEV representations are especially useful because unlike a perspective image, the BEV map is a metric-space representation, with distance encoded explicitly in the spatial layout of the grid rather than left implicit. Yet prior works consume this perception once, as a static snapshot of the scene, rather than continually as the robot moves through it [43, 50, 68]. As a result, the model must implicitly infer which spatial features are relevant from a single observation, leaving the BEV's explicit spatial structure underutilized and limiting the model's terrain-awareness.

Challenges in Real World Deployment

A more accurate, terrain-aware dynamics model is only valuable if it improves a robot's behavior once deployed. Building such a model is a necessary first step, but two obstacles stand between it and reliable field performance.

First, a model that appears accurate on average is not necessarily one that is safe to deploy. A model deemed fit during evaluation may still fail on the edge cases that aggregate metrics can obscure. It is therefore unclear whether average-case performance alone is sufficient enough for rigorous evaluation, motivating an approach that actively seeks out the most challenging long-tail scenarios to test whether the model holds in the worst case as well as on average. This in turn requires collecting data that captures this long-tail, non-average behavior, and then designing a benchmark that mines the most difficult scenarios from it.

Second, even a rigorously validated model does not guarantee better closed-loop control. A large body of work, including this thesis, is devoted to reducing prediction error, yet it is worth stepping back to ask how much this effort ultimately matters. Open-loop prediction accuracy and closed-loop performance are not equivalent: a well-designed controller can be robust to certain modeling errors, such that beyond some threshold, additional model accuracy yields diminishing returns on the robot's actual behavior. Understanding where this threshold lies reveals how much modeling effort is truly warranted, and when it is better spent elsewhere.

1.3 Research Questions

Together, these gaps in modeling, evaluation, and closed-loop control motivate the key research questions which drive this thesis shown in Figure 1.3:

Research Question 1: How can dynamics models fully leverage the spatial structure of representations like BEV maps to achieve terrain-awareness and reduce prediction error across diverse and complex terrains?

Research Question 2: How can we rigorously expose the worst-case, long-tail failure modes of off-road dynamics models before real-world deployment?

Research Question 3: At what point does improved dynamics-model accuracy stop meaningfully improving closed-loop control performance?

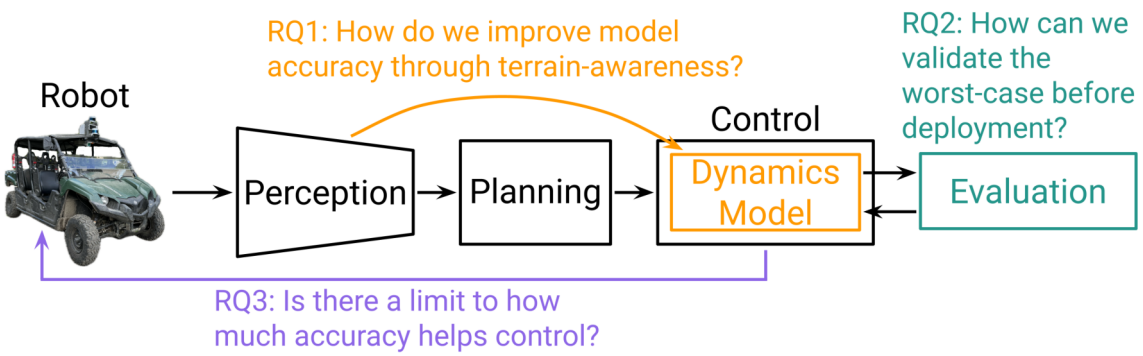


Figure 1.3: The key thesis research questions and contributions in the scope of a full off-road navigation autonomy stack.

1.4 Thesis Statement and Contributions

The work in this thesis describes novel improvements to the dynamics modeling and evaluation components of an autonomous off-road driving stack, deployed on a full-scale all-terrain vehicle (ATV). In this thesis, we will demonstrate that:

> **Thesis Statement**: Reliable navigation across diverse off-road environments requires terrain-aware dynamics models grounded in structured perception and validated against worst-case behavior in the open- and closed-loop.

To support this claim, this thesis makes three contributions:

1. A perceptually-aware dynamics model that leverages the spatial structure of BEV representations for terrain-aware prediction.
2. The collection of a challenging dataset with a benchmarking approach that mines its most difficult scenarios for rigorous evaluation.
3. A closed-loop study on our full-scale ATV that characterizes when improved model accuracy meaningfully affects control performance.

Chapter 2

Background

2.1 Preliminaries

Consider a robot at time t with a state $x_t \in \mathcal{X}$ and an action $u_t \in \mathcal{U}$. Given a sequence of actions $u_{t:t+T-1}$, a deterministic dynamics model f predicts a sequence of states $x_{t+1:t+T}$ known as a trajectory τ or a rollout. Under this definition, Section 2.2 will describe how related works use dynamics models for mobile robotics applications and how dynamics models are formulated to predict accurate rollouts. Next, we will formulate MPC for off-road driving in Section 2.3 and then briefly explain more deeply how base physics models can be derived Section 2.4.

2.2 Related Work

2.2.1 Dynamics Models for Mobile Robots

Dynamics models are used across many mobile robotics algorithms spanning motion control, kinodynamic planning, and reinforcement learning. Williams et al. [90] roll out thousands of sampled action sequences through a dynamics model in a sampling-based MPC formulation to select optimal trajectories over a given horizon. Hedegaard et al. [28], Likhachev and Ferguson [47] instead define a planner's search space with a library of dynamically feasible motion primitives generated from model rollouts. Dynamics models also have value for data augmentation. Xiao et al. [92] pretrain

on trajectories from simple physics models, while Levy et al. [45] generate synthetic rollouts from a learned model to accelerate real-world MBRL policy optimization.

Inaccurate dynamics models limit the success of these algorithms. As such, there is a large corpus of research in dynamics modeling using physics-based or learned methods for these tasks. While physics-based, parametric models exist for autonomous driving domains [26, 63, 73], they lack expressivity, breaking down under invalid assumptions, and require a large amount of time to tune. Learned dynamics models are a potential solution to these weaknesses.

2.2.2 Learning Physics-informed Dynamics Models

Both the learning objective and representations for the dynamics model are critical for robust control on varying terrains. Chrosniak et al. [10], Djeumou et al. [14] choose to learn the parameters of a physics-based 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. [20] address by learning the force residual to a tire model. Alternatively, Lee et al. [43] learn a velocity state and use explicit kinematics to integrate it into the robot's pose. Across many variants of dynamics residual learning, they all supplement the physics model by learning any unmodeled disturbances and are widely in literature for off-road platforms [19, 43, 44, 50, 51, 58], on-road platforms [15, 93], and aerial vehicles [54, 59], 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.

2.2.3 Proprioceptive Learned Dynamics Models

Deterministic Dynamics

Thus, much like MBRL [35, 55, 85], it is common to learn the deterministic forward states directly, without any physics-informed prior. This is most beneficial to remove primitive physics-based assumptions which may be inaccurate or for models which try to learn the general dynamics of a broader locomotion class. Xiao et al. [92]

learn the state directly to try to learn a general Ackermann-steer dynamics model, which can predict dynamics under any load or form factor. Kumar et al. [42] learn the dynamics of a quadruped without any physics-based constraints to formulate an environment-adaptable policy. Many other works deploy deterministic dynamics learning across various embodiments [4, 31, 32, 39, 62, 78, 79, 87].

Probabilistic Dynamics

These state representations can be further formulated as probabilistic or stochastic, predicting a distribution of states, instead of a single deterministic state. This is advantageous when combined with MPC or RL frameworks to represent risk via dynamics prediction uncertainty. For example, Fan et al. [17] learn quantiled distributions of predictions for use in a tube MPC framework on quadrotors. Chua et al. [11] use ensembles to study how a uncertainty can be embedded into dynamics models for MBRL and be helpful for preventing overfitting during training. Wang et al. [86] use probabilistic predictions for a risk-aware MPC formulation, and many other works use probabilistic state representations for dynamics modeling [12, 16, 41, 45, 73, 95].

2.2.4 Exteroceptive Learned Dynamics Models

To improve terrain awareness, dynamics models can be augmented with exteroception, i.e. external measurements, such as geometric and semantic perception representations including FPV images and BEV maps. Prior work conditions dynamics models on FPV images [5, 8, 23, 35], point clouds [39], or BEV representations, specifically elevation maps [20, 43, 68], RGB maps [51, 61], or VFM feature maps [20, 50]. However, most prior work treats perception as a fixed context signal with one input encoded into the latent state at the beginning of the forward prediction. Lupu et al. [50], Roth et al. [68] encode full BEV maps and defer relevant context extraction to the encoder. Lee et al. [43] instead crop the BEV map around the robot but over a large extent, encompassing a conservative region of feasible motion rather than predicted motion. Most similar to our approach, Gibson et al. [20] query features at tire points along the rollout; however, they use this context as a force residual integrated through a physics model and no further. In contrast, our approach takes full advantage of the spatial BEV representation by incrementally querying the local

features at each prediction step and *using those features to directly correct each predicted state*, which contributes substantially to prediction accuracy (Section 3.4.2).

World models offer a complementary perspective, learning latent representations of environment dynamics jointly with a predictive model [23, 35]. In this framing, perceptual inputs are not appended features but instead serve as the primary signal from which the model must infer the latent state of the world [5]. While these approaches demonstrate strong performance in visually rich settings, they are predominantly trained in simulation or constrained FPV settings and do not exploit the geometric consistency afforded by BEV representations.

2.3 Model-Predictive Control

MPC is widely used in robotics for trajectory planning and tracking. Standard MPC prescribes an optimization problem with the objective of finding the optimal control sequence that minimizes a cost function. Its appeal lies in its ability to incorporate dynamics constraints directly into the optimization, use predicted states to evaluate costs over a finite horizon, and accommodate additional objectives such as obstacle avoidance or stability costs with minimal reformulation. One such formulation is given in (2.1) which minimizes a running cost $\ell(x_t, u_t)$ over a prediction horizon of T steps, plus a terminal cost $\ell_T(x_T)$ evaluated at the final predicted state, subject to dynamics constraints (2.2).

$$\mathbf{u}^* = \underset{u_{0:T-1}}{\operatorname{argmin}} \quad \sum_{t=0}^{T-1} \ell(x_t, u_t) + \ell_T(x_T) \tag{2.1}$$

$$\begin{aligned} \text{s.t.} \quad & x_{t+1} = f(x_t, u_t) && (2.2)\\ & x_0 = x(t) \\ & x_t \in \mathcal{X} \\ & u_t \in \mathcal{U} \end{aligned}$$

Many cost terms can be incorporated into the running cost $\ell(x_t, u_t)$ such as obstacle proximity, path deviation, or smoothness penalties. For wheeled robots,

relevant constraints may include bounds on lateral acceleration, speed, or actuator effort. Terminal costs ℓ_T are commonly used to encode goal-reaching objectives or to approximate the cost-to-go beyond the horizon.

From this formulation, two components critically determine MPC performance:

1. **Cost function**: The cost function encodes the desired behavior by assigning high cost to undesirable states and low cost to desirable ones. A poorly specified cost function—for example, one that assigns low cost to obstacles—will produce trajectories that actively steer toward them. Defining effective cost functions and cost maps remains an active area of research [26, 76, 80].
2. **Dynamics model**: The dynamics model determines the predicted states used to evaluate cost. If these predictions are inaccurate, the optimizer may select an action sequence that appears optimal under the model but results in undesirable behavior on the real system—for instance, predicting obstacle clearance where a collision actually occurs. Improving the accuracy of the dynamics model is the focus of this thesis.

While MPC improves upon controllers such as PID or Linear Quadratic Regulator (LQR) by easily incorporating constraints and multi-step lookahead, conventional MPC typically requires convex cost functions so that the resulting optimization has a unique global minimum and can be solved efficiently with standard solvers such as Newton's Method. This formulation is insufficient for problems with non-convex costs, such as costmap-based navigation, motivating sampling-based MPC approaches.

2.3.1 Sampling-based Model-Predictive Control

Sampling-based MPC forgoes the necessity of convex cost functions and constraints by replacing gradient-based optimization with parallel evaluation of sampled action sequences. One widely used variant of sampling-based MPC is Model Predictive Path Integral control (MPPI) [89, 90], which samples multiple action sequences from a given distribution, evaluates the cost of each resulting trajectory through the dynamics model, and computes the optimal action sequence as the cost-weighted average over all samples. Because MPPI requires only cost evaluation on forward simulated trajectories via the dynamics model, not gradient computation, it naturally accommodates non-convex and non-differentiable cost functions. Another significant benefit of MPPI is

its support for parallel computation; since each sampled action sequence, or particle, is independent of one another, we can leverage GPU-parallelizable computation to increase computation efficiency.

Formally, for $i = 1, \ldots, K$ samples, MPPI draws perturbations from a distribution (2.3) and applies them to the previous solution $\bar{u}_t$ to yield candidate action sequences (2.4). A dynamics model computes the forward predicted states (2.5) and a possibly non-differentiable, non-convex cost is computed on the resulting trajectories (2.6). The cost function determines the weight of each trajectory (2.7), modulated by a temperature parameter λ. The MPPI solution is then computed as the weighted average of the sample action sequences (2.8).

$$\epsilon_{u,t}^{(i)} \sim \mathcal{N}(0, \Sigma) \tag{2.3}$$

$$u_t^{(i)} = \bar{u}_t + \epsilon_{u,t}^{(i)} \tag{2.4}$$

$$x_{t+1}^{(i)} = f(x_t^{(i)}, u_t^{(i)}), \quad x_0^{(i)} = x(t) \tag{2.5}$$

$$J^{(i)} = \sum_{t=0}^{T-1} \ell(x_t^{(i)}, u_t^{(i)}) + \ell_T(x_T^{(i)}) \tag{2.6}$$

$$w^{(i)} = \exp\left(-\tfrac{1}{\lambda} J^{(i)}\right) \tag{2.7}$$

$$\bar{u}_t \leftarrow \frac{\sum_i w^{(i)} u_t^{(i)}}{\sum_i w^{(i)}} \tag{2.8}$$

yielding the updated nominal sequence solution $\bar{\mathbf{u}} = [\bar{u}_0, \ldots, \bar{u}_{T-1}]$. Our MPPI implementation is detailed in Section A.2.1.

Temperature Modulation

The temperature modulation allows for an exploration versus exploitation paradigm, where a high temperature yields exploration because the exponential weight differences between low-cost and high-cost trajectories are smoothed out. This is useful when the dynamics are uncertain and the robot cannot be greedy; however, in multimodal cost landscapes such as a "fork-in-the-road" with an obstacle in the middle, the high temperature may produce a blended trajectory that collides with the obstacle instead of picking a path. Lower temperature yields exploitation with a sharp, aggressive

"filter" around lowest-cost trajectories. While this avoids the mode-averaging failure, it is much more sensitive to sampling noise and prone to get stuck in local optima if the sampled batch misrepresents the true cost landscape.

Sampling Strategies

While MPC depends on the cost function and dynamics model, MPPI additionally depends on the sampling strategy used to generate candidate action sequences. The base formulation (2.3) draws i.i.d. Gaussian noise independently at each timestep, which can produce temporally uncorrelated perturbations. A Gaussian walk, used in [6, 84], instead accumulates noise across the horizon via a cumulative sum (2.9), introducing temporal correlation that yields smoother candidate trajectories. Alternatively, a deterministic action library (2.10) bypasses stochastic sampling entirely; a fixed set of K precomputed action sequences $\mathcal{A}$ can be evaluated at every MPPI iteration, providing consistent coverage of the action space and improving exploration of local minima in the cost function. These sampling strategies can be composed together to cover multiple action distributions, encompassing a large diversity of action sequences.

$$\epsilon_{u,t}^{(i)} = \sum_{j=0}^{t} \delta_j^{(i)}, \quad \delta_j^{(i)} \sim \mathcal{N}(0, \Sigma) \tag{2.9}$$

$$u_t^{(i)} = a_t^{(i)}, \quad a^{(i)} \in \mathcal{A} \tag{2.10}$$

2.4 Dynamics Modeling

We next detail preliminary dynamics modeling approaches and necessary preliminaries for learned proprioceptive and exteroceptive approaches. Given a current state x_t and action u_t the next state x_{t+1} is governed by the deterministic robot dynamics f such that $x_{t+1} = f(x_t, u_t)$. For a sequence of T actions forward in time $u_{t:t+T-1}$, f predicts the future states $x_{t+1:t+T}$. We augment the model with proprioceptive context of K prior timesteps of state and action history $x_{t-K:t}$ and $u_{t-K:t-1}$ to form the context $c_t = [x_{t-K:t}, u_{t-K:t-1}]$. Additionally, to capture terrain and environment

conditions not encoded in state alone, the robot receives an observation $o_t \in \mathcal{O}$ from onboard sensors such as BEV maps or FPV images. Together, the dynamics function with context and observations is given as $x_{t+1} = f(x_t, u_t, c_t, o_t)$.

The dynamics model can take multiple forms, such as a proprioceptive physics model $f(x, u)$ (Section 2.4.1), a proprioceptive learned model $f_\theta(x, u, c)$ (Section 3.3), or an exteroceptive learned model $f_\theta(x.u, c, o)$ (Section 3.4).

2.4.1 Physics Modeling

Physics models are commonly employed for on-road driving scenarios where the ground surface is benign and a few friction or tire parameters can adequately encompass the locomotion of a wheeled robot. Such models can be kinematic models, which consider a robot's velocity only, and dynamic models, which consider the forces imparted on a robot[1]. These models are frequently simplified into single-track or "bicycle" models; however, double-track [9] or other forms are possible [64]. Approaches beyond single-track are not common for off-road modeling since single-track models are already dominated by unmodeled terradynamics, leaving little to gain from the added complexity of double-track.

Kinematic Models

Kinematic Bicycle Model A KBM describes motion purely through geometric relationships, without modeling forces or actuator dynamics. Given a steering angle δ and a longitudinal velocity v, the KBM computes the vehicle's motion by assuming the wheels roll without slip. For a front-steered vehicle with wheelbase L, the continuous equations of motion are in (2.11) and can be made discrete using integration techniques such as Forward Euler or 4^{th}-order Runge-Kutta.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} v\cos(\theta) \\ v\sin(\theta) \\ \frac{v}{L}\tan(\delta) \end{bmatrix} \tag{2.11}$$

[1]"dynamics models" often becomes an overloaded term in robotics, which just means "motion model," but formally, a dynamic model deals with dynamics, e.g. forces.

The no-slip assumption makes the KBM computationally efficient and adequate at low speeds, but increasingly inaccurate as tire slip or terrain effects becomes significant—for example, during high-speed cornering or on low-friction surfaces.

Extended Kinematic Bicycle Model In practice, we have modified these equations of motion to yield an "Extended KBM," which incorporates actuator dynamics while retaining the simplicity of a kinematic formulation (2.12). We model longitudinal acceleration through throttle dynamics, encompassing a throttle term k_{th}, a drag term k_d, a rolling resistance term k_{RR} and a gravity term k_g. We model steering actuation as a first-order lag with gain k_δ. We also use pitch ϕ from our odometry and we primitively assume the pitch remains constant for the entire rollout.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{v} \\ \dot{\delta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} v\cos(\theta)\cos(\phi) \\ v\sin(\theta)\cos(\phi) \\ \frac{v}{L}\tan(\delta) \\ k_{th}u_{\text{th}} - k_d v - k_{RR}\text{sgn}(v)\cos(\phi) - k_g g\sin(\phi) \\ k_\delta(u_\delta - \delta) \\ 0 \end{bmatrix} \tag{2.12}$$

Dynamic Models

Dynamic Bicycle Models (DBM) model forces acting on the robot, most notably the tires [19, 63]. This often improves accuracy for on-road cases at the cost of tuning difficulty and efficiency. The DBM considers lateral and longitudinal tire forces at each axle, typically modeled through a tire model such as a linear tire model or Pacejka's Magic Formula [60]. This formulation captures phenomena absent from the KBM, including tire slip, vehicle sideslip, and yaw rate dynamics, making it suitable for higher-speed operation and limit handling.

