arXiv:2607.13319v4 [cs.RO] 16 Oct 2026

Adapting Generalist Vehicle Models for High-Speed MPC Across Terrains

Rwik Rana[1], Jesse Quattrociocchi[1,2], Christian Ellis[1,2], Nathan Tsoi[1], Garrett Warnell[1,2], Joydeep Biswas[1]
[1]The University of Texas at Austin [2]DEVCOM Army Research Laboratory
rwik2000@utexas.edu

Abstract: High-speed off-road autonomy requires precise closed-loop control for a target vehicle while remaining robust across changing terrains. Recent forward kinodynamic (FKD) prediction foundation models suggest a promising path, starting from a generalist model and specializing it to the target platform. However, effective specialization remains challenging, as it often requires substantial real-world data, and models adapted to one setting can still overfit to specific terrains or driving regimes. We present `OptCar` (Optimized Car), a recipe for bridging the gap from generalist to specialist FKD models that preserves cross-terrain generalization while optimizing performance for a specific vehicle. `OptCar` introduces a transformer FKD architecture that uses FiLM to condition multi-step predictions on a single dynamics context token summarizing recent state-action history. It then specializes the generalist model using limited real-world data and targeted synthetic rollouts from environment-specific system identification. In closed-loop model predictive control (MPC) experiments across three terrains and an out-of-distribution cart-pulling task, the largest gains appear at 6 m/s, the highest speed evaluated and the regime in which slip dominates tracking error. On vegetation + dirt, the most slip-diverse terrain, `OptCar` reduces 6 m/s trajectory tracking error by roughly 55% relative to AnyCar fine-tuned on real data alone, and remains the most accurate even when an unseen cart payload changes the dynamics. With 5 minutes of real data per terrain, `OptCar` is competitive on road with a specialist trained on 30 minutes of road data and outperforms it when the terrain changes.

Keywords: Off-road vehicle dynamics, transformer models, history conditioning, model predictive control, sim-to-real adaptation

1 Introduction

High-speed driving over unstructured terrain underlies applications from search-and-rescue and agricultural robotics to autonomous off-road racing. At high speed, the tires operate near their friction limit, and the resulting slip varies with terrain and payload, so a controller must be both precise on the target vehicle and robust as the terrain changes. Analytical models such as the dynamic bicycle model are fast, but simplified vehicle and tire dynamics can limit prediction accuracy [1, 2], motivating learned models and corrections [3]. To run in real time on embedded hardware at high speed the model must be small, so accuracy must come under a tight fidelity-latency budget rather than from a larger network.

Forward kinodynamic (FKD) models predict future states from candidate action sequences and serve directly as the rollout model inside sampling-based model predictive control (MPC), such as MPPI [4], coupling planning and control. Generalist FKD models such as AnyCar [5] generalize across vehicles and environments but lack the per-vehicle accuracy needed for high-speed control, whereas inverse kinodynamic (IKD) methods [6, 7] adapt online to the current terrain through an

Project page: https://amrl.cs.utexas.edu/optcar/

10th Conference on Robot Learning (CoRL 2026), Austin TX, USA.