For off-road driving, the challenge is compounded: terrain properties such as surface friction, deformability, and roughness vary spatially and are difficult to parameterize a priori. Even a well-identified double-track model calibrated on pavement may fail on gravel, mud, or grass, where tire-terrain interaction departs significantly from on-road assumptions. This motivates data-driven approaches that learn dynamics directly from experience.

Chapter 3

Terrain-Aware Dynamics Models

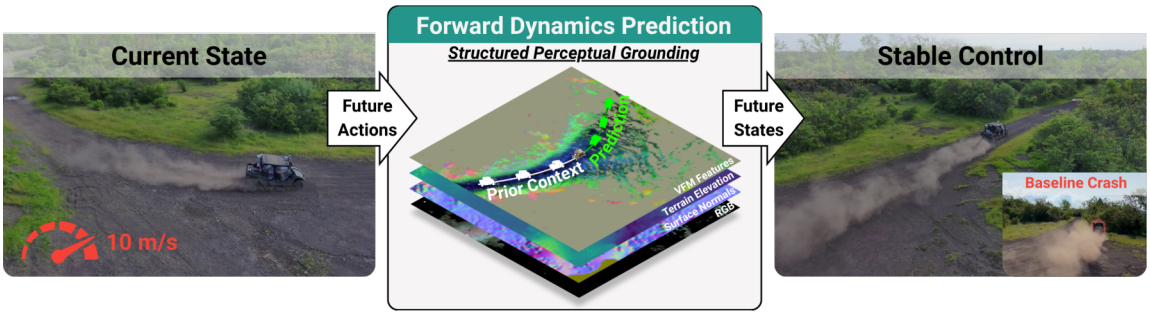


Figure 3.1: Terrain-aware dynamics models through perception conditioning. Leveraging the structure of the representation improves prediction accuracy, enabling stable control high-speed.

3.1 Introduction

This chapter is driven by research question 1: *How can dynamics models fully leverage the spatial structure of representations like BEV maps to achieve terrain-awareness and reduce prediction error across diverse and complex terrains?* To this end, we will discuss how dynamics models can be made more expressive through model learning and terrain-aware through conditioning on perception observations of a robot's environment.

First, we briefly discuss available perception inputs for conditioning in Section 3.2. Then, we will discuss the approaches to learn the model f_θ as outlined in Figure 3.2,

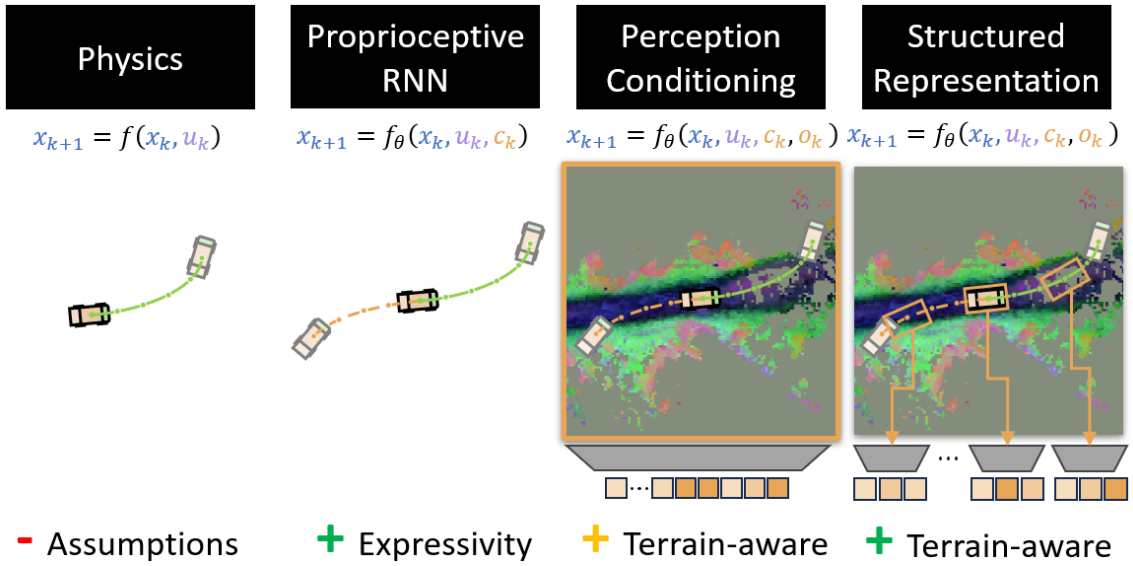


Figure 3.2: Different dynamics modeling formulations, ultimately leading to expressive and terrain-aware approaches through structured perception representations.

so that we can answer **RQ1**. First, we will define a simple proprioceptive network in Section 3.3 to improve model expressivity of physics approaches. This will lay the foundation for perception-conditioned models. Next, we will discuss how we can improve perception conditioning to be spatially structured in Section 3.4 and therefore improve terrain awareness. Finally, Section 3.5 uses inpainting to motivate our approach, showing that denser perception inputs do not always mean better environment context for simpler perception-conditioned approaches.

3.2 Perception Pipeline

Terrain-awareness can significantly be enhanced via perceptual observations of the environment. Thus, we detail our perception and mapping pipeline, which turns raw sensor inputs like pointclouds and FPV RGB images into processed, geometrically and semantically rich modalities for terrain-aware dynamics learning or other tasks like costmap learning [18, 76, 80]. We leverage a GPU-accelerated local mapping package, largely architected by Triest et al. [80, 81], that produces semantic images and geometric-semantic BEV maps from camera and LiDAR data. Figure 3.3 depicts this procedure, and we detail it briefly below.

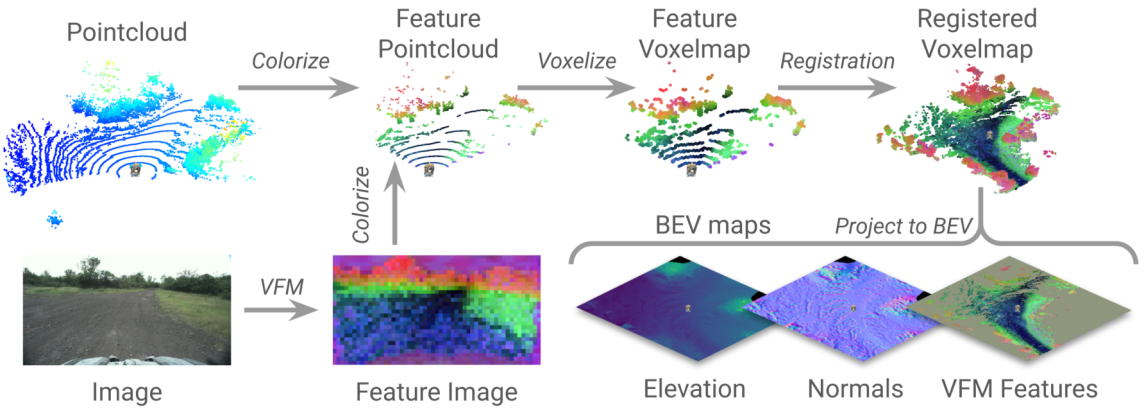


Figure 3.3: Perception and mapping flow diagram.

3.2.1 Feature Mapping

Salient visual features are extracted from the front-facing camera images using Visual Foundation Models (VFMs) such as c-Radiov3 [65] with a SIGLIP2 adaptor [94] and NACLIP attention [24] and are then passed through a domain-specific principal component analysis (in a similar fashion to RayFronts [1]). This results in a "feature image" with visual semantics. DINOv2 [57] is another common VFM which is used, but we found c-Radiov3 with SIGLIP2 to be better. This extraction is defined as as a mapping from an image I to feature image D via the VFM extractor f_{VFM}:

$$f_{\text{VFM}}(I_{W\times H\times 3}) = D_{W_f\times H_f\times F} \tag{3.1}$$

Figure 3.3 shows an example of this feature image, where the trail appears blue, the grass appears green, and the shrubbery appears pink. This is purely a visualization of the top three principal components for human interpretation. In practice, we map a $1024 \times 544 \times 3$ image to a $53 \times 28 \times 32$ feature image.

3.2.2 Colorizing

We lift this feature image into 3D via pointcloud colorization, in which the LiDAR pointcloud is projected into the feature image frame using standard camera geometry techniques, given a camera intrinsic matrix, $\mathbf{K}$, and the extrinsic rotation $\mathbf{R}$ and translation $\mathbf{t}$ between the two sensors (3.2). Points that lie within the image frame

inherit the feature from the pixel that they fall into, resulting in a "feature pointcloud". This obtains the corresponding image coordinates (u, v) for each point $P = (x, y, z)$. Likewise, we can do the same for an RGB image.

$$[u', v', w]^\intercal = \mathbf{K}[\mathbf{R}|\mathbf{t}][x, y, z, 1]^\intercal, \quad u = \frac{u'}{w}, \quad v = \frac{v'}{w} \tag{3.2}$$

3.2.3 Voxelizing and Registration

The feature pointcloud is then transformed into a fixed frame via an on-board LiDAR-interial odometry algorithm [96], voxelized, and aggregated as more voxel maps are captured, creating a registered voxel map centered at the robot's current position. More details are present in Triest et al. [81]. In practice, we voxelize our pointclouds using a $0.4 \times 0.4 \times 0.1\,\text{m}$ voxel resolution, over a $100 \times 100 \times 20\,\text{m}$ spatial extent, centered around the vehicle ($50\,\text{m}$ Chebyshev e.g. square radius along x and y, and $10\,\text{m}$ along z), yielding a voxel grid of dimension $N_x \times N_y \times N_z = 250 \times 250 \times 200$ cells. This spatial extent is suffice for most of our needs; however when driving at high-speed ($\geq 10\,\text{m/s}$), the dynamics model rollout will occasionally go beyond the extent, meaning, there is no geometric or semantic information to leverage. We additionally choose to use this resolution and spatial extent largely because most of our prior work for traversability estimation works with BEV maps of the same resolution and extent. More analysis on the BEV resolution can be found in Section A.1.1.

3.2.4 BEV Projection

Finally, we project the registered voxel grid into BEV space to yield a BEV map $\mathbf{B} \in \mathbb{R}^{N_x \times N_y \times F_b}$. We use the registered voxel grid to estimate the underlying terrain surface of the robot's local environment using a Markov random field [53, 88]. 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 $= 1.7\,\text{m}$) of the terrain surface. In practice, our BEV maps stack 36 channels with 1 elevation channel, 3 surface normal channels, and 32 VFM feature channels.

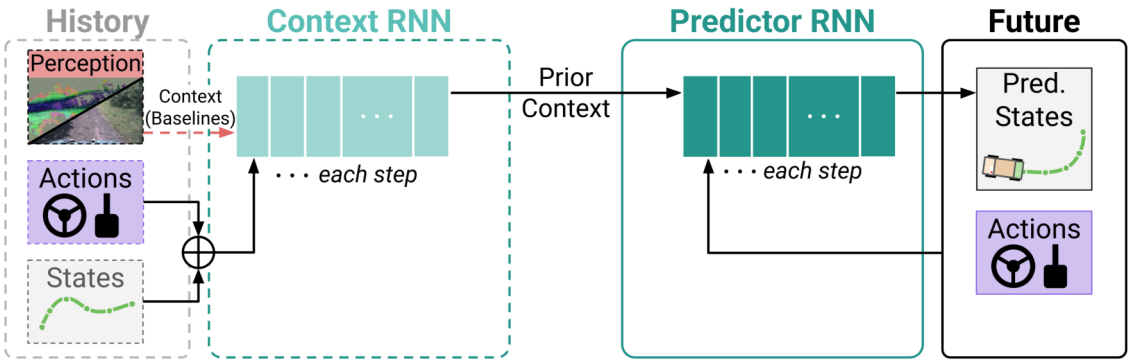


Figure 3.4: A simple proprioceptive dynamics model, to serve as the base architecture for perception-conditioned approaches. Also shown is an optional perception conditioning approach that baselines will use.

3.2.5 BEV Preprocessing

While these BEV maps are sufficient for dynamics learning, we apply two minor preprocessing steps before inference to reduce the load of the model:

1. **Ego Alignment**: 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 up in the BEV map. 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. **Elevation Cropping**: We modify the features to limit local terrain elevation from $[-10\,\text{m}, 10\,\text{m}]$ and we rotate geometric data, such as surface normals, to be aligned in ego frame.

3.3 Proprioceptive Modeling

We detail the base proprioceptive model architecture. While proprioceptive models lack terrain-awareness, we detail learned proprioceptive dynamics models as a foundational architecture for forward prediction, action encoding, and decoding. We architect the core dynamics model as a Recurrent Neural Network [70]. Similar to Gibson et al. [19], we compose it of two components, depicted in Figure 3.4:

1. A Predictor Network, composed of RNN cells Φ_P which takes an action sequence with prior context to predict the future states
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 later, current observations.

We choose to use an RNN rather than a Transformer [83] because its sequential recurrence lets us insert our structured spatial representation between cells, while keeping the memory footprint small.

For time t, the predictor cell takes action $\mathbf{u}_t$, encoded as $\mathbf{z}_t$ (3.3), as input with a prior latent state initialization $\mathbf{h}_t$ to predict the next hidden state via the prediction RNN cell Φ_P and recurrently produces this output for $\mathbf{u}_{t:t+T-1}$ (3.4), shown in Figure 3.5. Similar to Nagabandi et al. [55], 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 $\mathbf{x}_{t+1} = \mathbf{x}_t + f_\theta(\mathbf{x}_t, \mathbf{u}_t, c_t)$. For our RNN representation, we decode $\mathbf{h}_t$ via ϕ^x to produce our $\Delta\mathbf{x}_t$ (3.5) and apply the state difference step to yield our next predicted state (3.6). This is advantageous when predicting states which are too similar [55]. Predicting derivatives over small timesteps would yield difficult to learn, noisy signals, whereas state differences do not encounter this shortcoming. This localizes the target near zero for a better learnable signal.

$$\mathbf{z}_t = \phi^u(\mathbf{u}_t) \tag{3.3}$$

$$\mathbf{h}_{t+1} = \Phi_P(\mathbf{h}_t, \mathbf{z}_t), \quad \text{for } t = 0, \ldots, T \tag{3.4}$$

$$\Delta\mathbf{x}_{t+1} = \phi^x(\mathbf{h}_{t+1}) \tag{3.5}$$

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \phi^x(\mathbf{h}_{t+1}) \tag{3.6}$$

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 $\mathbf{u}_{t-K:t-1}$, it initializes the predictor's hidden state $\mathbf{h}_0$ (3.7) 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

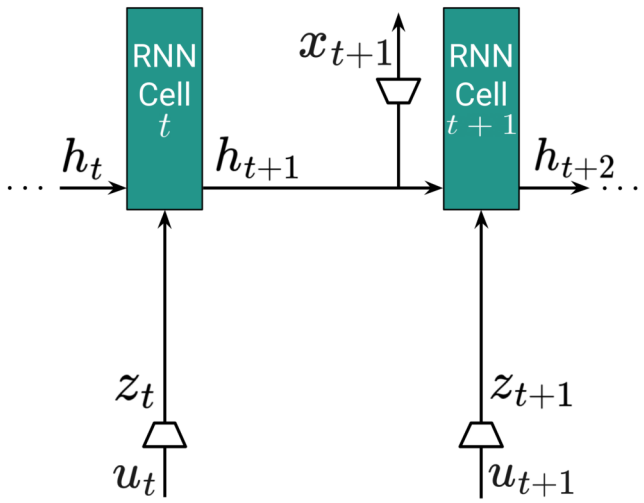


Figure 3.5: A simple proprioceptive predictor RNN cell.

variable ground cover need perception for greater information on the terrain ahead.

$$\mathbf{h}_{k+1} = \Phi_C(\mathbf{h}_k, \mathbf{z}_k, c_k), \quad \text{for } k = -K, \ldots, -1, \quad \mathbf{h}_{-K} \triangleq \vec{0} \tag{3.7}$$

3.4 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 [7]. Prior work [43, 50, 68] has incorporated perceptual conditioning through inputs such as FPV images or BEV maps as fixed context signals, denoted as "Vanilla BEV" for the BEV variant. 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 our streamlined perception pipeline relying on LiDAR pointclouds for geometric features and map registration and FPV imagery for semantic features, detailed in Section 3.2.

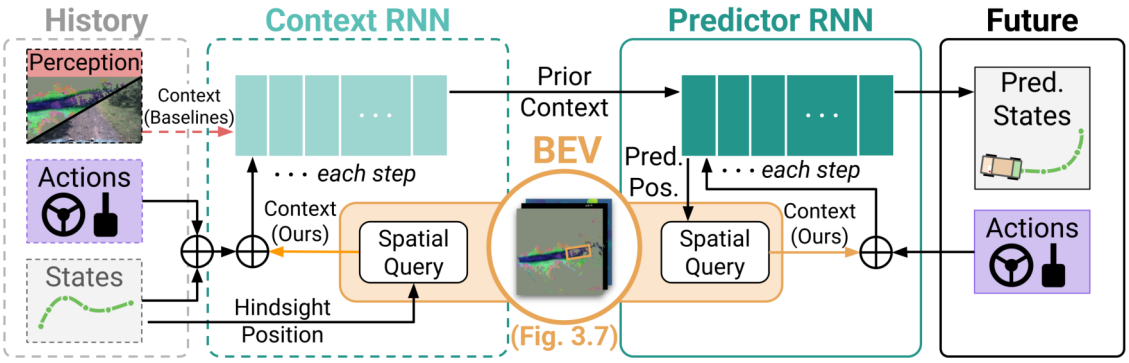


Figure 3.6: An overview of our terrain-aware dynamics model. We query each prior or predicted position in the static, structured BEV map at each step. This provides geometric and semantic context to our model for more accurate dynamics rollouts. Figure 3.7 details the structured representation formally.

3.4.1 Recurrent Spatial 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 3.7).

Concretely, given our BEV map $\mathbf{B}$ and the hidden state prediction $\hat{\mathbf{h}}_{t+1}$ at step $t+1$, predicted 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 $\mathcal{F}(\hat{\mathbf{x}}_{t+1}) \subset \mathbb{R}^2$ and use it to sample over $\mathbf{B}$ to obtain the BEV feature context vector $\mathbf{b}_{t+1}$ in (3.8) where $\mathbf{B}[\cdot]$ denotes bilinear interpolation of the BEV grid at the footprint points and ϕ^b is a learned encoder.