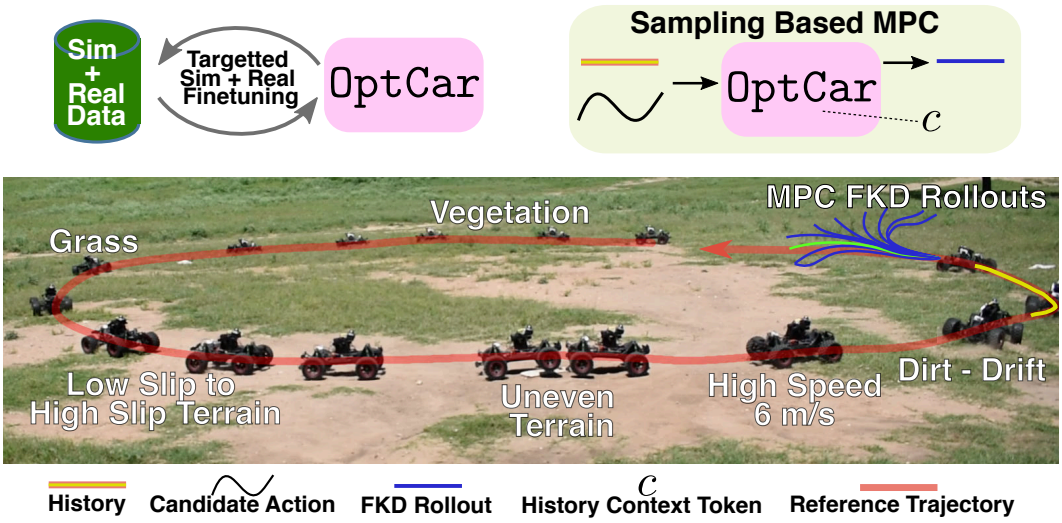


Figure 1: `OptCar` **driving at 6 m/s across mixed terrain.** The vehicle tracks a reference trajectory through a mixture of grass, vegetation, and dirt using sampling-based MPC. `OptCar` specializes a generalist FKD model through two contributions: a history context vector $\mathbf{c}$ that conditions the rollout decoder through FiLM to adapt to the current vehicle-terrain interaction, and a targeted real-and-synthetic fine-tuning recipe.

explicit history component, but rely on a separate planner. At the other extreme, specialist models fit a single vehicle and terrain from real data [8, 9], achieving aggressive performance in that setting, while transfer to new terrains may require further adaptation. Specializing a generalist FKD model to a known vehicle while remaining robust across terrains is the central problem we address.

We present `OptCar` (Figure 1), which addresses these limitations and is validated by placing the model inside an MPPI controller [4] for closed-loop trajectory tracking. Its contributions are:

1. **A targeted real-and-synthetic fine-tuning recipe** that specializes the model to a target vehicle from only minutes of real data per terrain, augmenting it with synthetic rollouts from a per-terrain system-identified bicycle model to cover the high-slip regimes. This lowers 6 m/s tracking error by 17% to 46% over fine-tuning on real data alone.
2. **FiLM-based history conditioning for FKD prediction** that compresses recent state-action history into a single dynamics context token and uses FiLM to condition every rollout decoder block, rather than cross-attending to individual history tokens. With `OptCar` and AnyCar, the history cross-attention baseline, both fine-tuned on the same real-and-synthetic dataset, `OptCar` reduces trajectory tracking error at 6 m/s by 19%–30%.
3. **Closed-loop validation against generalist and specialist baselines.** Against AnyCar fine-tuned on real data alone, `OptCar` fine-tuned on real and synthetic data cuts 6 m/s tracking error by 42% to 57%. Against a specialist trained on 30 minutes of road data, it remains within 14% on road yet tracks 46% more accurately on unseen terrain. Under an unseen cart payload, it attains the lowest tracking error at every speed regime.

2 Related Work

We review learned dynamics models for vehicle control, online dynamics adaptation, and data-efficient specialization via sim-to-real.

2.1 Learned Dynamics Models for Vehicle Control

We frame high-speed vehicle control as model-based planning under dynamics that vary with vehicle, terrain, and payload. Analytical kinematic and dynamic bicycle models [1, 2] are fast and interpretable, but simplified vehicle and tire dynamics can limit prediction accuracy, while end-to-end policies [10] forgo the predictive rollouts needed to evaluate alternative futures. Learned forward kinodynamic (FKD) models occupy the middle ground we adopt, predicting future trajectories

from candidate actions and serving as rollout models inside sampling-based MPC [4, 8, 11]. Within FKD-MPC, generalist models such as AnyCar [5] train across many vehicles and environments for broad transfer, whereas specialist methods fit a single vehicle, terrain, or regime from real data: drifting and tire models [9], off-road and terrain-aware rollouts [12, 13], and information-theoretic or residual-dynamics racing controllers [8, 3, 14, 15], while transfer to new terrains may require further adaptation. Our setting lies between the two, with a fixed vehicle that makes specialization worthwhile and a varying terrain that makes single-terrain specialization insufficient.

2.2 Online Dynamics Adaptation

Online adaptation methods use recent experience to account for changes in system dynamics, using state-action histories to condition policies [16, 17], represent changing disturbances [18, 19], or perform terrain-adaptive ground-vehicle tracking [6, 7]. For learned dynamics models, prior work adapts through online parameter updates [20, 21] or history-derived latent context for next-state prediction, as in CaDM [22]. Wang et al. [23] use a transformer-derived context to condition a probabilistic LSTM dynamics model rolled forward within MPPI, while HyperMPC [24] predicts time-varying dynamics-model parameters from history and expected future controls. `OptCar` adapts in a single forward pass: a single history-derived dynamics context token modulates every block of a multi-step transformer FKD decoder through FiLM, directly shaping the candidate-action rollouts MPC evaluates.