This context is used in two key manners. Firstly, 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

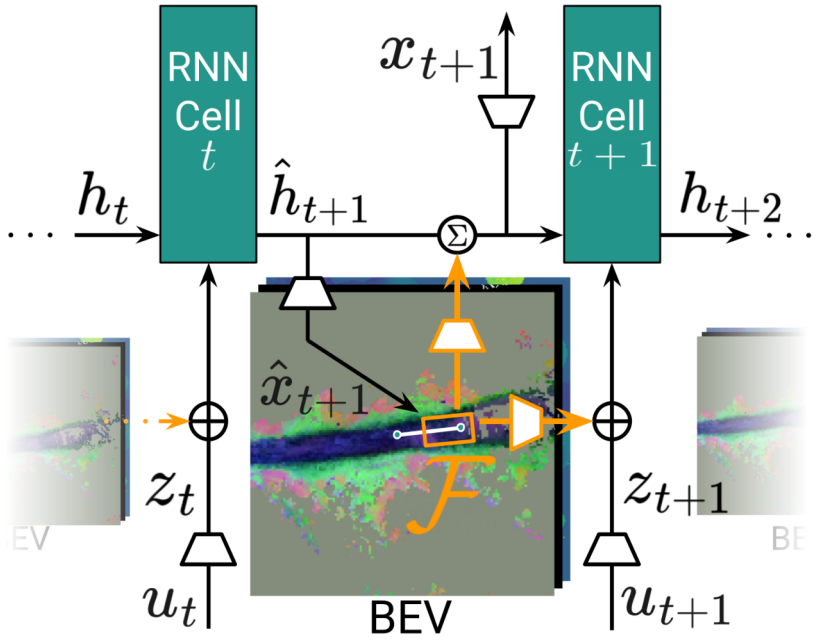


Figure 3.7: Recurrent BEV Cells. At each step, the next predicted state determines a footprint over the BEV map. Sampled features are passed into the subsequent cell.

in (3.9).

$$\mathbf{b}_{t+1} \triangleq \phi^b(\text{vec}\left(\mathbf{B}[\mathcal{F}(\hat{\mathbf{x}}_{t+1})]\right)) \tag{3.8}$$

$$\mathbf{x}_{t+1} = \mathbf{x}_t + \phi^x\left(\Phi_P\left(\mathbf{h}_t, \mathbf{z}_t, \mathbf{b}_t\right) + \mathbf{b}_{t+1}\right) \tag{3.9}$$

The key difference between our structured, spatial BEV approach and the Vanilla BEV approach is in this spatial query. As shown in Figure 3.8, the Vanilla BEV approach queries the entire BEV map at once to initialize the Context RNN. In contrast, our Spatial BEV approach incrementally queries at each prediction locality using an explicit footprint, extracting semantically and geometrically meaningful features to serve as an incremental BEV context.

Spatial Attention Analogy

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 [46] assigns learned weights to every spatial location and recovers footprint locality through training. Learned offset approaches such

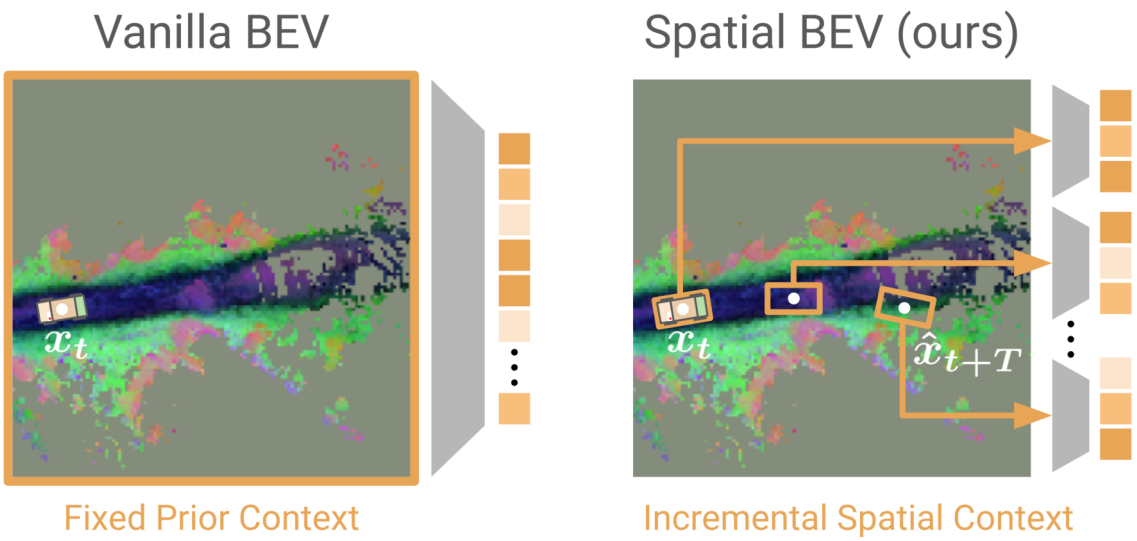


Figure 3.8: Comparing key difference between Vanilla BEV and our structured, spatial approach. We incrementally query a footprint around where we predict the robot to be, whereas Vanilla BEV encodes the entire map at once without leveraging the spatial structure of BEV maps.

as deformable attention [98] learn to predict where to sample. Our formulation reflects that of Jaderberg et al. [33], 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. More details on our footprint can be found in Section A.3.2.

3.4.2 Correction through Observation

The correction step, while simple in practice, plays a large role in model performance. The key intuition behind the correction step is to use observations at the predicted location to determine if, given this new information, this prediction is realistic. If it is not realistic, this correction step applies a correction to the hidden state $\mathbf{h}_t$, given this new BEV context observation. So, instead of relying on one observation at the current time t, this architecture relies on the BEV contexts $\mathbf{b}_t$ and $\mathbf{b}_{t+1}$, computed from the current state $\mathbf{x}_t$ and predicted state $\hat{\mathbf{x}}_{t+1}$, respectively.

Table 3.1 ablates the correction step in our perception representation across three

	Snow		Slopes		Speed	
Method	Mean	95%	Mean	95%	Mean	95%
Non-corrective Spatial BEV	3.54 ± 3.50	10.52	2.15 ± 3.61	8.86	1.89 ± 4.39	6.25
Corrective Spatial BEV	**2.45 ± 2.66**	**8.06**	**1.77 ± 2.54**	**6.91**	**1.50 ± 3.90**	**5.42**

Table 3.1: Ablation on spatial representations and why the correction step is a key component of our approach.

challenging sub-datasets, which are later defined in Section 5.2.1. Removing the correction step consistently increases prediction error, with a 22.9% rise in average error across datasets. This confirms that the correction step is essential for updating state predictions based on encountered terrain, particularly for trajectories where terrain features evolve significantly over time.

3.4.3 Intuition on Structured Representations

It is clear to understand why this explicit structure can be advantageous, given Figure 3.8, because the aforementioned vanilla approach may struggle to encode the most relevant information, given an entire BEV map with no additional context. The Vanilla BEV is at a disadvantage because the BEV encoder:

1. Disregards the spatial structure of the BEV representation
2. Needs to learn the robot motion to extract the most relevant data
3. Wastes computation on cells without data

Figure 3.9 visualizes these disadvantages, where we compute the activation maps of the Vanilla BEV approach to see where the encoder is "attending" to to create its context vector. Vanilla BEV's activations fall on irrelevant map regions, outside of the robot's motion and on nonsensical features such as trees or bushes. Meanwhile, our queries fall along our prediction, which closely follows the robot's true path.

Figure 3.10 further supports these claims with visualization across many samples. Across five unique examples, Vanilla BEV has high activation weight arbitrary or irrelevant sections of the input, while our approach queries features almost exactly where the robot went. (A) shows high activation on much of the VFM-covered region, including where the robot went; this is one of the better examples for Vanilla BEV. Our approach, on the other hand, queries nearly exactly and only where the robot

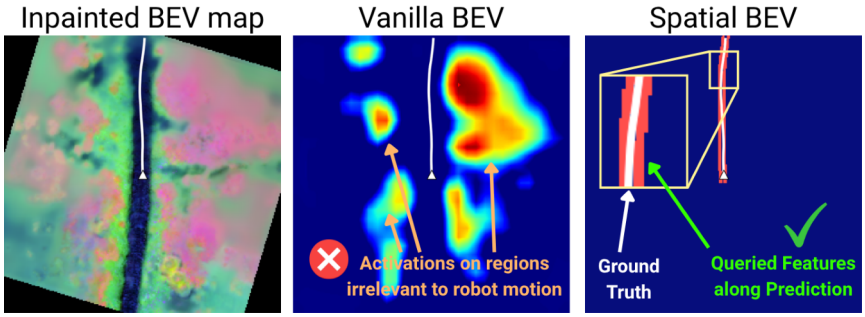


Figure 3.9: Vanilla BEV activations and our queries with reference BEV map and ground truth trajectory (white). Vanilla BEV's activations fall on irrelevant map regions; our queries fall along our prediction, which closely follows the robot's true path.

went. (B) through (E) show poor attention to relevant regions for Vanilla BEV where it appears to attend to regions around where the robot went, but not on, or at the edge of observation limits. Meanwhile, our approach consistently queries close to where the robot went.

We create these activation maps using GradCAM [74], 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 3.10. We choose to back-propagate from the x position because we saw, qualitatively, the activations were similar between state fields and along the rollout.

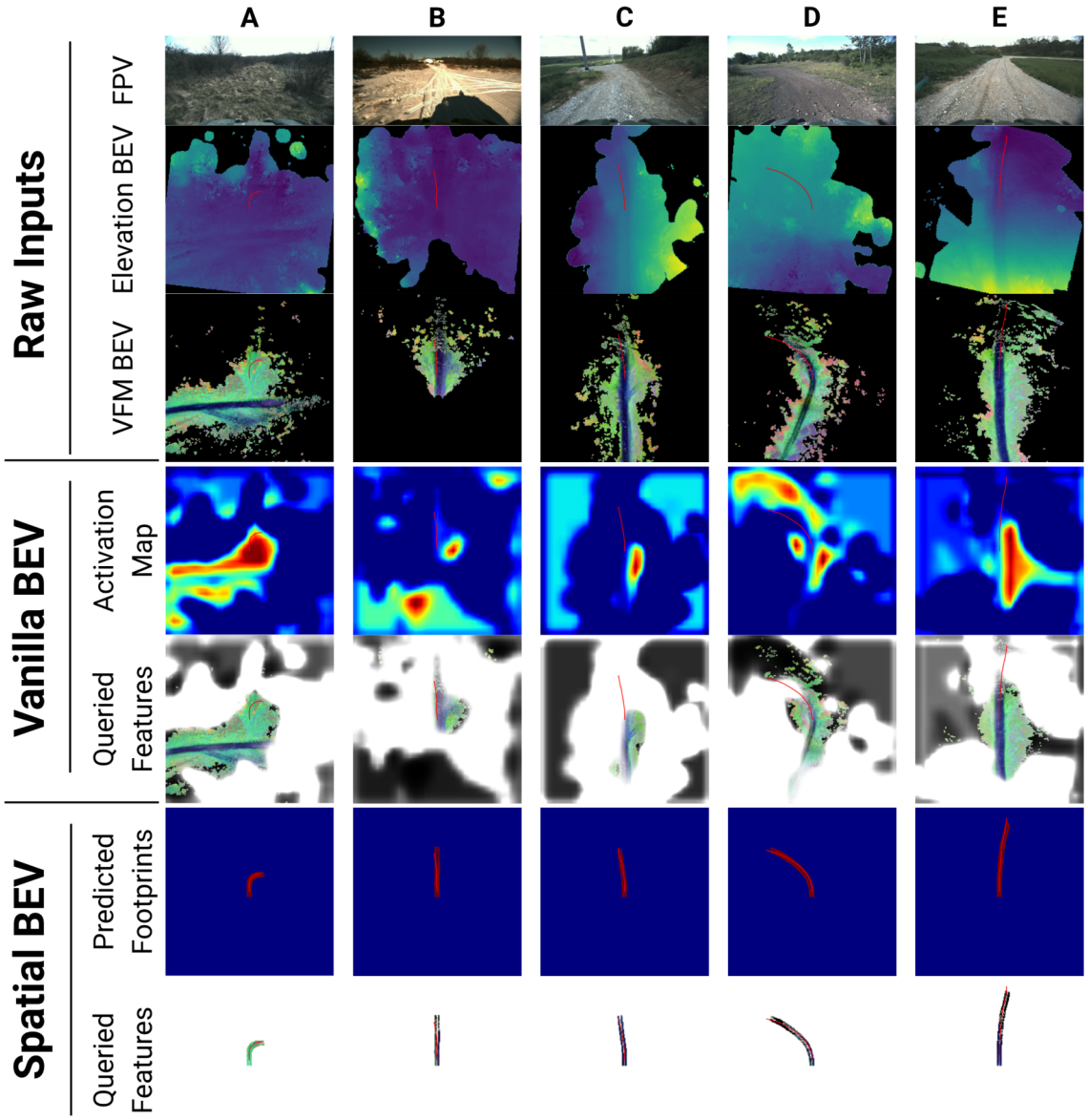


Figure 3.10: Regions of interest for Vanilla BEV and our Spatial BEV approach on five different samples. Vanilla BEV attends to non-sensical regions for predicting motion, whereas our approach queries exactly where it predicts the robot to be, yielding a more informed environment context. The red line is the ground truth trajectory on all BEV plots. (A) through (E) should various environments and motions.

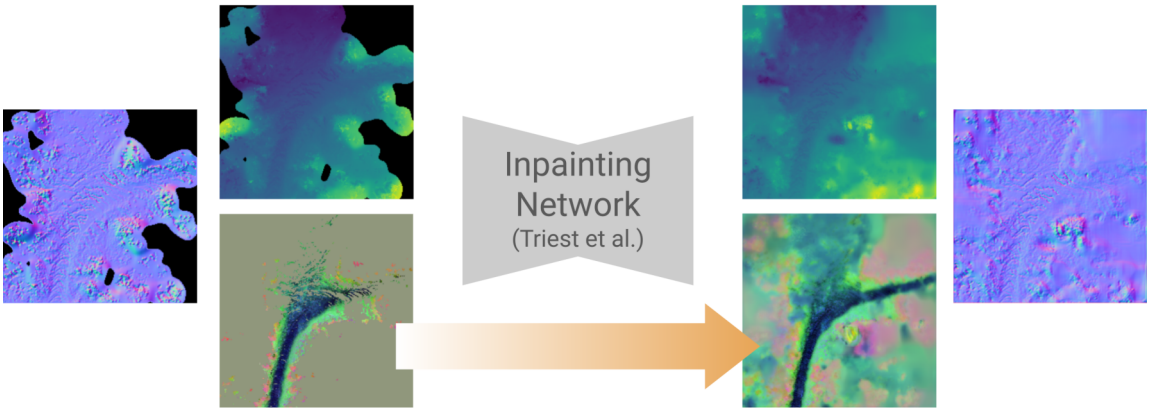


Figure 3.11: The inpainting architecture of Triest et al. [81] creates fully filled BEV maps, predicting trails and shrubbery out of view.

3.5 Improving Terrain-Aware Dynamics Models with Inpainted Features

While BEV maps entertain structure and depth-encoding benefits, they still are a sparse representation, given obstacle occlusions and sensor range limitations, as seen prominently in Figure 3.10, where both the geometric terrain maps and semantic, image-frame-dependent, maps have blank regions. We can use inpainting networks to predict what lies in this unobserved space, otherwise known as inpainting. This work leverages prior inpainting networks from Triest et al. [81] and is out of scope of this thesis. Figure 3.11 shows an example of how we use these inpainting networks to predict unobserved space, where trails are predicted beyond the sensor range.

We motivate the design of our approach by inspecting Vanilla BEV's feature attention maps when given denser, more complete information. Intuitively, denser information should help existing approaches attend to more relevant features; however, Vanilla BEV's activation maps in Figure 3.12 suggest the opposite. Even with inpainted features, its attention appears even more irrelevant to the robot motion. This motivates our approach, which queries only where it predicts the robot to be, allowing it to make full use of these denser, more informed features. We evaluate this benefit quantitatively in Section 5.2.2.

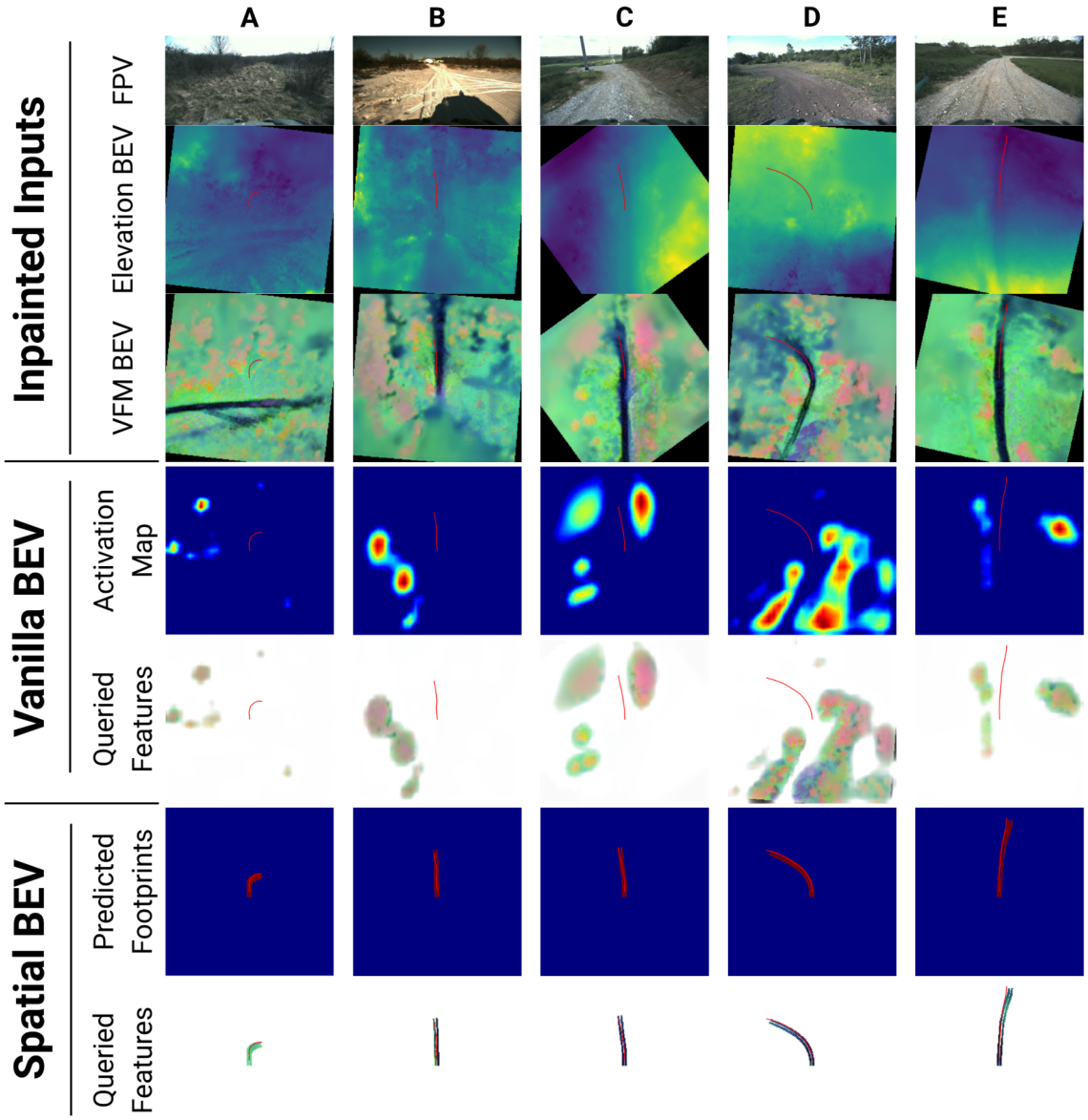


Figure 3.12: When presented with denser, inpainted BEV maps, Vanilla BEV leverages even less relevant features, when compared to the robot's motion. On the contrary, our approach queries only where we predict to be and leverages the denser features completely.

Chapter 4

Rigorous Evaluation

4.1 Introduction

This chapter is driven by research question 2: *How can we rigorously expose the worst-case, long-tail failure modes of off-road dynamics models before real-world deployment?* Dynamics models must be rigorously evaluated before deployment in the field. Aggregate metrics over large datasets are dominated by easy, frequent samples and obscure the rare failure modes that determine real-world safety. In order to rigorously evaluate our approach, we need challenging datasets, which we can create with two parts as seen in Figure 4.1: challenging data (Section 4.2) to collect data across all seasons and motion profiles, and benchmark design (Section 4.3) to mine the most challenging samples of our datasets.

Figure 4.1: Rigorous evaluation requires challenging data and a benchmark design to create challenging datasets.

Figure 4.2: Our dataset provides broad environment and season diversity.