2.3 Data-Efficient Specialization and Sim-to-Real

`OptCar`'s second contribution, a targeted real-and-synthetic fine-tuning recipe, relates to the sim-to-real literature that uses limited real data to calibrate simulators or domain-randomization distributions [25, 26, 27, 28, 29]. These methods typically calibrate a simulator and then train or control within it, or tune a global randomization distribution. `OptCar` instead uses a per-terrain calibrated dynamic bicycle model purely as a targeted data generator that amplifies scarce real data into the high-slip regimes it under-samples; the learned FKD model, not the calibrated simulator, remains the rollout model deployed in MPC. This keeps the synthetic data anchored to the target vehicle rather than spread across broad randomization.

3 Problem Formulation

We consider forward kinodynamic (FKD) prediction for a target vehicle: given a history of recent states and applied actions together with a candidate future action sequence, predict the resulting future state trajectory. Let $\mathbf{s}_t \in \mathcal{S}$ denote the vehicle state (pose and velocity) at time t and $\mathbf{a}_t \in \mathcal{A}$ the applied action (body-frame velocity and steering command). Given a history spanning indices $t - M$ through t ($M + 1$ entries) and prediction horizon N, we seek a deterministic prediction model f_θ that maps the recent history and a candidate future action sequence to the predicted future trajectory,

$$\hat{\mathbf{s}}_{t+1:t+N} = f_\theta(\mathbf{s}_{t-M:t},\ \mathbf{a}_{t-M:t+N-1}), \tag{1}$$

with parameters θ chosen to minimize prediction error over the predicted future states $\hat{\mathbf{s}}_{t+1:t+N}$ on the operational distribution of the target vehicle across its terrains. Because it runs in closed-loop control, the model must be small enough to roll out many candidate actions in real time.

4 Methodology

`OptCar` specializes a pretrained FKD model to a target vehicle while adapting across terrains (Figure 2). The approach is built around two ideas. First, changing vehicle-terrain interaction is treated as a latent dynamics condition inferred online from recent motion; this inferred context then conditions all candidate future rollouts evaluated by MPC. Second, target-vehicle specialization is performed with limited real data augmented by targeted synthetic rollouts from terrain-specific system identification, expanding coverage of high-slip regimes that are difficult to collect safely.

4.1 History-Conditioned FKD Prediction

At high speed, the future motion of the vehicle is not determined by the measured state and commanded action alone. The same action sequence can produce different rollouts depending on latent factors such as terrain friction, slip regime, payload, and actuator response. These factors are

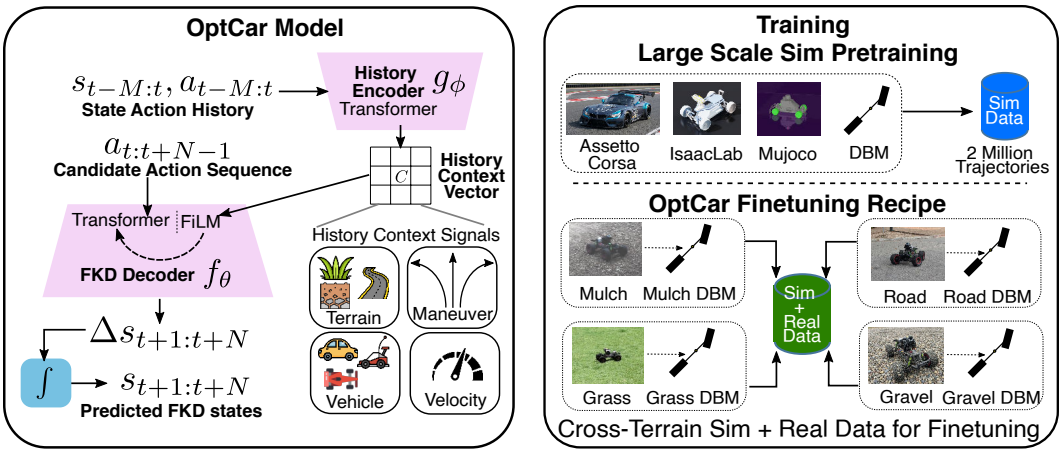


Figure 2: `OptCar` **overview.** *Left:* Recent state-action history is compressed into a dynamics context vector $\mathbf{c}$ that conditions FKD rollouts for candidate future actions. *Right:* `OptCar` is pretrained as a generalist FKD prior on simulation, then specialized to the target vehicle by fine-tuning on real data combined with targeted synthetic rollouts from per-terrain system identification.

not directly observed by the controller, but recent state-action transitions provide a local system-identification signal. `OptCar` therefore realizes the predictor of Eq. (1) in two stages: a history-to-context map g_ϕ and a context-conditioned rollout map f_θ whose predicted increments integrate to the future trajectory:

$$\mathbf{c}_t = \underbrace{g_\phi(\mathbf{s}_{t-M:t},\ \mathbf{a}_{t-M:t})}_{\text{history-to-context map}}; \quad \Delta\hat{\mathbf{s}}_{t+1:t+N} = \underbrace{f_\theta(\mathbf{s}_t,\ \mathbf{a}_{t:t+N-1},\ \mathbf{c}_t)}_{\text{context-conditioned rollout map}}. \tag{2}$$