4.2 Challenging Data Collection

We collected 11.4 hr of aggressive driving data across multiple test sites, in all seasons, with speeds up to 13.67 m/s, and slopes as steep as 19.10° (35% grade[1]). It includes traversal through 6-9" of snow, over slick ice, through muddy puddles, and across flowing streams. This dataset serves as an extension of TartanDrive 1 and TartanDrive 2 [75, 79], and our 11.4 hr of driving data with reported statistics of in Table 4.1 and analysis do not include these two prior works. While TartanDrive 1 published the initial dataset and TartanDrive 2 expanded its sensor modalities, our dataset expands environment diversity with all seasons and more test sites shown in Figure 4.2, and we prioritized collecting more unique robot-terrain interactions. We detail the collection platform and environments below.

[1]For context, the American Association of State Highway and Transportation Officials permits a max of 6% in mountainous regions [2]

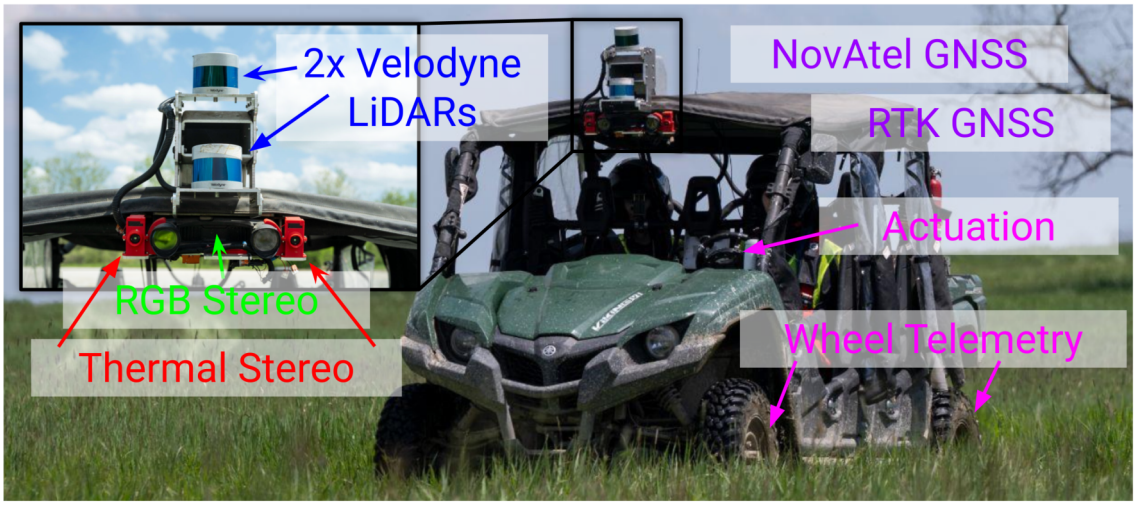


Figure 4.3: Autonomous ATV platform, retrofitted with LiDAR, RGB stereo, thermal stereo, GNSS with RTK corrections, actuation, and wheel telemetry.

4.2.1 Platform

We collect this aggressive data on a Yamaha Viking ATV, retrofitted with actuators for teleoperation, data collection, and autonomy, originally designed by Mai [52]. We further modified it by adding two LiDARs, a stereo camera, a stereo thermal camera, and an RTK GNSS unit as seen in Figure 4.3.

Actuators

The vehicle's action space consists of a target steering angle and throttle position, with no brake actuation available. Teleoperation actions saturate at control limits, since operators snap the gamepad's low-friction joysticks to their extremes rather than modulate them smoothly, while autonomous actions are relatively smooth. Dynamics models trained on this mixed-mode data must therefore generalize from discrete teleoperation and smooth autonomy commands during training to a smooth-only action distribution during closed-loop autonomous evaluation. Commands are sent to the actuators at 30 Hz.

Additionally, due to the resistance of off-road terrain and limitations of the steering DC motor, there is steering actuation delay, with $\sim$5 s elapsing from commanding max steer to reaching max steer, detailed in Mai [52]. This delay is particularly prominent

when there is significant rolling resistance on the terrain surface, like when driving through thick snow or mud. Throttle dynamics are standard, as for any vehicle. Any dynamics model for this platform needs to learn these actuation dynamics implicitly to have accurate predictions.

Sensors

The robot is equipped with exteroceptive, external environment-perceiving sensors:

LiDAR : Two Velodyne VLP-32 LiDARs, which have 100 m of range, 360° horizontal FOV, and ±15° vertical FOV, excluding robot self-hits. One is mounted horizontally, and the other is mounted with a 10° downward tilt. Each is streamed at 10 Hz.

Stereo RGB : One Carnegie Robotics Multisense S21 stereo pair with image resolution 1024×544, 4.8 mm focal length, and 91° horizontal FOV. Only one camera provides color, but both provide grayscale images. The camera is streamed at 10 Hz.

Stereo Thermal : An in-house stereo thermal system, made of two FLIR BOSON+ 640 cameras with resolution 640×512, 4.9 mm focal length, 95° horizontal FOV. We used the industrial-grade variant which has a sensitivity ≤ 20 mK. More details can be found in Liu [49].

The robot is also equipped with many proprioceptive, internal, sensors:

IMU & GNSS : One NovAtel PROPAK-V3-RT2i providing IMU at 100 Hz and non-RTK GNSS pose at 50 Hz. The LiDARs and Stereo RGB additionally come with their own IMU.

RTK GNSS : One MicroStrain 3DM-GQ7 GNSS with RTK corrections for centimeter-level precision GPS poses. This is streamed at 25 Hz.

Wheel Telemetry : One Racepak G2X Pro Data Logger which logs wheel RPM for all four wheels and rear-wheel suspension travel measurements, both streamed at 50 Hz.

We additionally run SuperOdoemtry, a LiDAR-Inertial Odometry solution [96], at 100 Hz as an additional state estimation source and when running autonomy.

	Mean	Std	Min	5%	Median	95%	Max
Forward vel. (m/s)	4.38	2.04	-0.03	1.23	4.12	8.56	13.67
Lateral vel. (m/s)	0.03	0.28	-1.61	-0.39	0.02	0.47	3.89
Slip (deg)	0.39	5.15	-49.58	-7.46	0.23	8.99	51.23
Yaw Rate (rad/s)	-0.00	0.15	-0.89	-0.25	-0.00	0.26	0.83
Pitch (deg)	-0.12	3.87	-19.24	-7.29	0.04	6.39	21.11
Roll (deg)	0.03	3.60	-18.88	-6.09	0.21	5.33	24.97

Table 4.1: Dataset statistics

4.2.2 Dataset

Collection Site Diversity

We collect our data at two sites: Gascola and Renegade Ridge. Gascola is a ∼285-acre site in Western Pennsylvania, U.S., featuring narrow dirt and gravel trails, dense foliage and rocky terrain, with slopes and ruts. Renegade Ridge is a ∼675-acre site in Eastern Ohio, U.S., that is more geometrically complex than Gascola, with vast open fields containing deep ruts as well as extreme slopes and embankments. Figure 4.4 highlights how Renegade Ridge has a broader pitch and roll distributions, with higher peak slopes, and more challenging geometries.

We collected coverage of Gascola in all four seasons and observed the environment evolve due changing climate, overgrowth, geological erosion, fallen trees, ephemeral streams, and more. Figure 4.5 shows how a single location in Gascola changes in the four different seasons. Winter coverage includes the most dynamically interesting data with low-friction icy stretches and thick 6-9" snow with significant rolling resistance. Additionally, the cloudy lighting conditions and barren trees render the wintry images more featureless.

4.3 Benchmark Design

With the challenging dataset $\mathcal{D}$ created, we need to mine the most challenging samples from this dataset. Much of the extreme motion or interesting terradynamics present in this challenging dataset occur for a short period time with benign interactions around it. When evaluating with aggregate metrics, these benign occurrences overwhelm the

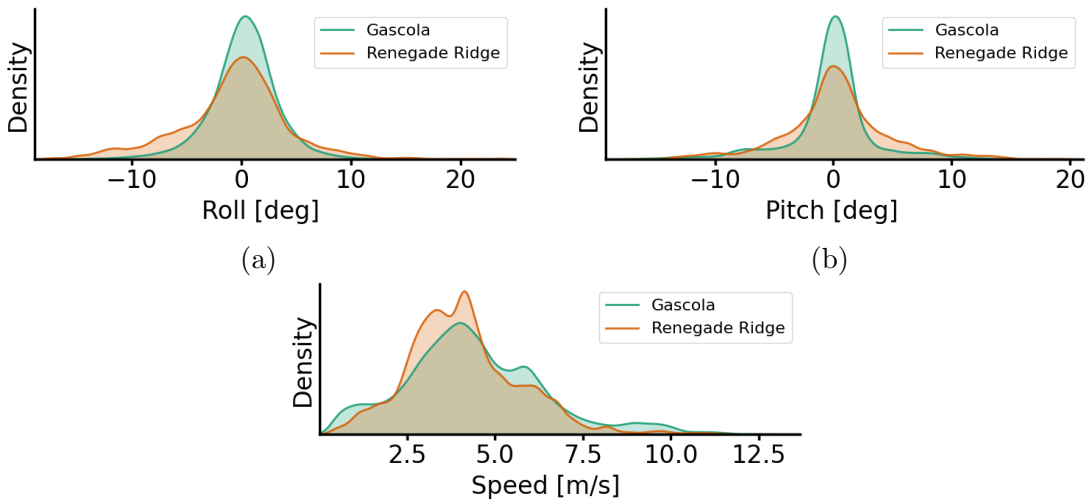


Figure 4.4: Renegade Ridge offers more geometric complexity, compared to Gascola, with a higher roll and pitch variance and steeper slopes with higher max pitch but lower top speeds.

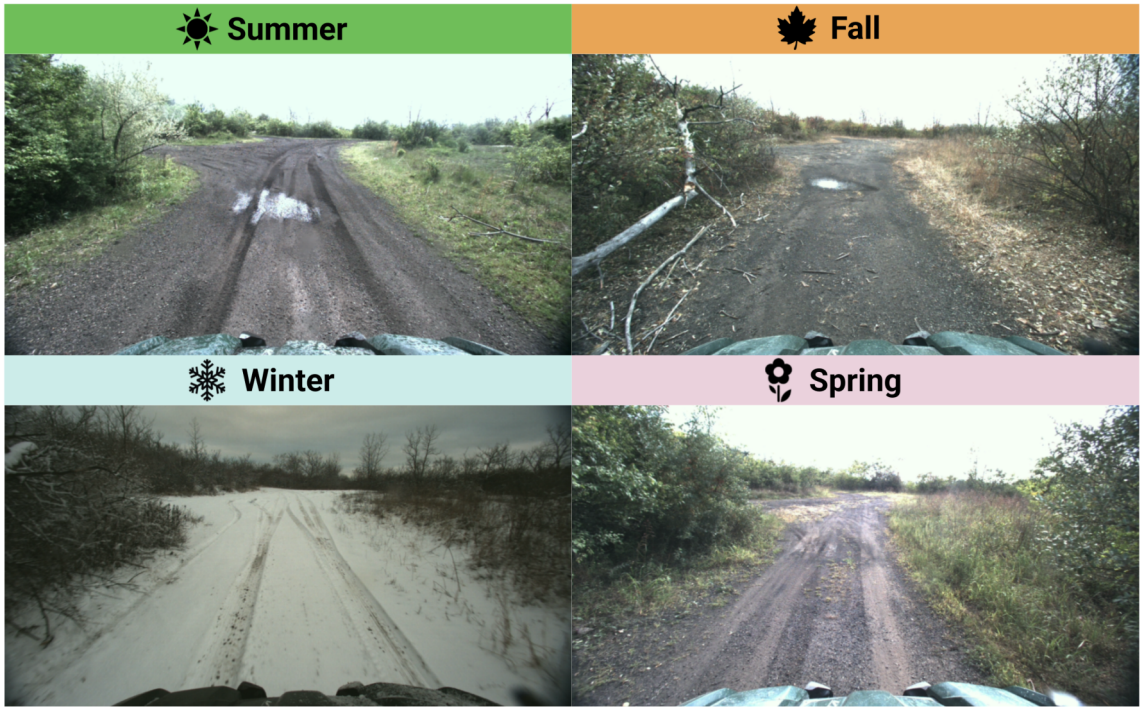


Figure 4.5: Our dataset features environments encountered in all seasons. The same location in different seasons highlights the stark differences in visual and geometric appearance, as well as the terrain surface.

most challenging samples and mask potential failure cases in these long-tail scenarios. We introduce a systematic method for identifying the most challenging samples in a dataset, enabling targeted evaluation on the state sequences models are most likely to mispredict. Our findings highlight the necessity for perception representations, revealing the impact of terrain geometry and semantics on prediction quality.

4.3.1 Data Mining Intuition

To capture the most challenging samples for a benchmark, we present an approach to group similar action sequences which yield different forward dynamics. Intuitively, identical actions executed in identical environments will yield identical trajectories. In reality however, the environments are rarely identical, which motivates learning environment terradynamics to improve model accuracy. Terradynamics are most challenging to model for off-road driving [30, 67, 73], so our goal is to formulate datasets that maximize the trajectory variance under similar actions and initial conditions, attempting to highlight where terradynamics are most prevalent.

4.3.2 Clustering Approach

Similarity between two candidate samples is defined by comparing their action sequences and initial conditions. For each pair of candidates, the $\mathcal{L}_2$ distance is computed between their normalized action sequences and initial conditions, and connect pairs whose distance falls within a threshold. The resulting similarity graph is partitioned into cliques, which are subsets of samples that are pairwise within the threshold of one another; these form our groups. The threshold is selected by sweeping a range of values and choosing the one that maximizes the mean silhouette coefficient [69], which is high when candidates lie closer to their own group than to the nearest competing group. 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 difficulty metric (4.1). We use this as a way to capture trajectory variance. Figure 4.6 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 snow.

$$\bar{d}_H(\mathcal{G}) = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j=1}^{N} d_H\left(\tau^i, \tau^j\right), \quad j \neq i \tag{4.1}$$

4.3.3 Benchmark Metric

The clustering procedure yields a set of groups $\{\mathcal{G}_1, \ldots, \mathcal{G}_K\}$, each scored by its trajectory variance $\bar{d}_H(\mathcal{G}_k)$ (4.1). Sorting the groups by this score and splitting them into ten equal parts partitions the dataset into disjoint difficulty deciles $\mathcal{D}_1, \ldots, \mathcal{D}_{10}$, where $\mathcal{D}_1$ holds the highest-variance groups and $\mathcal{D}_{10}$ the lowest. $\mathcal{D}_1$ herefore contains the regimes where similar commanded actions produced the most divergent motion, and where terradynamics dominate the prediction. The mined subset used for evaluation is the union of the hardest n deciles,

$$\mathcal{D}_{\text{hard}} = \bigcup_{j=1}^{n} \mathcal{D}_j \subset \mathcal{D}, \tag{4.2}$$

where n controls how aggressively the benchmark targets the long tail. To put this metric all together, model error is commonly reported in expectation over the full dataset,

$$\text{err}_{\mathcal{D}} = \mathbb{E}_{\tau \sim \mathcal{D}} \left[\text{err}(\hat{\tau}, \tau)\right], \tag{4.3}$$

which our benchmark replaces with the same error over the mined subset,

$$\text{err}_{\text{hard}} = \mathbb{E}_{\tau \sim \mathcal{D}_{\text{hard}}} \left[\text{err}(\hat{\tau}, \tau)\right]. \tag{4.4}$$

The error metric is unchanged; only the dataset distribution it is taken over differs. 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 benign performance. Reporting error per decile, as in Figure 5.5, shows this degradation directly.

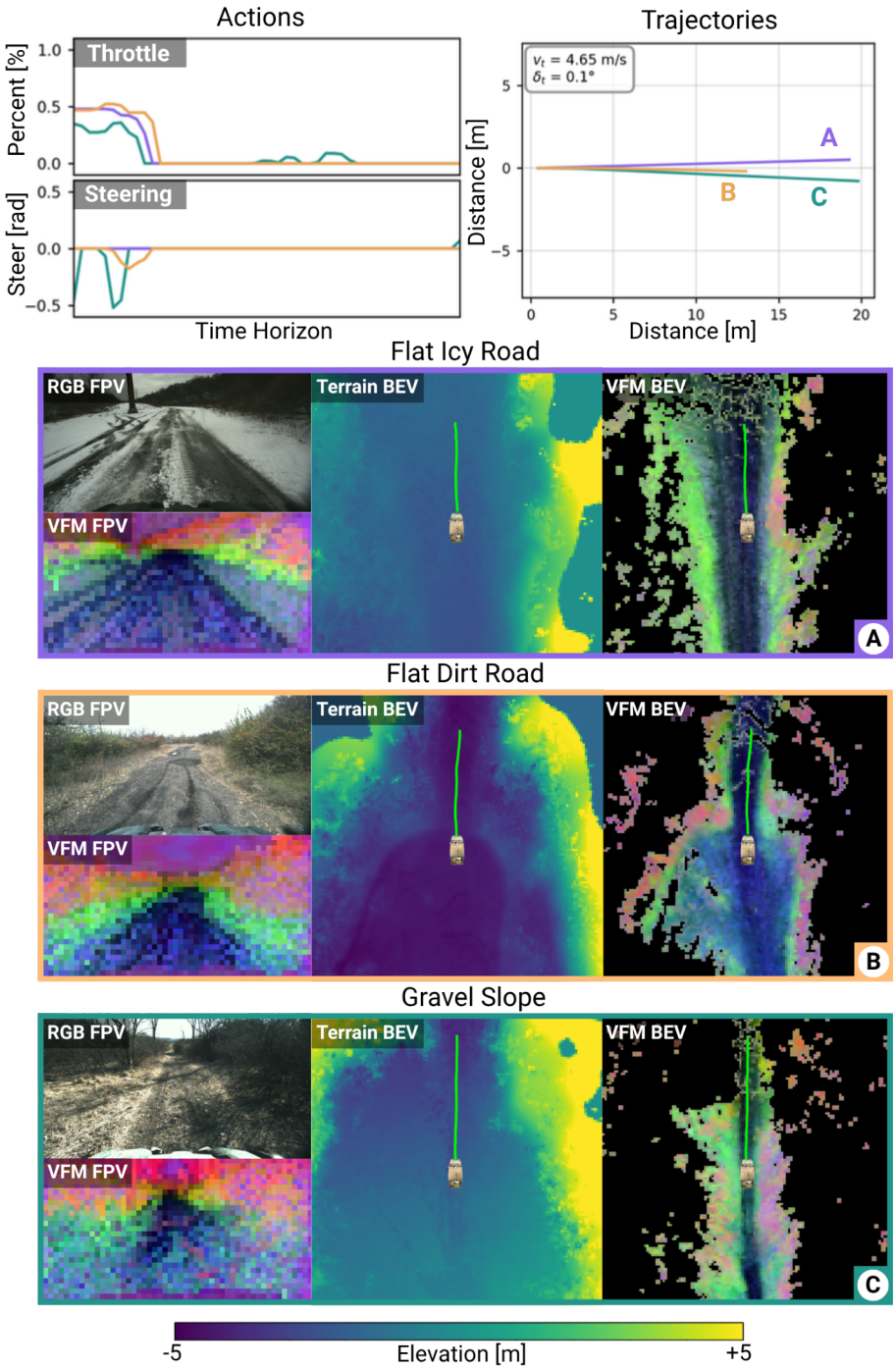


Figure 4.6: Similarly clustered actions yield different trajectories. Perception highlights how terradynamics are causing this difference. (A) travels further, going down a hill, as seen by the terrain map, (B) moves the least on flat dirt, while (C) travels further on flat, low-friction ice.

Chapter 5

Experiments

We next evaluate our terrain-aware dynamics modeling method and the efficacy of our benchmarking design method. We guide our evaluation by the key research questions this thesis seeks to answer:

Research Question 1: How can dynamics models fully leverage the spatial structure of representations like BEV maps to achieve terrain-awareness and reduce prediction error across diverse and complex terrains?