The context $\mathbf{c}_t \in \mathbb{R}^d$ is not supervised with terrain or regime labels; it is learned end-to-end as the information from recent motion that is useful for predicting future rollouts. This factorization creates a bottleneck between history processing and rollout prediction: all history-dependent information available to the rollout model must pass through $\mathbf{c}_t$. Rather than cross-attending to individual history tokens, `OptCar` conditions every block of the rollout map f_θ on $\mathbf{c}_t$:

$$\mathbf{h}^{(l+1)} = \mathcal{B}_\theta^{(l)}\left(\mathbf{h}^{(l)};\ \mathbf{c}_t\right), \qquad l = 1, \ldots, L, \tag{3}$$

where $\mathcal{B}_\theta^{(l)}$ is the l-th block of the rollout map f_θ and $\mathbf{h}^{(l)}$ its hidden representation, with conditioning realized by FiLM [30] in our implementation. Every candidate rollout for the sampling-based MPC is conditioned on the same dynamics context, and $\mathbf{c}_t$ is computed once per planning step and reused across all candidate action sequences.

The decoder predicts body-frame pose increments $\Delta\hat{\mathbf{s}}_{t+k} = (\Delta\hat{\mathbf{p}}_{t+k}, \Delta\hat{\mathbf{q}}_{t+k})$ rather than global poses. This local representation removes dependence on global translation and supports rollouts over non-planar terrain. Architecture and training details are summarized in Section 5, with exact conditioning and integration equations in Appendix A.

4.2 Pretraining as a Generalist FKD Prior

Before target-vehicle specialization, `OptCar` is pretrained on a large simulation corpus spanning diverse vehicles, terrains, and driving regimes. The purpose of pretraining is to learn a broad prior over action-conditioned vehicle motion and to learn how recent state-action history can identify the active dynamics. Both the history-to-context map g_ϕ and the rollout map f_θ are trained jointly from random initialization. The training target is the body-frame future increment sequence in Eq. (2); corpus and loss details are summarized in Section 5 and Appendix A.

4.3 Targeted Real-and-Synthetic Specialization

Fine-tuning must balance relevance and coverage. Real target-vehicle data are most relevant but sparse in the aggressive high-slip regimes that matter for MPC, while broad simulation randomization adds coverage at the cost of capacity spent on unrelated vehicles and terrains. `OptCar` instead

uses each per-terrain real dataset twice, as direct training data and as the calibration signal for a terrain-specific synthetic data generator anchored to the same vehicle.

Let $\mathcal{D}^e_{\text{real}}$ denote the real target-vehicle trajectories collected on terrain $e \in \mathcal{E}$. For each terrain, we identify the parameters ψ_e of a dynamic bicycle model (DBM) F_{ψ_e} by fitting one-step transitions:

$$\psi_e^* = \arg\min_{\psi_e} \sum_{(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}) \in \mathcal{D}^e_{\text{real}}} \left\| F_{\psi_e}(\mathbf{s}_t, \mathbf{a}_t) - \mathbf{s}_{t+1} \right\|^2 . \tag{4}$$

These calibrated terrain-specific models are also used in the DBM-MPC baseline. The calibrated model $F_{\psi_e^*}$ is not deployed as `OptCar`'s rollout model. Instead, it is used offline to generate targeted synthetic rollouts $\mathcal{D}^e_{\text{synth}}$ under randomized action sequences that enter high-slip regions undersampled by the real data while remaining tied to the identified vehicle-terrain dynamics. The final fine-tuning set is

$$\mathcal{D}_{\text{FT}} = \bigcup_{e \in \mathcal{E}} \left(\mathcal{D}^e_{\text{real}} \cup \mathcal{D}^e_{\text{synth}} \right), \tag{5}$$

and the pretrained model is fine-tuned end-to-end on $\mathcal{D}_{\text{FT}}$ with the same pretraining prediction loss.

4.4 Deployment in Sampling-Based MPC

The fine-tuned model is the rollout function inside an MPPI controller [4]. At each planning step the history window is encoded into $\mathbf{c}_t$, candidate actions are rolled out with $f_\theta(\cdot, \cdot, \mathbf{c}_t)$, and the first action of the cost-weighted average of the sampled action sequences is applied. Since adaptation occurs through the forward pass for $\mathbf{c}_t$, deployment needs no terrain classifier, mode switch, or online parameter update.

5 Implementation Details

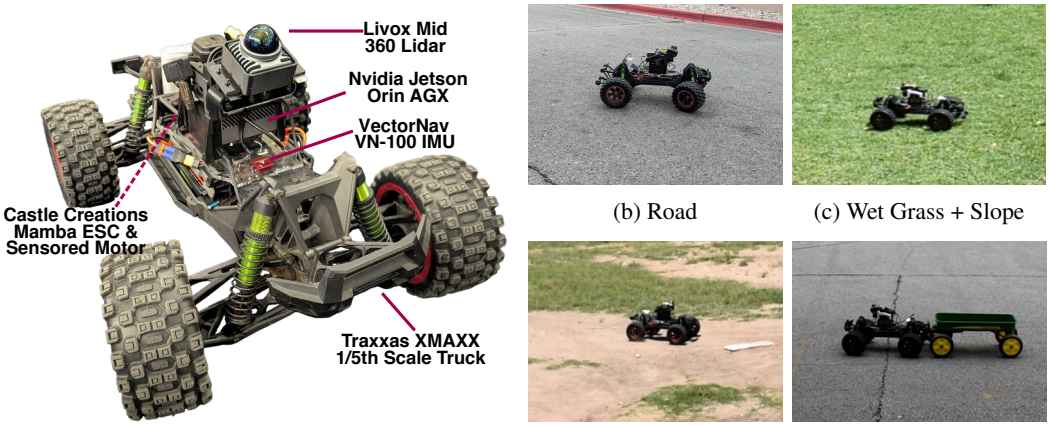
Architecture. `OptCar` uses the transformer FKD backbone and simulation protocol of AnyCar [5], with history processing factored as in Eq. (2). Historical state deltas, actions, and current-state features are embedded by per-stream MLPs into a shared dimension $d = 64$. Historical and candidate future actions share the same embedding MLP. The history encoder is a 2-layer bidirectional transformer that maps $M + 1 = 250$ history entries at $\Delta t = 0.02$ s to the context vector $\mathbf{c}_t$. The rollout decoder is a 2-block causal transformer with 4 attention heads and feed-forward dimension 256, conditioned by $\mathbf{c}_t$ through FiLM at each block, and predicts $N = 50$ body-frame SE(3) increments. Appendix A gives the conditioning equations and body-frame integration.

Training. The hyperparameters for the shared FKD backbone and optimization follow the AnyCar protocol [5] and are kept identical for `OptCar` and the AnyCar baseline. We pretrain on approximately 2 million simulated history–future windows from the AnyCar protocol across Assetto Corsa, MuJoCo, Isaac Lab, and randomized analytical dynamic bicycle models. The objective is $\mathcal{L} = 1.0\,\mathcal{L}_{\text{pos}} + 0.3\,\mathcal{L}_{\text{quat}}$ (Appendix A); optimization uses AdamW with weight decay 10^{-2}, learning rate 10^{-3}, cosine annealing, linear warmup, 500 epochs, and batch size 4096.

Fine-tuning uses approximately 5 minutes of real driving per terrain on road, gravel, grass, and mulch, yielding about 60K real history+future windows. The real fine-tuning data includes 6 m/s maneuvers. Targeted simulation adds about 400K windows by rolling out each $F_{\psi_e^*}$ for 10 s at $\Delta t = 0.02$ s under randomized Bézier velocity and random-walk steering commands within $[-2, 8]$ m/s and $[-0.41, 0.41]$ rad, with velocity noise $\sigma = 0.2$. We reserve 10% of the driving runs for validation and construct training and validation windows from separate runs. The combined real-and-synthetic fine-tuning dataset is randomly shuffled. Fine-tuning uses the same optimizer for 200 epochs at learning rate 10^{-3} and batch size 4096.

Deployment. At each planning step, we find the point on the reference curve nearest to the current vehicle position and sample H reference points ahead at equal time intervals $\Delta t = 0.02$ s, advancing along the curve at v_{ref}. The reference point for lookahead step k is therefore $v_{\text{ref}} k \Delta t$ along the curve from the nearest point and is shared by all candidate rollouts. MPPI samples 600 candidate action sequences per step and scores each rollout using

$$C = \sum_{k=1}^{H} \gamma^k \left[w_{\text{long}} e^2_{\text{long},k} + w_{\text{lat}} e^2_{\text{lat},k} + w_\psi \Delta\psi_k^2 + w_v (v_k - v_{\text{ref}})^2 \right], \tag{6}$$