Research Question 2: How can we rigorously expose the worst-case, long-tail failure modes of off-road dynamics models before real-world deployment?

Research Question 3: At what point does improved dynamics-model accuracy stop meaningfully improving closed-loop control performance?

We address **RQ1** in Section 5.2 through offline, open-loop evaluations that measure predictive accuracy alone, with no feedback and no consequences to our actions. We address **RQ2** in Section 5.3 by using our benchmark to sort our dataset into challenging and non-challenging subsets and comparing how dynamics models' predictive accuracy differs across them. We address **RQ3** in Section 5.4 by deploying our method and baselines on hardware across multiple off-road courses, evaluating each dynamics model's effect on online, closed-loop control, where actions do have consequences. Additionally, Section 5.1 details shared setup, such as evaluation baselines, model training data, and model implementation, used across these experiments.

5.1 Shared Setup

5.1.1 Baselines

We evaluate our approach against 4 baselines, spanning physics-based and learned methods:

1. **KBM**: A purely physics-based single-track kinematic bicycle model with gravtiy and actuator modeling as in Section 2.4.1. Its parameters were fit using least squares on ground-truth trajectories. This approach has been tested extensively throughout many years and used for other work evaluations [49, 51, 76, 80].
2. **State**: An LSTM trained with state proprioception only while using state-action history context. This is a faithful recreation of Gibson et al. [19] in its non-hybrid form.
3. **FPV**: An FPV-conditioned LSTM, in which the a VFM image is encoded once at the start of propagation and combined with the state-history context as shown in the baseline input of Figure 3.4.
4. **Vanilla BEV**: A BEV-conditioned LSTM, in which a BEV map is encoded once at the start of propagation and combined with the state-history context as shown in the baseline input of Figure 3.4. This is a faithful recreation of the dynamics model from Roth et al. [68].

5.1.2 Training Dataset

The approach is trained on 79.8k samples (2.2h) of autonomous and teleoperated navigation data around Gascola. Importantly, this dataset includes data collected in the spring, fall, and summer. We choose to leave snow held-out to show if our perceptually conditioned models can perform well in out-of-distribution (OOD) visual cases. We train using state estimates from the LiDAR-Inertial Odometry solution [96], but our models use RTK during real-world deployment[1]. This does highlight performance robustness to state estimation solution.

[1]RTK was not available throughout the collection of this dataset, hence the mismatch.

5.1.3 Model Implementation

Our approach uses a Long Short-Term Memory (LSTM) [29], trained to predict 50 states forward with a horizon of 5 s and a Δt of 0.1 s. Alongside the initial state and action sequence of $T-1$ actions, this prediction depends on 5 s of prior historical state-action pairs and a BEV map at 100 m $\times$ 100 m at 40 cm resolution with geometric and visual semantic features, with 1 elevation channel, 3 surface normal channels, and 32 VFM feature channels.

We train to minimize the Mean Squared Error (MSE) loss using the Adam optimizer [40]. Our state reflects an extended kinematic state as explained in Section 2.4.1 and is comprised of pose in SE(2), speed, current steer, and pitch: $\mathbf{x} = \begin{bmatrix} x & y & \psi & v & \delta & \phi \end{bmatrix}$. We choose to predict the same state to be capable of full-state evaluations. More implementation details in Section A.3.

5.2 Offline Experiments

Our objective of **RQ1** is to determine if terrain-aware spatial perception representations and leveraging the spatial structure of the problem improve dynamics predictions. We evaluate our method and baselines on manually-defined challenging datasets quantitatively and qualitatively and discuss interesting insights. We will evaluate two prediction metrics: the normalized full-state prediction, which is important for the control cost functions and constraints, and the unnormalized position (X-Y) prediction, which shows how "close" a trajectory is in metric space and is also relevant for MPC costmap evaluation. We evaluate these metrics on average and worst-case (95^{th} percentile) as a crude way to deter conclusions drawn from aggregate metrics alone.

5.2.1 Test Datasets

We present all offline results using a test held-out dataset of 75.3k samples (2.1h) of autonomous and teleoperated navigation around Gascola and Renegade Ridge. Renegade Ridge 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

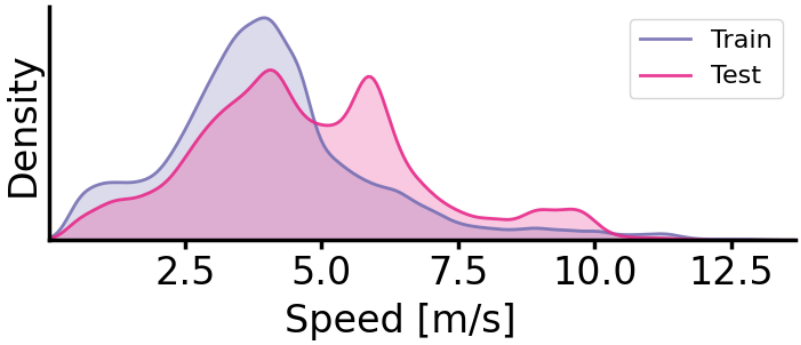


Figure 5.1: Comparing test and train dataset speed distributions, the test dataset has a higher density of higher speed.

additional tire forces and reduced friction outside the training dataset. Figure 5.1 shows this test dataset having a larger density of higher speed data. Alongside our Full test dataset, we break it up into Snow, Slopes, and Speed sub-datasets for evaluation and additional insights on the strengths and weaknesses of the approaches. These sub-datasets are defined in the following ways:

1. **Snow**: A subset of the dataset, which includes snow, manually defined from run log metadata. Having snow was defined as having ∼2" of snow covering the ground.
2. **Slopes**: A subset of the dataset, where for a given trajectory τ, the maximum pitch experienced is at at least 0.1 radians (5.8° or 10% grade[2]).
3. **Speed**: A subset of the dataset, where 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 (Section 4.2.1).

5.2.2 Quantitative Evaluation

On all datasets, our approach outperforms all methods on average, including its Vanilla BEV counterpart, when predicting the full state space as seen in Figure 5.2.

[2]The American Association of State Highway and Transportation Officials permits a max of 6% in mountainous regions [2]

Interestingly, we have 17.6% lower average prediction error than Vanilla BEV on the high-speed dataset, supporting **RQ1**. This suggests that at high speed, since the robot travels much further per horizon, the flattened representations struggle to attend to features outside its current locality. Our structured representation, by contrast, can query the most relevant features regardless of distance from the initial ego position. Our hypothesis is reinforced by our introspective activation map analysis on the Vanilla BEV approach in Section 3.4.3, where frequently, the model's "attention" is on random sections of the input, while ours concentrates on where the robot is expected to be, even at high speeds.

We see that in general, perceptually-conditioned approaches do perform better on the Snow, Slopes, and Speed datasets, highlighting how terrain-awareness is a necessary component of dynamics models for challenging terrains. Additionally, while on the OOD Snow dataset, our method performs equivalently to the state-only approach. Figure 5.2 shows our method performing best of all perceptually-conditioned methods on the OOD Snow dataset by 17.3%. This indicates that the less spatially aligned metrics are querying arbitrary features which are explicitly harmful for the prediction.

When evaluating only position predictions in Table 5.1, we outperform Vanilla BEV on all metrics; however, the FPV method still outperforms our approach on Snow and Slopes datasets when only evaluating the position prediction. Most notably in support of **RQ1**, Table 5.1 shows the worst-case position prediction is 23.8% lower than its Vanilla BEV counterpart and 61.5% lower than FPV on the high-speed dataset. While our method does not perform best on position predictions across all datasets, predicting position alone is not fit for many MPC formulations. Robot yaw can be an important state field for orientation-derived costs such as waypoint following, and velocity is another important field for velocity constraints, as we will see in Section 5.4.

These results show that quantatively, while perception improves terrain-awareness and thus prediction accuracy, our spatial approach improves prediction accuracy the most on average and in worst case for most of the selected challenging datasets with respect to the Vanilla BEV approach; however, the FPV baseline still outperforms our approach in many cases. We address this in Section 5.2.2

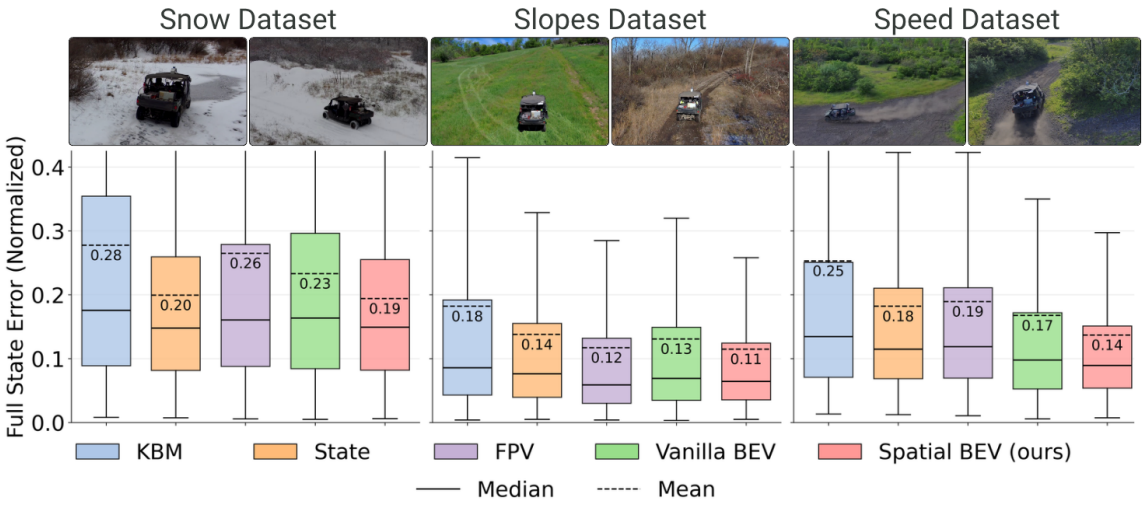


Figure 5.2: Offline full state prediction error distributions across 3 datasets. The snow dataset proves most challenging for all methods with a high average prediction error. The speed dataset yields the most variance with the KBM performing poorly. Across all datasets, our method outperforms the baselines.

	Snow		Slopes		Speed		Full	
Method	MSE [m]	95%	MSE [m]	95%	MSE [m]	95%	MSE [m]	95%
KBM	2.68 ± 4.23	8.92	2.26 ± 4.64	7.75	3.00 ± 7.17	11.16	2.08 ± 4.20	7.13
State	2.54 ± 2.52	7.62	2.16 ± 3.11	7.31	1.83 ± 3.62	7.27	1.97 ± 2.73	6.65
FPV	**2.34 ± 2.37**	**6.93**	**1.65 ± 2.91**	**5.74**	2.14 ± 4.60	7.98	1.74 ± 2.80	**5.95**
Vanilla BEV	2.49 ± 2.89	8.12	1.94 ± 3.34	7.06	1.55 ± 3.62	6.12	1.71 ± 2.85	6.62
Spatial BEV	2.45 ± 2.66	8.06	1.77 ± 2.54	6.91	**1.46 ± 4.10**	**4.94**	**1.67 ± 2.67**	6.21

Table 5.1: Offline position prediction results across 3 datasets, alongside the full test set.

Evaluating Models with Inpainted BEV Maps

As shown in Section 3.5, inpainted maps yield complete feature in BEV maps beyond the sensing range. We evaluate if this denser information improves prediction accuracy for Vanilla BEV and our approach. Figure 5.3 and Table 5.2 shows our approach with inpainting reduces our approach without inpainting's average prediction error by at least $\geq 18.1\%$, across all datasets, with as high as 40.6% improvement on the snow dataset. This supports **RQ1** and demonstrates, how when given a denser input,

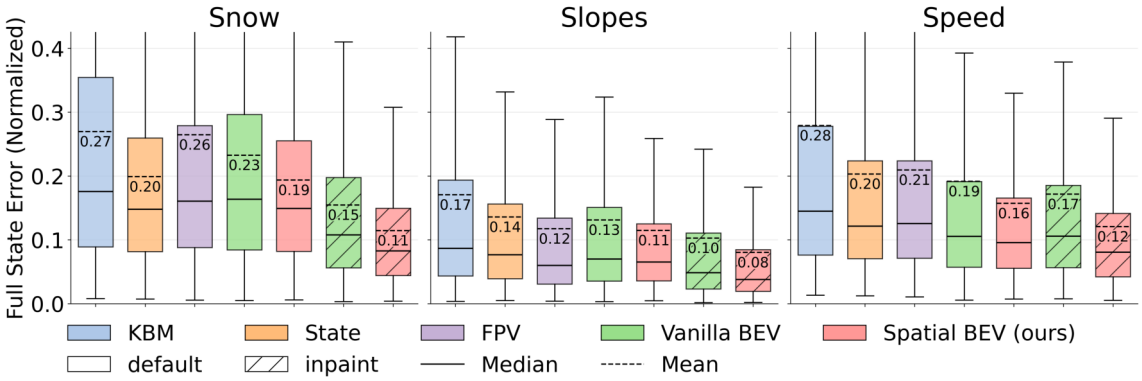


Figure 5.3: When using inpainted BEV inputs, our approach improves significantly, outperforming all baselines on all datasets on average. Inpainted BEV inputs improve our approach on all datasets on average.

	Snow		Slopes		Speed		Full	
Method	Mean	95%	Mean	95%	Mean	95%	Mean	95%
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	5.82	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
Spatial BEV	2.45 ± 2.66	8.06	1.78 ± 2.51	6.87	1.50 ± 3.90	5.42	1.70 ± 2.68	6.26
Vanilla BEV	2.10 ± 2.44	6.76	1.97 ± 3.28	7.70	1.41 ± 2.93	5.17	1.46 ± 2.50	5.63
Spatial BEV	**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 5.2: When given inpainted BEV maps, indicated by red text, our approach has lowest average and worst-case prediction error across all datasets.

our structured perception representation takes full advantage of this structure. When comparing our approach to all baselines, excluding inpainted Vanilla BEV and our default approach, our inpainted approach reduces prediction error by at least 34.0% on Snow, 12.2% on Slopes, 40.8% on Speed, and 42.9% on Full datasets.

Interestingly, despite Vanilla BEV having poor ability to query the most relevant features to a robot's motion, as shown in Section 3.5, Vanilla BEV does also reduce its average prediction error by as high as 16.9% compared to its non-inpainted counterpart. In most cases, Vanilla BEV beats our non-inpainted approach as well, suggesting how important complete feature maps are for environment contextualization and learning.

Unfortunately, this breakthrough was discovered after field tests were complete, so the remainder of these experimental results will not include the inpainted variants

of the BEV networks.

5.2.3 Qualitative Evaluation

While quantitatively our approach performs well, we can gain additional insights from qualitative rollout comparisons. We qualitatively evaluate the success and failure cases of our approach by analyzing the prediction results in Figure 5.4. 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 terrain 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. We see terrain impact significantly in (D) where the robot is doing a slalom up a hill at very low speeds (2.4m/s). Despite this low speed, the predictions are still varied significantly laterally, but of all percepetive terrain-aware methods, ours is most accurate.

The most difficult case arises in (J), where the robot is command slight left steering but 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 because this rolling resistance from ice breaking is not accounted for, and all predict trajectories to the left, given the left steer, yet the actual path is to the right, since the rut constrains the robot's motion to the right. No method is able to predict this complex phenomenon, regardless of representation.

From these various examples across all seasons and motion profiles, we can see our method frequently has the best prediction, qualitatively, yet there are many extreme long-tail data that causes failure for our method and all methods.

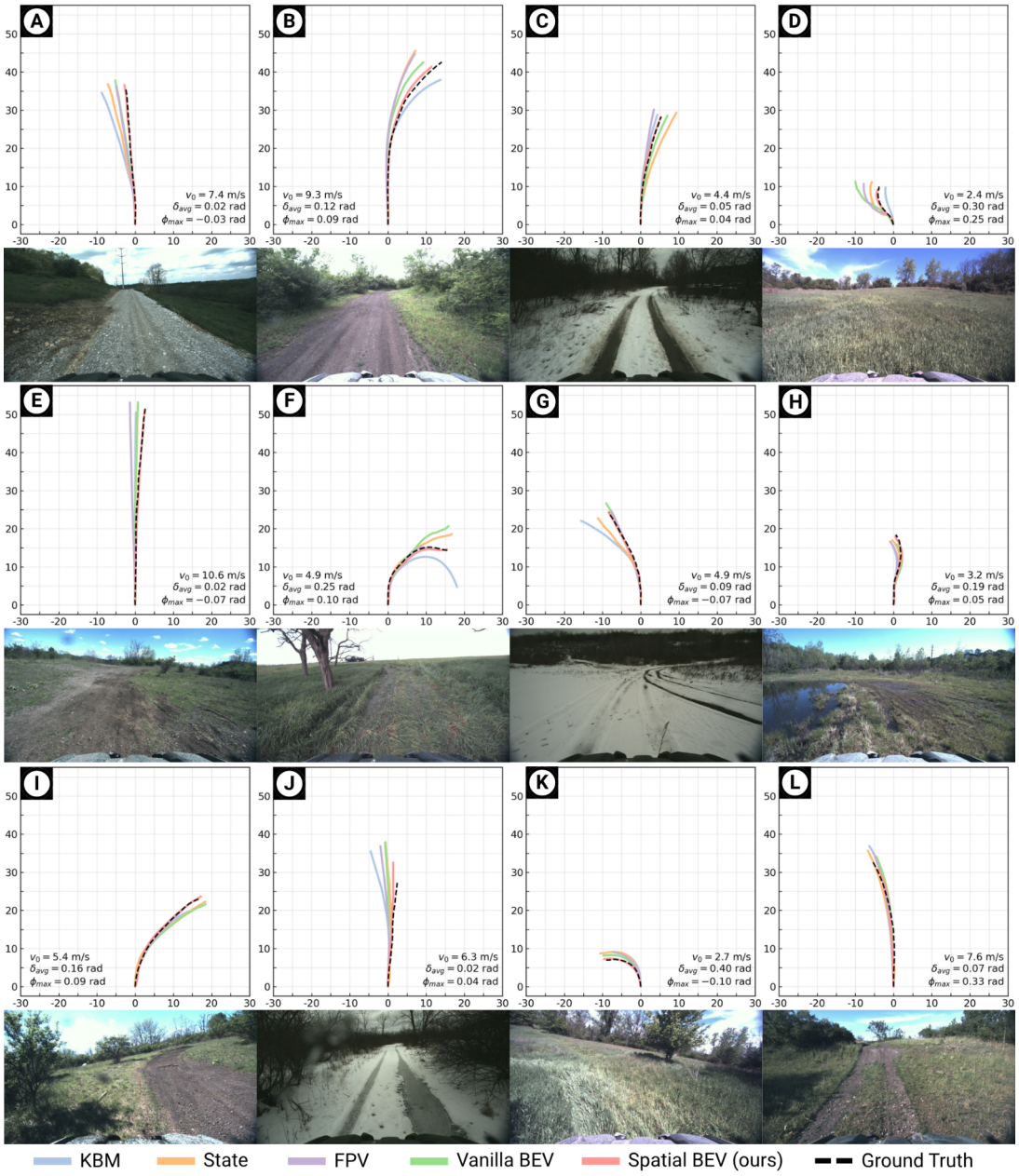


Figure 5.4: Offline qualitative results highlighting interesting prediction examples and environments, including high-speed turns and deep snow traversal. Provided are the rollout predictions for all methods, an FPV image showing the environment, and scalar statistics to characterize the motion profile of the run, including inital velocity, average steer, and max pitch.

5.3 Benchmark Efficacy

We want to validate our benchmark 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**.

We follow our procedure from Section 4.3.3 to compute the highest trajectory variance deciles $\mathcal{D}_1$ through to the lowest variance deciles $\mathcal{D}_{10}$. We evaluate our method and baselines on each decile in Figure 5.5 and find that all models' worst-case prediction error is highest on the highest variance decile and lowest on the lowest variance decile. In fact, all methods have strong downwards trends going from lowest variance decile to highest variance decile. This demonstrates how our benchmark can identify the highest variance runs and how this correlates to dynamics modeling difficulty.

We also see our approach performing best across nearly all deciles, except for the 6^{th} decile, showing our methods ability to predict accurate motion across challenging and easy data samples. Moreover, the State and FPV methods have 109% higher worst-case error than our approach on the most challenging bin. This suggests the models' inability to reason about terradynamics, since these challenging bins identify where terrain is most relevant as shown in Figure 4.6. This emphasizes the necessity of evaluating on challenging data before deployment, else data similar to this may be experienced and the dynamics model will output poor or nonsensical predictions, leading to failure.

5.4 Online Evaluations

Finally, we close the loop and evaluate our approach and baselines on hardware with control in the loop. We evaluate **RQ3** to see if our better performing Spatial BEV approach shows similar gains when evaluated with control in the loop. We answer this by evaluating our models on a line-following task using only control.

5.4.1 Line-following Control

To properly evaluate only the dynamics model and its affect on control, we remove all confounding factors such as state estimation, traversability estimation, or planning.

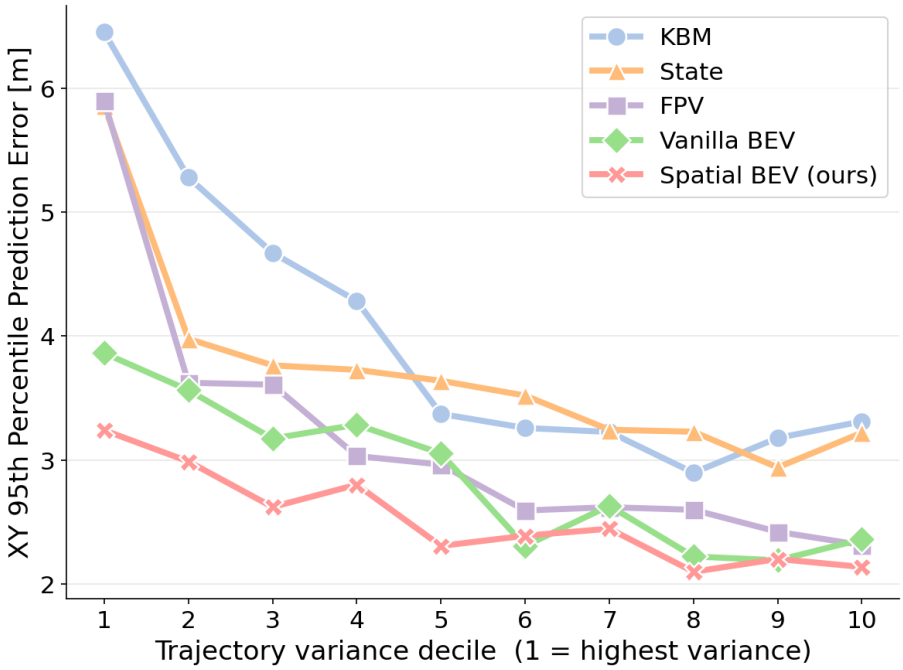


Figure 5.5: Evaluation on the highest to lowest variance deciles of the clustered benchmark. Model performance increases with lower variance deciles.

We set up a line-tracking task, where our controller is tasked to follow a line using RTK GPS odometry. We measure success, we evaluate cross-track-error (CTE) of all methods and number of operator interventions as a light heuristic. In addition to line following, our controller also has lateral acceleration and speed constraints. We evaluate on two courses at Gascola:

1. **High-Speed Course**: A 739m-long course over benign terrain, including flat gravel, dirt, and a few divots, with target speed of 10m/s. The course has wide and sharp turns with long straights.
2. **Terrain Geometry Course**: A 524m-long course over challenging terrain, including steep slopes and banked turns, with target speed of 8m/s. The course has wide and sharp turns.

Task Setup

We use MPPI [89, 90] to track the reference lines. We roll out 825 unique action sequences and execute MPPI at 10 Hz; however, a lower-level controller executes the

action sequence given by MPPI at 30 Hz. This low rate of execution helps highlight the importance of accurate dynamics models because the robot will execute more of the commands before its next solution is computed. The robot uses its RTK GNSS for centimeter-level state estimation, and perception maps are created from an RTK-registered pointcloud. Our controller uses a combination of path-tracking costs and physics-derived safety limits, such as lateral acceleration and speed limits (5.1) with more details on implementation and our cost function in Section A.2.1.

$$\ell(\tau) = \ell_{\text{speed}}(\tau, v_{\max}) + \ell_{a_y}(\tau, a_{\max}) + K_{\text{track}}\ell_{\text{track}}(\tau, \tau_{\text{ref}}) + K_{\text{progress}}\ell_{\text{progress}}(\tau, \tau_{\text{ref}}) \tag{5.1}$$

Does our model improve line-following at high speed?

As seen in Figure 5.6, all models track straight lines well at speed. This makes sense because even with inaccurate throttle modeling, the robot's trajectory remains close to the line as long as heading is roughly correct. Turns at speed are more challenging; inaccurate dynamics modeling sends the robot off the reference trajectory, and compensation is required to recover. Turns therefore prove to be the sections where modeling error is most pronounced. The KBM baseline consistently understeers through both low- and high-curvature turns, resulting in 3 interventions. The learned baselines and our approach track turns considerably better, with occasional oversteer. We do incur one intervention in case (D), but this stems from an MPPI cost function issue rather than a dynamics modeling error. In this case, the optimal trajectory determined by MPPI undercut the turn. As the robot executed these actions, the solution continued to undercut the turn, until the operator felt the need to intervene because the robot was approaching dense vegetation. It did not "see" this vegetation, since there was no costmap used during this evaluation and only line following; see Section A.2.2 for details.

The metrics in Table 5.3 show a similar pattern. Our approach has higher average CTE than Vanilla BEV, but lower worst-case CTE. While we have higher average CTE, our approach's max velocity is closest to the 10 m/s target, while the baselines settle near 11 m/s, suggesting the baselines are less accurate at predicting velocity at

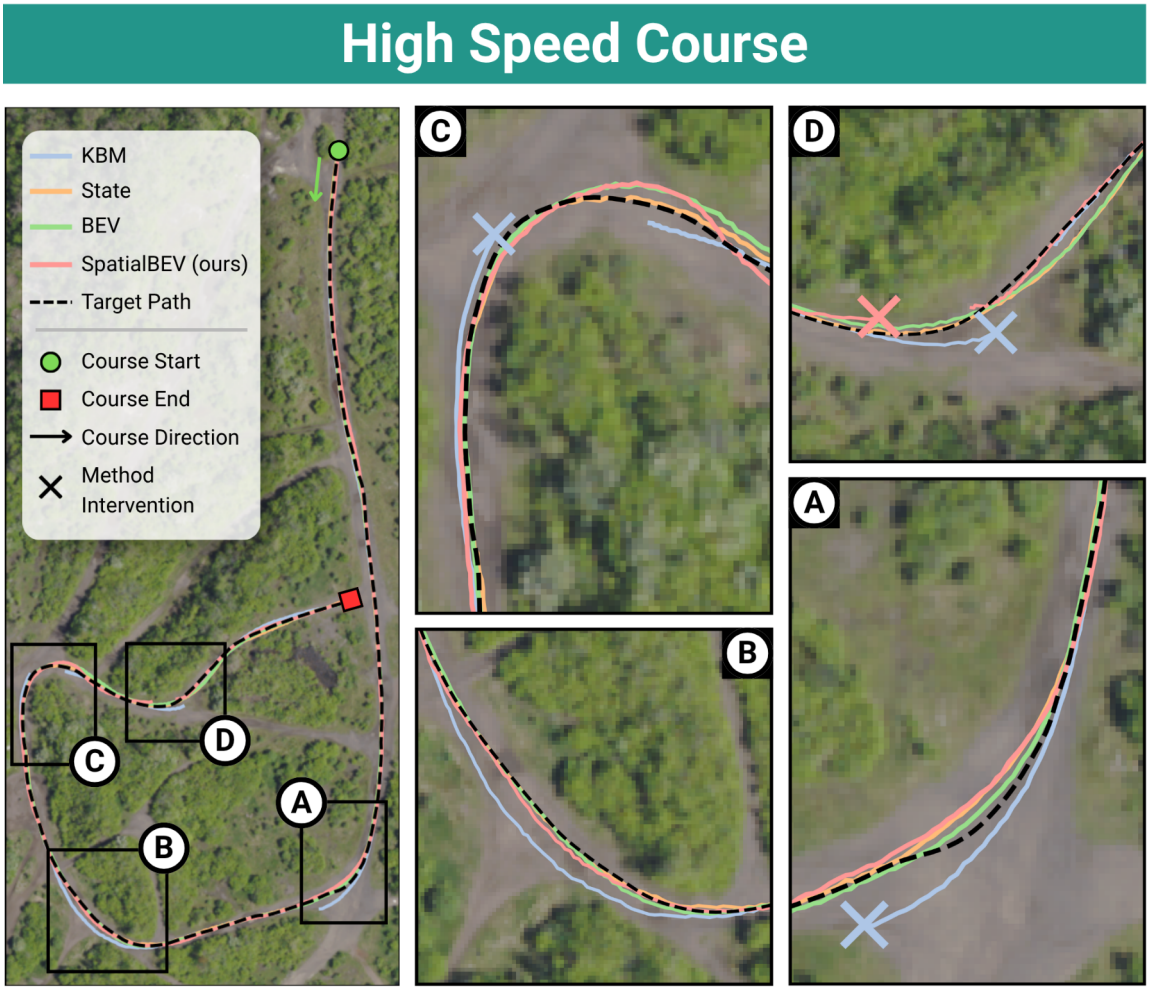


Figure 5.6: High-speed hardware experiments. Significant understeer can be seen by the KBM Baseline leading to numerous interventions (A,D). Additionally all learned methods, including ours, experience cases of understeer like in the tight turn in C.

Method	# Trials	# Intv	Avg # Intv	vel. avg	vel. max	CTE avg	CTE max
KBM	1	3	3.00	7.75	11.20	0.96	6.35
State	1	**0**	**0.00**	8.39	10.96	**0.55**	3.11
FPV	1	**0**	**0.00**	7.85	10.98	0.65	3.30
Vanilla BEV	1	**0**	**0.00**	8.44	10.94	0.67	4.01
Spatial BEV	1	1	1.00	8.02	10.41	0.70	**2.65**

Table 5.3: Hardware deployment results on the High Speed Course.

speed. This supports the importance in predicting the full state correctly, not just the positions.

Does our model improve line-following over challenging terrain?

Similar to the high-speed case, Figure 5.7 shows that while all models track straight lines well, turns remain the segments that expose failure. The KBM continues to understeer through low- and high-curvature turns, requiring two interventions in case (E). The learned baselines also understeer in this case but do not warrant intervention, while our method instead oversteers. In case (G), all baselines exhibit significant understeer, whereas our approach tracks the curve tightly.

Quantitatively, Table 5.4 shows our method achieving the best worst-case performance, though slightly worse average performance, compared to the learned baselines. As in the high-speed evaluation, our method performs comparably to or better than the learned baselines both qualitatively and quantitatively, again suggesting either a ceiling on achievable performance or the need for a more challenging evaluation setting to further differentiate the approaches. We were able to perform this experiment across four trials.

How does prediction accuracy compare when evaluating closed-loop data offline?

We conduct a small experiment to evaluate prediction accuracy on the data collected during the closed-loop experiments, allowing us to compare open-loop and closed-loop performance on the same data. Table 5.5 tells a different story than the closed-loop results: while Table 5.3 shows State having the lowest average and worst-case CTE, the BEV perceptually-conditioned methods achieve the lowest position prediction MSE when evaluated offline. In particular, Vanilla BEV has the lowest average error but with high variance, while our method has the second-lowest average error with lower variance and the lowest worst-case error. The KBM, meanwhile, has substantially higher average and worst-case error than the other methods.

This difference in result ordering from the offline metrics suggests the present, minor entanglement of our current evaluation. It also highlights how, with control in the loop, the best-performing dynamics models may not be guaranteed to yield the

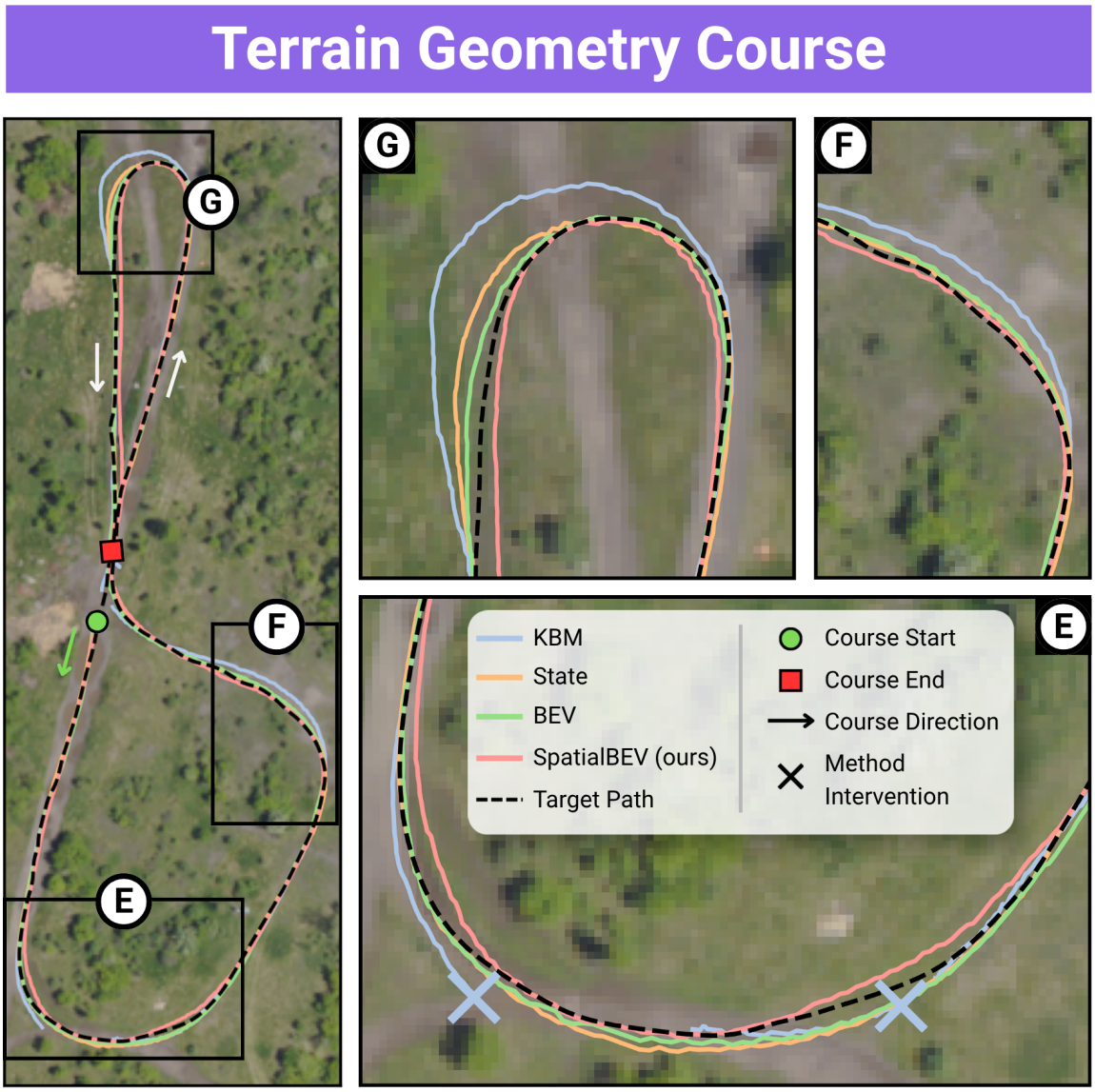


Figure 5.7: Challenging geometry hardware experiments. Significant understeer can be seen by the KBM Baseline leading to numerous interventions (E) and additional understeer which did not lead to intervention (F,G). Baselines struggled to follow a tight turn (G), while our approach could follow it.

Method	# Trials	# Intv	Avg # Intv	vel. avg	vel. max	CTE avg	CTE max
KBM	4	15	3.75	6.13	9.01	0.81	5.51
State	4	**0**	**0.00**	6.80	8.68	0.45	3.35
FPV	4	3	0.75	6.50	8.90	0.65	4.90
Vanilla BEV	4	**0**	**0.00**	6.94	8.60	**0.43**	3.67
Spatial BEV	4	**0**	**0.00**	6.58	8.66	0.60	**2.74**

Table 5.4: Hardware deployment results on the Geometry Course.

Method	MSE Avg [m]	95%
KBM	5.57 ± 7.11	20.06
State	2.76 ± 3.63	11.93
FPV	3.28 ± 3.53	11.58
Vanilla BEV	**2.42 ± 3.60**	8.90
Spatial BEV	2.68 ± 2.66	**7.80**

Table 5.5: Offline position prediction results on closed-loop experiment data

best control, given inference time, cost function construction, noisy measurements, and more. We go into more of this limitation in Section 6.1 with ideas for rectification and modeling isolation.

Chapter 6

Conclusion

This thesis argues that reliable navigation across diverse off-road environments requires terrain-aware dynamics models grounded in structured perception and validated against worst-case behavior in the open- and closed-loop. Toward this end, we first showed that BEV representations carry spatial structure that prior perception-conditioned models leave underutilized when perception is consumed as a static prior. By instead querying the BEV local to the robot's predicted position at every step of the rollout, our model grounds each incremental state in the terrain the robot is expected to traverse; this context also drives a residual correction. This explicit model structure improved open-loop average prediction accuracy substantially, especially when given more complete BEV maps via inpainting.

Accuracy on average, however, is a weak guarantee for a system that must operate at its limits in any environment. Our second contribution addressed this by establishing a rigorous evaluation framework with an aggressive driving dataset, spanning environments and seasons, and a benchmarking method for mining the most challenging evaluation samples. Clustering similar actions & initial conditions and isolating high trajectory variance regimes surfaced failure modes for baselines that aggregate metrics obscured. For example, the FPV-conditioned model which was competitive on large datasets had 109% higher worst-case prediction error than our approach. Our benchmark yields a clearer understanding of when a model is fit to deploy in the real world by determining if its performance holds up on the most challenging, long-tail scenarios where failure is most likely.

Finally, we evaluated how important dynamics modeling accuracy is in the context of closed-loop control. We deployed our dynamics model in an MPC framework on a full-scale autonomous ATV and completed high-speed and challenging geometry courses. Our experiments confirmed the importance of accurate dynamics models, where physics baselines failed repeatedly. Among the learned models, however, we observed diminishing returns. Beyond a threshold, the controller absorbs the model error and further prediction accuracy no longer registers in tracking performance. Future work will investigate new navigation tasks and approaches to disentangle modeling error from controller error.

Taken together, these contributions support our thesis statement by providing a terrain-aware dynamics model that is validated against worst-case behavior on challenging offline datasets and on rigorous real-world deployment. The terrain-aware formulation developed here extends naturally beyond control to MBRL architectures with data augmentation [45], traversability estimation [68], kinodynamic planning [30, 47], and simulation. We demonstrated how our dynamics model leveraged structured perception representations to become terrain-aware, improving both prediction and closed-loop navigation across diverse and challenging off-road environments.

6.1 Limitations & Future Work

Our terrain-aware dynamics modeling and validation approaches improve modeling accuracy and evaluation rigor difficult relative to the state-of-the-art, yet limitations remain. In this section, we discuss the scope of each contribution and its limitations, alongside future research directions to address them.