(a) Platform components

(b) Road

(c) Wet Grass + Slope

(d) Vegetation + Dirt

(e) Cart Pulling

Figure 3: **Experimental platform and test environments.** Platform sensor and compute stack, followed by the terrain and payload conditions used for figure-8 tracking.

where $H = N = 50$, $\gamma = 0.97$, $w_{\text{long}} = 5$, $w_{\text{lat}} = 75$, $w_{\psi} = 1$, and $w_{v} = 15$. Here, $e_{\text{long},k}$ and $e_{\text{lat},k}$ are longitudinal and lateral tracking errors, $\Delta\psi_k$ is heading error, v_k is predicted vehicle speed, and v_{ref} is commanded speed. The system is implemented in PyTorch and CuPy and runs onboard an NVIDIA Jetson AGX Orin. Both `OptCar` and AnyCar use FP16 TensorRT inference for onboard deployment. With 600 candidate rollouts and a 50-step prediction horizon, `OptCar` replans at 37 Hz, compared with 32 Hz for AnyCar.

6 Experiments

We design experiments to answer three research questions:

- **RQ1. Specialized generalist versus baselines:** Can `OptCar` match or exceed both generalist and specialist models in cross-terrain tracking and out-of-distribution dynamics settings?
- **RQ2. Targeted real+sim fine-tuning:** Does fine-tuning on limited real data combined with targeted synthetic rollouts improve closed-loop tracking over fine-tuning on just real data?
- **RQ3. History Conditioning:** Does FiLM-based history conditioning improve closed-loop tracking compared with context concatenation and cross-attention over history tokens?

6.1 Experimental Setup

Platform setup: All experiments use our target 1/5-scale off-road vehicle with onboard 3D LiDAR, IMU, and an NVIDIA Jetson AGX board (Figure 3). Localization is provided by SuperOdometry [31], and a low-level PD controller maps commanded velocities to PWM signals.

Reference trajectories and terrains: We evaluate closed-loop figure-8 tracking at three commanded speeds: easy ($v_{\text{ref}} = 2$ m/s), medium ($v_{\text{ref}} = 4$ m/s), and hard ($v_{\text{ref}} = 6$ m/s). Road uses a tighter 17 m figure-8, while wet grass with slope and vegetation + dirt use a 30 m figure-8 so that 6 m/s tracking remains feasible while preserving the high-slip modeling challenge. The trajectory parameterization is given in Appendix B.

Metrics: We report trajectory tracking error (TTE), the mean distance from the vehicle to the closest point on the reference trajectory, and velocity tracking error (VTE), the mean absolute deviation from the commanded speed. Means and standard deviations are computed over per-timestep tracking errors pooled across three runs.

Baselines: We compare models along two axes: how history conditions the rollout decoder and the fine-tuning recipe. **AnyCar** [5] cross-attends to individual history tokens, while `OptCar` and **Concat** compress history into a single dynamics context token and inject it through FiLM or input concatenation, respectively. For fine-tuning, **FT-R** uses real target-vehicle data, while **FT-RS** additionally includes targeted synthetic rollouts from per-terrain system identification. We also compare against a terrain-calibrated dynamic bicycle model (**DBM**), **IKD FT-R** (Appendix D), and `OptCar`

Env.	Method	Easy (2 m/s)		Medium (4 m/s)		Hard (6 m/s)	
		TTE	VTE	TTE	VTE	TTE	VTE
Road	DBM	**0.011**	**0.004**	0.034	**0.019**	0.562	1.234
	IKD FT-R	0.016	0.032	0.067	0.189	0.412	1.849
	AnyCar FT-R	0.013	0.009	0.025	0.025	0.248	0.912
	AnyCar FT-RS	0.014	0.008	0.028	0.022	0.152	0.882
	Concat FT-RS	0.016	0.009	0.027	0.035	0.176	1.002
	`OptCar` Specialist	0.012	0.007	**0.020**	0.020	**0.093**	**0.711**
	`OptCar` FT-R	0.018	**0.004**	0.034	0.027	0.196	0.911
	`OptCar` FT-RS	0.016	0.013	0.026	0.023	0.106	0.901
Wet Grass + Slope	DBM	0.012	0.017	0.156	0.044	1.356	1.456
	IKD FT-R	0.013	0.029	0.062	0.255	2.371	1.004
	AnyCar FT-R	**0.011**	0.025	0.100	0.083	1.082	**0.892**
	AnyCar FT-RS	0.014	0.022	0.052	0.049	0.771	0.922
	Concat FT-RS	0.015	0.026	0.058	0.067	0.785	1.083
	`OptCar` Specialist	0.013	**0.016**	0.146	0.051	1.149	1.056
	`OptCar` FT-R	0.013	0.027	**0.031**	0.078	0.751	0.933
	`OptCar` FT-RS	0.012	0.025	0.044	**0.011**	**0.625**	0.940
Vegetation + Dirt	DBM	0.061	0.041	0.276	0.312	0.824	1.118
	IKD FT-R	0.055	0.033	0.232	0.452	0.669	1.452
	AnyCar FT-R	**0.054**	**0.019**	0.207	0.199	0.776	0.991
	`OptCar` FT-R	0.055	0.020	0.171	0.172	0.519	0.981
	`OptCar` FT-RS	0.059	0.023	**0.157**	**0.072**	**0.348**	**0.937**
Cart Pulling	DBM	0.129	0.014	0.212	0.197	1.012	1.881
	IKD FT-R	0.067	0.086	0.244	0.102	0.711	1.782
	AnyCar FT-R	0.042	0.031	0.182	0.049	0.489	1.338
	`OptCar` FT-R	0.028	0.028	0.174	0.062	0.319	1.290
	`OptCar` FT-RS	**0.011**	**0.014**	**0.085**	**0.044**	**0.229**	**1.023**