6.1.1 Terrain-Aware Dynamics Modeling

Modeling Uncertainty

Our approach predicts the deterministic state-transition function f_θ. The largest limitation of a deterministic model is its inability to represent modeling uncertainty. A probabilistic state-transition function instead predicts a distribution over next states, whose spread reflects that uncertainty. As mentioned in Section 2.2.3, there

are many ways to formulate probabilistic state representations, such as learning a Gaussian Process residual disturbance over a prior prediction [16, 58], using ensembles of models [11, 45], or using quantile regression [17].

We can leverage this uncertainty for risk-aware navigation approaches [17, 21, 86] or early-stage training exploration [11]. Gibson et al. [21] find that high-speed navigation in narrow corridors with high uncertainty leads to the prediction distribution spreading over costly areas, resulting in suboptimal solutions. The risk-aware objective therefore favors lower-speed trajectories which have tighter distributions and stay clear of these regions. Chua et al. [11] find prediction uncertainty to reduce overfitting in the earlier stages of training, helping MBRL formulations reach similar performance to model-free RL algorithms, with several orders of magnitude less training data.

Performance in Novel Domains

Our dynamics model could be further improved through online adaptation when experiencing novel inputs. While our approach performed comparably to proprioceptive baselines on OOD winter data in Section 5.2.2, an adaptive mechanism could allow the model to correct for its inaccuracy online. One such approach could be to treat the trained network as a learned basis and adapt only a small set of final-layer weights or coefficients, driven by an adaptation signal [32, 44, 50, 59]. This adaptation signal could be formulated from prediction uncertainty [44] or modeling error [59, 77].

Cross-Platform Generalization

We train and evaluate our dynamics model on a single platform. The architecture itself makes no assumptions specific to our vehicle, but the learned weights do. Therefore, differing action spaces, actuator dynamics, and physical parameters would likely degrade performance on another wheeled platform without retraining. Transfer to an entirely different locomotion mechanism, such as a legged quadruped, is less likely still, given the vastly different action space and contact mechanics. We do expect the architecture to transfer given retraining on a new platform, but retraining per platform is not cross-platform generalization.

One approach to improve the cross-platform generalization could be to train on larger multi-platform datasets, as Xiao et al. [92] do, though this particular

approach constrains all platforms to a shared action space. Liu et al. [48] train on a multi-platform, multi-action-space dataset with a long-context RL formulation which adapts to new action spaces online. Larger datasets are a crude remedy, and collecting multi-platform data at this scale is costly. Investigating data-scarce methods and general contact-based models could be a promising future direction for cross-platform generalization [27, 56]. For example, by reasoning about the mechanics of contact itself rather than the morphology that produces it, whether a wheel or a foot, we could design architectures that can generalize across many locomotion mechanisms without requiring training data for each platform.

Perception Resolution

While we demonstrate the benefit of terrain-aware perception representations, our BEV resolution of 40 cm is coarse. Additional perception representations could help, such as hierarchical maps with high resolution in nearby cells and low resolution in far-away cells. Optimizing this hierarchical structure against prediction accuracy as a co-training task is another promising direction.

Inference Latency

One limitation of our explicit BEV-querying approach is inference latency. On hardware specified in Section A.3.3, with 825 parallelized rollouts, our approach has a 28.3 ms inference time, whereas Vanilla BEV has an inference time of 12.9 ms. While our additional structure improves prediction, it is at the cost of a ∼2x greater inference. We also evaluate on Nvidia Thor edge-compute, where MPPI with our dynamics rollouts takes 112.3 ms but 44.7 ms. Our approach will currently struggle to run at 10 Hz on edge compute, likely impacting control & navigation task performance.

6.1.2 Worst-Case Benchmarking

Sensitivity to Ground-Truth Noise

Our benchmark ranks samples by trajectory variance within clusters of similar actions and initial conditions, then evaluates on the highest-variance decile. This makes the benchmark sensitive to noise in the ground-truth states themselves, which we

derive partially from state estimation [96]. A state-estimation outlier may inflate apparent trajectory variance and thus undermine the premise that high-variance clusters isolate difficult terrain. The mined set may therefore contain samples that are hard to predict because the ground truth is wrong, not because the maneuver is demanding, and a model penalized on these is being penalized for failing to fit noise. We do not currently distinguish these cases. Future work could filter samples with large state-estimation covariance.

6.1.3 Closed-Loop Evaluation

Our third contribution asks how much dynamics-model accuracy actually matters once the model is placed inside a controller. Answering this cleanly requires separating error attributable to the model from error attributable to the controller, which our current experimental design achieves only partially.

Isolating Modeling from Control Error

We attempt to separate modeling error from control error in Section 5.4, but failures attributable to the controller alone remain entangled with those of the model, as discussed in Section A.2.2. The key issue is that our MPPI controller carries numerous cost terms and sampling strategies, each with tunable parameters, so a tracking failure can originate in sampling sparsity or the cost weighting rather than in the predicted rollout. One alternative would be to replace MPPI with a gradient-based optimizer, such as LQR, whose fewer parameters make control-induced error easier to account for; however, linearizing our learned dynamics about the reference path may poorly approximate the model in aggressive regimes and add additional computation time.

Experiments for Attributing Failures

Other experiments could measure the impact of dynamics models without isolating them outright. One approach is to compare the accumulated traversability cost of a full autonomy stack under different dynamics models while holding the MPC and traversability estimation fixed. A less accurate model may steer the robot into more costly terrain and accrue more cost over a course.

Another approach is to run the same course twice: once following a dense reference line, and once with full autonomy over a sparse set of waypoints. If the line-following run requires no interventions while the full autonomy run does, the failure likely originates upstream of the dynamics model, in planning or traversability estimation, rather than in prediction itself.

Appendix A

Implementation & Analysis

A.1 Perception Pipeline

A.1.1 Mapping Resolution Analysis

As mentioned in Section 3.2.3, we choose to use $0.4 \times 0.4 \times 0.1$ m or 40 cm voxel resolution over a $100 \times 100 \times 20$ m spatial extent largely because most of our prior work for traversability estimation works with BEV maps of the same resolution and extent. While a higher resolution along the x and y dimensions would give finer detail, it also will require more onboard memory and storage to store training datasets. Additionally, high resolution reaches an upper bound, where the resolution of the LiDAR starts to become an issue, shown by Figure A.1. Future work could explore using 20 cm resolution or hierarchical resolution data structures, where voxels near the robot are high resolution and voxels far away are low resolution. For context, our primary robot's tire are $\sim$25 cm wide.

A.2 MPPI Implementation

A.2.1 Cost Function

Our path tracking results used MPPI [89, 90] as a model-predictive controller. We follow the standard formulation of MPPI, which requires us to specify 1) a dynamics

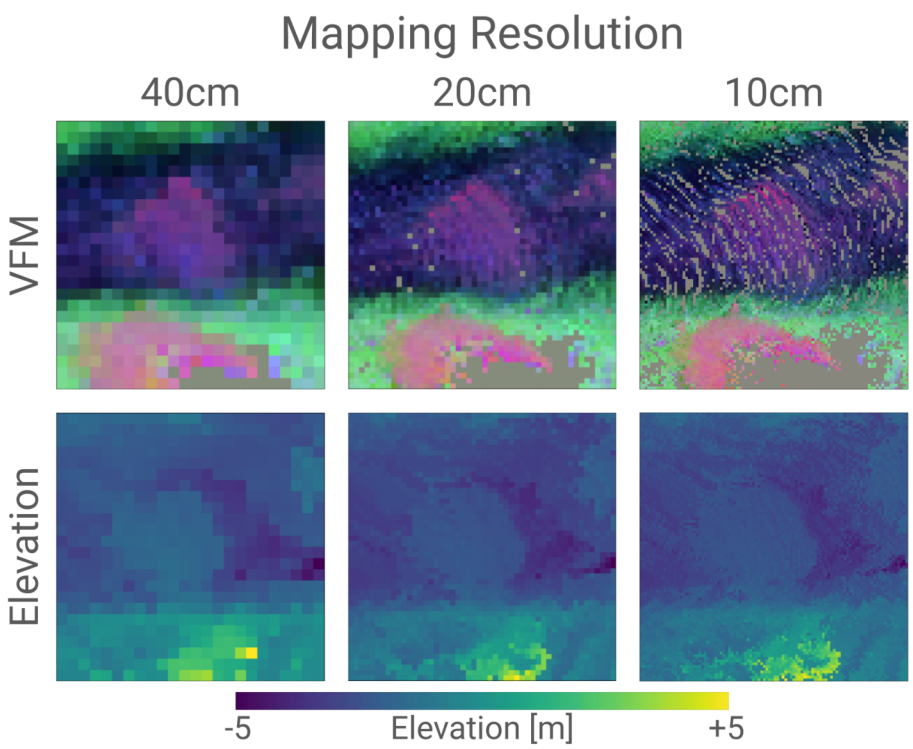


Figure A.1: A comparison of BEV resolutions, zoomed in around a 10×10 m section of a BEV map. We use the 40 cm resolution. While the 10 cm resolution map increases sharpness, it is too sharp, and the LiDAR scan sparsity becomes apparent through the empty regions where no points were.

model, 2) an action sampler, and 3) a cost function. We have deployed this MPPI implementation on numerous platforms, including Ackermann-steered wheeled robots, skidsteer wheeled robots, tracked robots, and quadrupeds.

Our learned models serve as the dynamics model that MPPI optimizes over. Following recent advances in action-sampling techniques for MPPI [6, 84], we generate candidate action sequences via additive noise generated from a set of Gaussian walks with varying σ as in (2.9). We also add a fixed set of uniform action sequences as in (2.10) 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 (A.1). In a nutshell, this cost function encourages the robot

to make forward progress along the reference path τ_{ref} while adhering to its shape (A.4, A.5), subject to speed and lateral acceleration constraints (A.2, A.3). 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 5.4 and analyzed in Section A.2.2, 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 (A.4, A.5). Hyperparameters are presented in Table A.2.

$$\ell(\tau) = \ell_{\text{speed}}(\tau, v_{\max}) + \ell_{a_y}(\tau, a_{\max}) + K_{\text{track}}\ell_{\text{track}}(\tau, \tau_{\text{ref}}) + K_{\text{progress}}\ell_{\text{progress}}(\tau, \tau_{\text{ref}}) \tag{A.1}$$

$$\ell_{\text{speed}}(\tau, v_{\max}) = \sum_{x \in \tau} \left[\begin{cases} \infty \text{ if } x_v > v_{\max} \\ 0 \text{ otherwise} \end{cases} \right] \tag{A.2}$$

$$\ell_{a_y} = \sum_{x \in \tau} \left[\begin{cases} \infty \text{ if } \frac{x_v^2 \tan(\delta)}{L} > a_{\max} \\ 0 \text{ otherwise} \end{cases} \right] \tag{A.3}$$

$$\ell_{\text{progress}}(\tau, \tau_{\text{ref}}) = \sum_{i=i_{\text{start}}}^{i<i_{\text{end}}-1} ||\tau_{\text{ref}}^{(i+1)} - \tau_{\text{ref}}^{(i)}||_2 \tag{A.4}$$

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

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

$$\ell_{\text{track}}(\tau, \tau_{\text{ref}}) = \frac{1}{H} \sum_{x \in \tau} \left[\min_{x_{\text{ref}} \in \tau_{\text{ref}}} [||x_{\text{ref}} - x||_2] \right] \tag{A.5}$$

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 A.1: 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 A.2: MPPI cost function parameters

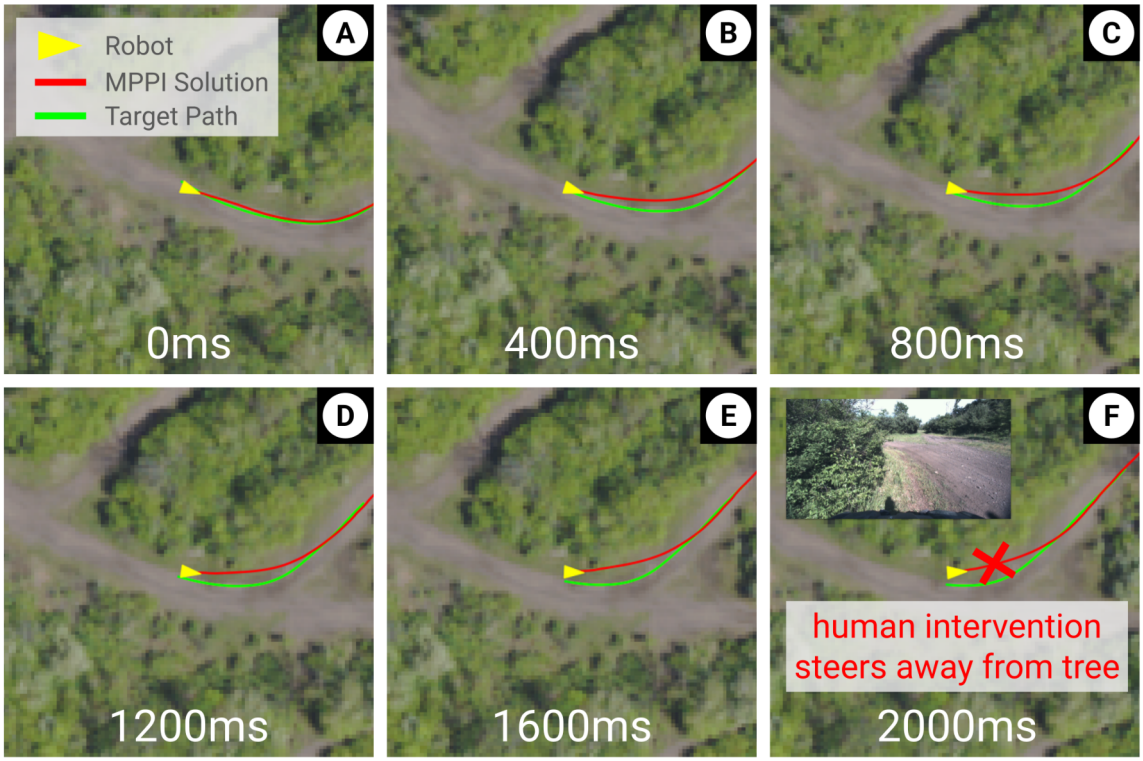


Figure A.2: A robot following a target path with its optimal MPPI solution visualized. It is closely following the line in (A), highlighting accurate dynamics and control, but from (B) onwards, MPPI determines the optimal path undercuts the target path, nearly resulting in collision with dense vegetation.

A.2.2 MPPI Failure Analysis

We detail why our model intervention from Figure 5.6 is not a modeling failure but instead a control failure due to the cost function tuning. Consider the scene

in Figure A.2 where each frame progresses after 400ms 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, 400ms 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 2000ms of following these undercutting solutions in (F), the human must intervene as the robot nears too dense of 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.

A.3 Learned Dynamics Model Implementation

A.3.1 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 A.3.

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× downsampling blocks. The BEV LSTM uses a CNN encoder with 4× downsampling

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×
Vanilla BEV CNN Downsample block	–	4×
BEV Features	F	36
Visual BEV Features	F_{VFM}	32
Terrain BEV Features	$F_{Terrain}$	4

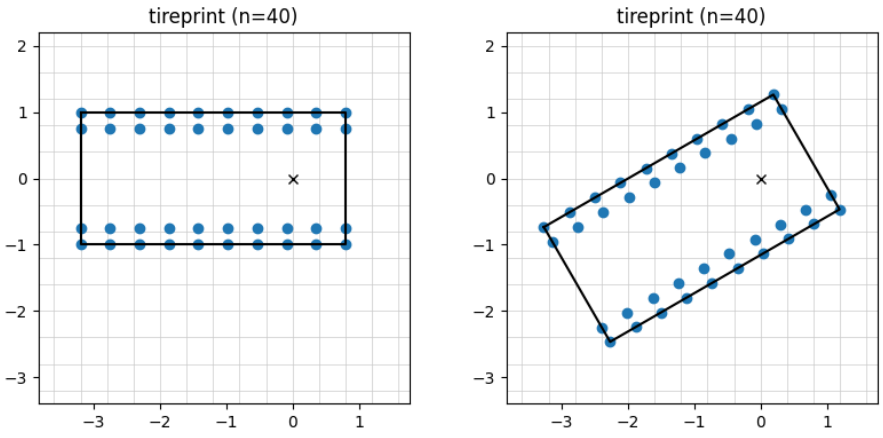
Table A.3: Hyperparameters for the LSTM architecture and perception encoders.

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 prediction, and it gets concatenated with the hidden input and then added to apply the correction step. Hyperparameters are in Table A.3.

A.3.2 BEV Footprint

Our footprint is two "tracks," separated 1m laterally from vehicle frame and uniformly spanning 4×0.25 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 A.3 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.



(a) (b)

Figure A.3: Tire track footprint visualization over a BEV map of the $100 \times 100 \times 0.4\,\mathrm{m}$ resolution aligned with the map (a) and rotated (b) to show how many new cells are queried.

A.3.3 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 an Nvidia GeForce RTX 5080 with 16GB of VRAM.

A.3.4 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 takes $\sim 30m$ to train the state-based methods and $\sim 6hr$ to train the BEV-based RNNs. The BEV-based transformer took about $\sim 24hr$ to train.