Table 1: **Closed-loop figure-8 tracking.** TTE (m ↓) and VTE (m/s ↓) for `OptCar` and baselines across terrains, speeds, and the unseen cart-pulling payload. We compare history-conditioning designs (Concat, `OptCar`, and AnyCar), fine-tuning recipes (FT-R, FT-RS, and Specialist), and DBM and IKD baselines. Results with standard deviations are provided in Appendix F, Table 3.

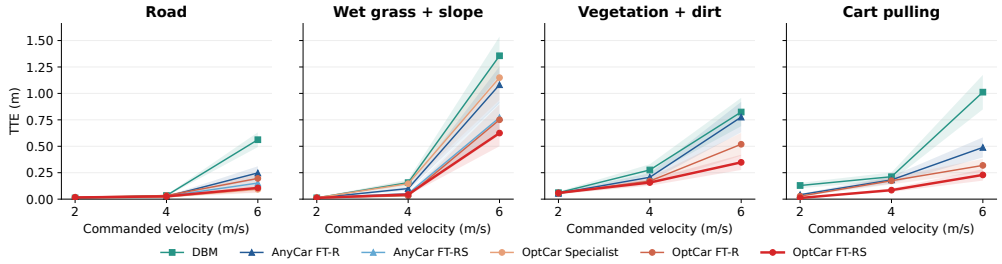


Figure 4: **Closed-loop tracking across speeds.** TTE (m ↓) for `OptCar` and baselines across terrains and the unseen cart-pulling payload. Shaded bands indicate ±1 standard deviation. Each panel shows the subset of these methods evaluated in that condition.

Specialist, trained only on 30 minutes of real road data. DBM and the IKD planner use road parameters for road and cart pulling, grass parameters for wet grass + slope, and dirt parameters for vegetation + dirt. `OptCar` is not fine-tuned on dirt, but is exposed to slippery terrains including mulch during fine-tuning and randomized dynamics during pretraining. AnyCar FT-RS, Concat FT-RS, and `OptCar` Specialist are evaluated only on road and wet grass + slope.

6.2 RQ1: Baseline Comparisons

We evaluate `OptCar` against generalist, specialist, and analytical baselines in three settings: cross-terrain figure-8 tracking, an unseen cart-pulling payload, and transfer away from a road-only specialist model. Table 1 summarizes the closed-loop tracking results, and Figure 4 shows how TTE varies with commanded velocity across environments.

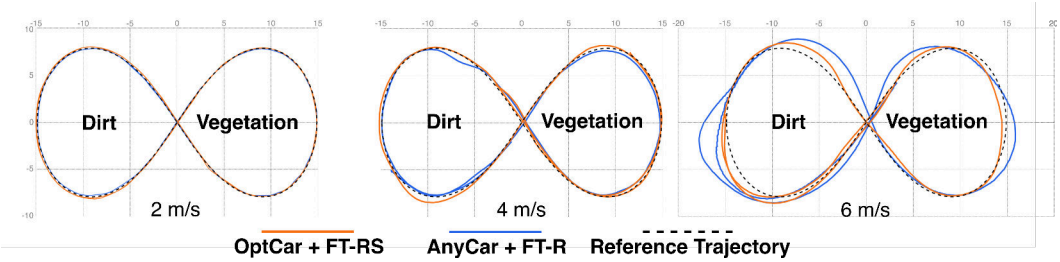


Figure 5: `OptCar` **FT-RS vs. AnyCar FT-R on vegetation + dirt.** Bird's-eye view of figure-8 tracking in the vegetation + dirt environment at three speeds. Slip is more pronounced in the dirt region, where `OptCar` FT-RS better maintains the reference trajectory. Grid axes are in meters.