Bibliography

[1] Omar Alama, Avigyan Bhattacharya, Haoyang He, Seungchan Kim, Yuheng Qiu, Wenshan Wang, Cherie Ho, Nikhil Keetha, and Sebastian 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. 3.2.1

[2] American Association of State Highway and Transportation Officials. *A Policy on Geometric Design of Highways and Streets*. American Association of State Highway and Transportation Officials, Washington, DC, 4th edition, 2001. ISBN 1-56051-156-7. URL https://law.resource.org/pub/us/cfr/ibr/001/aashto.green.2001.pdf#page=556. GDHS-4; pp. 507 (design speed), 510 (Exhibit 8–1: Maximum Grades for Rural and Urban Freeways). 1, 2

[3] Anelia Angelova, Larry Matthies, Daniel Helmick, Pietro Perona, and Sebastian Scherer. Learning and prediction of slip from visual information. *Robotics: Science and Systems 2026*, 24(3):205–231, 2007. 1.1

[4] Pranav Atreya, Haresh Karnan, Kavan Singh Sikand, Xuesu Xiao, Sadegh Rabiee, and Joydeep 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. 2.2.3

[5] Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15791–15801, June 2025. 2.2.4

[6] Mohak Bhardwaj, Balakumar Sundaralingam, Arsalan Mousavian, Nathan D Ratliff, Dieter Fox, Fabio Ramos, and Byron 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. 2.3.1, A.2.1

[7] Michael Bode. Learning the forward predictive model for an off-road skid-steer vehicle. Master's thesis, Carnegie Mellon University, 2007. 1.1, 3.4

[8] Jiaqi Chen, Jonas Frey, Ruyi Zhou, Takahiro Miki, Georg Martius, and Marco

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. 2.2.4

[9] Fabian Christ, Alexander Wischnewski, Alexander Heilmeier, and Boris Lohmann. Time-optimal trajectory planning for a race car considering variable tyre-road friction coefficients. *Vehicle System Dynamics*, 59:588 – 612, 2021. URL https://api.semanticscholar.org/CorpusID:213649836. 2.4.1

[10] John Chrosniak, Jingyun Ning, and Madhur Behl. Deep dynamics: Vehicle dynamics modeling with a physics-constrained neural network for autonomous racing. *IEEE Robotics and Automation Letters*, 9(6):5292–5297, 2024. doi: 10.1109/LRA.2024.3388847. 2.2.2

[11] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31, 2018. 1.1, 1.2, 2.2.3, 6.1.1

[12] William D. Compton, Noel Csomay-Shanklin, Cole Johnson, and Aaron 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. 2.2.3

[13] Jared Di Carlo, Patrick M. Wensing, Benjamin Katz, Gerardo Bledt, and Sangbae Kim. Dynamic locomotion in the mit cheetah 3 through convex model-predictive control. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–9, 2018. doi: 10.1109/IROS.2018.8594448. 1.1

[14] Franck Djeumou, Jonathan Y.M. 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, 2023. doi: 10.1109/ICRA48891.2023.10161370. 2.2.2

[15] 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. 2.2.2

[16] Andreas Doerr, Christian Daniel, Martin Schiegg, Nguyen-Tuong Duy, Stefan Schaal, Marc Toussaint, and Trimpe Sebastian. Probabilistic recurrent state-space models. In Jennifer Dy and Andreas 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. 2.2.3, 6.1.1

[17] David D. Fan, Ali-akbar Agha-mohammadi, and Evangelos A. Theodorou. Deep learning tubes for tube mpc. *Robotics: Science and Systems XVI*, 2020. 2.2.3, 6.1.1

[18] Jonas Frey, Matias Mattamala, Nived Chebrolu, Cesar Cadena, Maurice Fallon, and Marco Hutter. Fast Traversability Estimation for Wild Visual Navigation. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.054. 3.2

[19] Jason Gibson, Bogdan Vlahov, David Fan, Patrick Spieler, Daniel Pastor, Ali-akbar Agha-mohammadi, and Evangelos A. Theodorou. A multi-step dynamics modeling framework for autonomous driving in multiple environments. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7959–7965, 2023. doi: 10.1109/ICRA48891.2023.10161330. 1.2, 2.2.2, 2.4.1, 3.3, 2

[20] Jason Gibson, Anoushka Alavilli, Erica Tevere, Evangelos A. Theodorou, and Patrick Spieler. Dynamics modeling using visual terrain features for high-speed autonomous off-road driving. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9809–9815, 2025. doi: 10.1109/ICRA55743.2025.11128031. 2.2.2, 2.2.4

[21] Jason Gibson, Bogdan Vlahov, Patrick Spieler, and Evangelos A. Theodorou. Multistep belief space dynamics learning for risk-aware control, 2026. URL https://arxiv.org/abs/2605.12628. 6.1.1

[22] Jonathan Y. Goh, Tushar Goel, and J. Christian Gerdes. Toward automated vehicle control beyond the stability limits: Drifting along a general path. *Journal of Dynamic Systems, Measurement, and Control*, 142(2):021004, 11 2019. ISSN 0022-0434. doi: 10.1115/1.4045320. URL https://doi.org/10.1115/1.4045320. 1.1

[23] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=S1lOTC4tDS. 1.1, 2.2.4

[24] Sina Hajimiri, Ismail Ben Ayed, and Jose 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. 3.2.1

[25] Tyler Han, Alex Liu, Anqi Li, Alexander Spitzer, Guanya Shi, and Byron 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. 1.1, 1.2

[26] Tyler Han, Sidharth Talia, Rohan Panicker, Preet Shah, Neel Jawale, and

Byron Boots. Dynamics models in the aggressive off-road driving regime. In *ICRA 2024 Workshop on Resilient Off-road Autonomy*, 2024. URL https://openreview.net/forum?id=CLBScrE0M7. 2.2.1, 1

[27] Zihao He, Bo Ai, Yulin Liu, Weikang Wan, Henrik I Christensen, and Hao Su. Cross-embodiment dexterous manipulation through world model learning. In *Dexterous Manipulation: Learning and Control with Diverse Modalities*, 2025. URL https://openreview.net/forum?id=KvFGpgHmIA. 6.1.1

[28] Benned Hedegaard, Ethan Fahnestock, Jacob Arkin, Ashwin Menon, and Thomas M. Howard. Discrete optimization of adaptive state lattices for iterative motion planning on unmanned ground vehicles. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5764–5771, 2021. doi: 10.1109/IROS51168.2021.9636181. 2.2.1

[29] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735. 5.1.3

[30] Thomas M Howard and Alonzo Kelly. Optimal rough terrain trajectory generation for wheeled mobile robots. *The International Journal of Robotics Research*, 26 (2):141–166, 2007. 1.1, 1.2, 4.3.1, 6

[31] Jemin Hwangbo, Joonho Lee, Alexey Dosovitskiy, Dario Bellicoso, Vassilios Tsounis, Vladlen Koltun, and Marco Hutter. Learning agile and dynamic motor skills for legged robots. *Science Robotics*, 4(26):eaau5872, January 2019. ISSN 2470-9476. doi: 10.1126/scirobotics.aau5872. URL https://www.science.org/doi/10.1126/scirobotics.aau5872. 2.2.3

[32] Tyler Ingebrand, Adam J Thorpe, and Ufuk Topcu. Zero-shot transfer of neural odes. *Advances in Neural Information Processing Systems*, 37:67604–67626, 2024. 2.2.3, 6.1.1

[33] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. *Advances in neural information processing systems*, 28, 2015. 3.4.1

[34] Edo Jelavic, Dominic Jud, Pascal Egli, and Marco Hutter. Robotic precision harvesting: Mapping, localization, planning and control for a legged tree harvester. *Field Robotics*, 2:1386–1431, 2022. doi: 10.55417/fr.2022046. 1.1

[35] Gregory Kahn, Pieter Abbeel, and Sergey 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. 2.2.3, 2.2.4

[36] Elia Kaufmann, Leonard Bauersfeld, Antonio Loquercio, Matthias Müller, Vladlen Koltun, and Davide Scaramuzza. Champion-level drone racing using deep reinforcement learning. *Nature*, 620(7976):982–987, 2023. 1.1

[37] Alonzo Kelly and Neal Seegmiller. A Vector Algebra Formulation of Mobile Robot Velocity Kinematics. In Kazuya Yoshida and Satoshi 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. URL https://doi.org/10.1007/978-3-642-40686-7_41. 1.2

[38] Alonzo Kelly, Omead Amidi, Mike Bode, Michael Happold, Herman Herman, Thomas Pilarski, Peter Rander, Anthony Stentz, Nick Vallidis, and Randy Warner. *Toward Reliable Off Road Autonomous Vehicles Operating in Challenging Environments*, volume 25, pages 599–608. 01 2004. ISBN 978-3-540-28816-9. doi: 10.1007/11552246_57. 1.1

[39] Yunho Kim, Chanyoung Kim, and Jemin 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. 2.2.3, 2.2.4

[40] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980. 5.1.3

[41] Alec Koppel, Jonathan Fink, Garrett Warnell, Ethan Stump, and Alejandro 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, October 2016. doi: 10.1109/IROS.2016.7759118. URL https://ieeexplore.ieee.org/document/7759118/. ISSN: 2153-0866. 2.2.3

[42] Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra 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. 2.2.3

[43] Hojin Lee, Taekyung Kim, Jungwi Mun, and Wonsuk Lee. Learning terrain-aware kinodynamic model for autonomous off-road rally driving with model predictive path integral control. *IEEE Robotics and Automation Letters*, 8(11):7663–7670, 2023. doi: 10.1109/LRA.2023.3318190. 1.1, 1.2, 2.2.2, 2.2.4, 3.4

[44] Jacob Levy, Jason Gibson, Bogdan Vlahov, Erica Tevere, Evangelos Theodorou, David Fridovich-Keil, and Patrick 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. 2.2.2, 6.1.1

[45] Jacob Levy, Tyler Westenbroek, and David Fridovich-Keil. Learning to walk from three minutes of real-world data with semi-structured dynamics models. In Pulkit Agrawal, Oliver Kroemer, and Wolfram 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. 1.1, 2.2.1, 2.2.3, 6, 6.1.1

[46] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng 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. 3.4.1

[47] Maxim Likhachev and Dave Ferguson. Planning long dynamically-feasible maneuvers for autonomous vehicles. In *Proceedings of Robotics: Science and Systems*, Zurich, Switzerland, 2008. doi: 10.15607/RSS.2008.IV.028. URL https://www.roboticsproceedings.org/rss04/p28.pdf. 1.1, 2.2.1, 6

[48] Min Liu, Deepak Pathak, and Ananye Agarwal. Locoformer: Generalist locomotion via long-context adaptation. In *9th Annual Conference on Robot Learning*, 2025. 6.1.1

[49] Yifei Liu. 3d thermal perception for autonomous navigation in visually degraded, unstructured environments. Master's thesis, Carnegie Mellon University, 2025. 4.2.1, 1

[50] Elena Sorina Lupu, Fengze Xie, James Alan Preiss, Jedidiah Alindogan, Matthew Anderson, and Soon-Jo 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. 1.2, 2.2.2, 2.2.4, 3.4, 6.1.1

[51] Parv Maheshwari, Wenshan Wang, Samuel Triest, Matthew Sivaprakasam, Shubhra Aich, John G. Rogers, Jason M. Gregory, and Sebastian Scherer. PI-Aug – Physics Informed Augmentation for Learning Vehicle Dynamics for Off-Road Navigation, November 2023. URL http://arxiv.org/abs/2311.00815. arXiv:2311.00815 [cs]. 2.2.2, 2.2.4, 1

[52] John Mai. System design, modelling, and control for an off-road autonomous ground vehicle. Master's thesis, Carnegie Mellon University, Pittsburgh, PA, July 2020. 4.2.1, 4.2.1

[53] Daniel Maturana. *Semantic Mapping for Autonomous Navigation and Exploration*. PhD thesis, Carnegie Mellon University, 2022. 3.2.4

[54] Nima Mohajerin, Melissa Mozifian, and Steven 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. 2.2.2

[55] Anusha Nagabandi, Gregory Kahn, Ronald S. Fearing, and Sergey 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. 2.2.3, 3.3

[56] Shafeef Omar and Majid Khadiv. Learning to act through contact: A unified view of multi-task robot learning. In *Proceedings of The 8th Annual Learning for Dynamics & Control Conference (L4DC)*, Proceedings of Machine Learning Research. PMLR, 2026. 6.1.1

[57] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=a68SUt6zFt. Featured Certification. 3.2.1

[58] Chris J. Ostafew, Angela P. Schoellig, and Timothy 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. 1.2, 2.2.2, 6.1.1

[59] Michael O'Connell, Guanya Shi, Xichen Shi, Kamyar Azizzadenesheli, Anima Anandkumar, Yisong Yue, and Soon-Jo 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. 1.1, 2.2.2, 6.1.1

[60] Hans B. Pacejka and Egbert Bakker. THE MAGIC FORMULA TYRE MODEL. *Vehicle System Dynamics*, 21(sup001):1–18, January 1992. ISSN 0042-3114, 1744-5159. doi: 10.1080/00423119208969994. URL http://www.tandfonline.com/doi/abs/10.1080/00423119208969994. 2.4.1

[61] Anuj Pokhrel, Mohammad Nazeri, Aniket Datar, and Xuesu 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. 2.2.4

[62] Ilija Radosavovic, Bike Zhang, Baifeng Shi, Jathushan Rajasegaran, Sarthak Kamat, Trevor Darrell, Koushil Sreenath, and Jitendra Malik. Humanoid locomotion as next token prediction. *Advances in neural information processing systems*, 37:79307–79324, 2024. 2.2.3

[63] Ayoub Raji, Alexander Liniger, Andrea Giove, Alessandro Toschi, Nicola Musiu, Daniele Morra, Micaela Verucchi, Danilo Caporale, and Marko Bertogna. Motion Planning and Control for Multi Vehicle Autonomous Racing at High Speeds. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, pages 2775–2782, October 2022. doi: 10.1109/ITSC55140.2022.9922239. URL https://ieeexplore.ieee.org/document/9922239/. 1.1, 2.2.1, 2.4.1

[64] Ayoub Raji, Nicola Musiu, Alessandro Toschi, Francesco Prignoli, Eugenio Mascaro, Pietro Musso, Francesco Amerotti, Alexander Liniger, Silvio Sorrentino, and Marko Bertogna. A tricycle model to accurately control an autonomous racecar with locked differential. In *2023 IEEE 11th International Conference on Systems and Control (ICSC)*, pages 782–789, 2023. doi: 10.1109/ICSC58660.2023.10449744. 2.4.1

[65] Mike Ranzinger, Greg Heinrich, Jan Kautz, and Pavlo 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. 3.2.1

[66] Charles Richter, Adam Bry, and Nicholas Roy. *Polynomial Trajectory Planning for Aggressive Quadrotor Flight in Dense Indoor Environments*, pages 649–666. Springer International Publishing, Cham, 2016. ISBN 978-3-319-28872-7. doi: 10.1007/978-3-319-28872-7_37. URL https://doi.org/10.1007/978-3-319-28872-7_37. 1.1

[67] Forrest Rogers-Marcovitz, Neal Seegmiller, and Alonzo Kelly. Continuous vehicle slip model identification on changing terrains. In *RSS 2012 Workshop on Long-term Operation of Autonomous Robotic Systems in Changing Environments*, 2012. 4.3.1

[68] Pascal Roth, Jonas Frey, Cesar Cadena, and Marco Hutter. Learned Perceptive Forward Dynamics Model for Safe and Platform-aware Robotic Navigation. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.001. 1.2, 2.2.4, 3.4, 4, 6

[69] Peter 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. URL https://www.sciencedirect.com/science/article/pii/0377042787901257. 4.3.2

[70] David E. Rumelhart and James L. McClelland. *Learning Internal Representations by Error Propagation*, pages 318–362. 1987. 3.3

[71] Andrew Saba, Aderotimi Adetunji, Adam Johnson, Aadi Kothari, Matthew Sivaprakasam, Joshua Spisak, Prem Bharatia, Arjun Chauhan, Brendan Duff,

Noah Gasparro, Charles King, Ryan Larkin, Brian Mao, Micah Nye, Anjali Parashar, Joseph Attias, Aurimas Balciunas, Austin Brown, Chris Chang, Ming Gao, Cindy Heredia, Andrew Keats, Jose Lavariega, William Muckelroy, Andre Slavescu, Nickolas Stathas, Nayana Suvarna, Chuan Tian Zhang, Sebastian Scherer, and Deva Ramanan. Fast and modular autonomy software for autonomous racing vehicles. *Field Robotics*, 4:1–45, 2024. doi: 10.55417/fr.2024001. 1.1

[72] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature*, 588:604–609, 2020. doi: 10.1038/s41586-020-03051-4. 1.1

[73] Neal Seegmiller, Forrest Rogers-Marcovitz, Greg Miller, and Alonzo Kelly. Vehicle model identification by integrated prediction error minimization. *The International Journal of Robotics Research*, 32(8):912–931, 2013. 1.2, 2.2.1, 2.2.3, 4.3.1

[74] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv 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. 3.4.3

[75] Matthew Sivaprakasam, Parv Maheshwari, Mateo Guaman Castro, Samuel Triest, Micah Nye, Steve Willits, Andrew Saba, Wenshan Wang, and Sebastian 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. 4.2

[76] Matthew Sivaprakasam, Samuel Triest, Cherie Ho, Shubhra Aich, Jeric Lew, Isaiah Adu, Wenshan Wang, and Sebastian 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. 1, 3.2, 1

[77] Jean-Jacques E. Slotine and Weiping Li. Composite adaptive control of robot manipulators. *Automatica*, 25(4):509–519, July 1989. ISSN 00051098. doi: 10.1016/0005-1098(89)90094-0. URL https://linkinghub.elsevier.com/retrieve/pii/0005109889900940. 1.1, 6.1.1

[78] Jean-François Tremblay, Travis Manderson, Aurélio Noca, Gregory Dudek, and David 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. 2.2.3

[79] Samuel Triest, Matthew Sivaprakasam, Sean J. Wang, Wenshan Wang, Aaron M. Johnson, and Sebastian 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. 2.2.3, 4.2

[80] Samuel Triest, Matthew Sivaprakasam, Shubhra Aich, David Fan, Wenshan Wang, and Sebastian Scherer. Velociraptor: Leveraging visual foundation models for label-free, risk-aware off-road navigation. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard, editors, *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pages 4483–4494. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/triest25a.html. 1, 3.2, 1

[81] Samuel Triest, Amirreza Shaban, David D. Fan, Wenshan Wang, and Sebastian Scherer. Travsuite: Traversability via self-supervised, uncertainty-aware irl and terrain estimation. *Robotics: Science and Systems 2026*, 2026. (document), 3.2, 3.2.3, 3.11, 3.5

[82] Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, M. N. Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, Michele Gittleman, Sam Harbaugh, Martial Hebert, Thomas M. Howard, Sascha Kolski, Alonzo Kelly, Maxim Likhachev, Matt McNaughton, Nick Miller, Kevin Peterson, Brian Pilnick, Raj Rajkumar, Paul Rybski, Bryan Salesky, Young-Woo Seo, Sanjiv Singh, Jarrod Snider, Anthony Stentz, William "Red" Whittaker, Ziv Wolkowicki, Jason Ziglar, Hong Bae, Thomas Brown, Daniel Demitrish, Bakhtiar Litkouhi, Jim Nickolaou, Varsha Sadekar, Wende Zhang, Joshua Struble, Michael Taylor, Michael Darms, and Dave Ferguson. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics*, 25 (8):425–466, 2008. doi: https://doi.org/10.1002/rob.20255. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.20255. 1.1

[83] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia 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. 3.3

[84] Bogdan Vlahov, Jason Gibson, David D Fan, Patrick Spieler, Ali-akbar Aghamohammadi, and Evangelos A Theodorou. Low frequency sampling in model predictive path integral control. *IEEE Robotics and Automation Letters*, 9(5): 4543–4550, 2024. 2.3.1, A.2.1

[85] Sean J. Wang, Samuel Triest, Wenshan Wang, Sebastian Scherer, and Aaron

Johnson. Rough Terrain Navigation Using Divergence Constrained Model-Based Reinforcement Learning. In *Proceedings of the 5th Conference on Robot Learning*, pages 224–233. PMLR, January 2022. URL https://proceedings.mlr.press/v164/wang22c.html. ISSN: 2640-3498. 2.2.3

[86] Sean J. Wang, Honghao Zhu, and Aaron M. Johnson. Pay attention to how you drive: Safe and adaptive model-based reinforcement learning for off-road driving. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 16954–16960, 2024. doi: 10.1109/ICRA57147.2024.10611059. 2.2.3, 6.1.1

[87] William Ward, Sarah Etter, Tyler Ingebrand, Christian Ellis, Adam 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. 2.2.3

[88] Carl Wellington, Aaron C Courville, and Anthony Stentz. Interacting markov random fields for simultaneous terrain modeling and obstacle detection. In *Robotics: Science and Systems*, volume 6, pages 1–8, 2005. 3.2.4

[89] Grady Williams, Paul Drews, Brian Goldfain, James M. Rehg, and Evangelos A. Theodorou. Aggressive driving with model predictive path integral control. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1433–1440, 2016. doi: 10.1109/ICRA.2016.7487277. 2.3.1, 5.4.1, A.2.1

[90] Grady Williams, Paul Drews, Brian Goldfain, James M. Rehg, and Evangelos 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. 2.2.1, 2.3.1, 5.4.1, A.2.1

[91] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS Datasets and Benchmarks 2021)*, 2021. 1.1

[92] Wenli Xiao, Haoru Xue, Tony Tao, Dvij Kalaria, John M. Dolan, and Guanya 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. 1.2, 2.2.1, 2.2.3, 6.1.1

[93] Haoru Xue, Edward L. Zhu, John M. Dolan, and Francesco 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. 2.2.2

[94] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11975–11986, 2023. 3.2.1

[95] Yanfu Zhang, Wenshan Wang, Rogerio Bonatti, Daniel Maturana, and Sebastian Scherer. Integrating kinematics and environment context into deep inverse reinforcement learning for predicting off-road vehicle trajectories. In Aude Billard, Anca Dragan, Jan Peters, and Jun 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. 2.2.3

[96] Shibo Zhao, Hengrui Zhang, Peng Wang, Lucas Nogueira, and Sebastian 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. 3.2.3, 4.2.1, 5.1.2, 6.1.2

[97] Zhipeng Zhao, Bowen Li, Yi Du, Taimeng Fu, and Chen Wang. PhysORD: A neuro-symbolic approach for physics-infused motion prediction in off-road driving. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11670–11677, 2024. 1.2

[98] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng 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. 3.4.1