6.2.1 Figure-8 Trajectory Tracking in Three Test Environments

The vehicle executes figure-8 trajectories on road, wet grass with slope, and vegetation + dirt at 2, 4, and 6 m/s. Each configuration is repeated three times for 15 s, and all models use identical MPPI parameters. Tracking accuracy is comparable across methods at low slip and diverges as speed and terrain-induced slip increase. At 2 m/s, all methods operate near the nominal regime and perform similarly. At 4 m/s, performance remains close on road, whose high traction keeps the dynamics predictable, while differences emerge on the lower-traction terrains. The separation is clearest at 6 m/s: `OptCar` FT-RS achieves lower trajectory tracking error than AnyCar FT-R on all three terrains, and on vegetation + dirt its error is less than half that of AnyCar FT-R (Figure 5). IKD FT-R degrades at high speed, as it tracks states produced by a separate DBM-MPPI planner rather than scoring candidate actions with the same learned dynamics used for execution.

6.2.2 Out-of-Distribution Cart Pulling

The cart-pulling task evaluates whether adaptation extends beyond the terrains seen during fine-tuning. The trailing cart alters the vehicle's effective inertia and tire loading, and no cart data is used during training. `OptCar` FT-RS attains the lowest tracking error across all speeds, with the largest margin at 6 m/s (Table 1, Figure 6). This demonstrates robustness to an unseen payload shift without cart-specific training data.

6.2.3 Comparison Against a Specialist Model

The road specialist provides a single-terrain reference: it uses the same `OptCar` architecture but is fine-tuned only on 30 minutes of real road data. On road, the specialist attains the best 6 m/s tracking, and `OptCar` FT-RS remains close despite using only 5 minutes of real data per terrain. On wet grass with slope, the specialist's accuracy degrades and is well behind `OptCar` FT-RS. This illustrates the RQ1 tradeoff: the road specialist is accurate in-distribution but transfers poorly to wet grass + slope, whereas multi-terrain specialization remains accurate in both conditions.

6.3 RQ2: Targeted Real+Sim Fine-Tuning

To evaluate the targeted real-and-synthetic fine-tuning recipe, we compare `OptCar` FT-R against `OptCar` FT-RS in Table 1. Both use the same history-conditioned architecture and real target-vehicle data; `OptCar` FT-RS additionally trains on synthetic rollouts generated from terrain-specific system identification. The real-only model remains competitive in moderate regimes, but the real+sim model is consistently stronger at 6 m/s, where aggressive slip dominates closed-loop error. `OptCar` FT-RS reduces trajectory tracking error at 6 m/s by 17%–46% relative to `OptCar` FT-R across all four conditions. On road and wet grass + slope, real-and-synthetic fine-tuning also improves AnyCar FT-RS over AnyCar FT-R at 6 m/s, reducing trajectory tracking error by 29%–39%. This supports the fine-tuning recipe: real data anchors the model to the target vehicle, while targeted synthetic data fills high-slip state-action regions that are difficult to collect safely.

6.4 RQ3: History Conditioning

To isolate the effect of the history-conditioning design, we compare Concat FT-RS, AnyCar FT-RS, and `OptCar` FT-RS on road and wet grass + slope in Table 1. All three are fine-tuned on the same real and targeted synthetic data; `OptCar` FT-RS uses a history context encoder and FiLM-conditioned

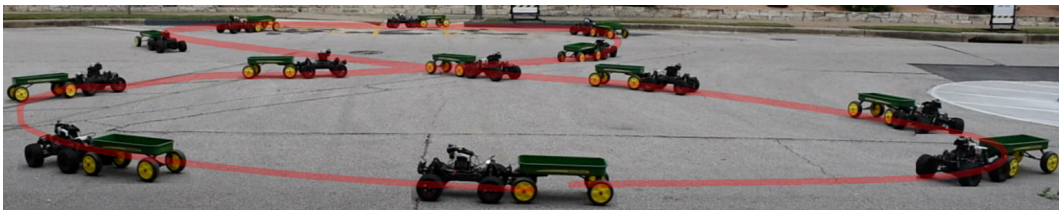


Figure 6: **Zero-shot cart pulling.** `OptCar` FT-RS tracking a figure-8 at 6 m/s with a trailing cart. No cart data is used during training.

decoder, whereas Concat supplies context as an additional input and AnyCar cross-attends to history tokens. Among the three conditioning designs, FiLM achieves the lowest trajectory tracking error at 4 and 6 m/s on both reported terrains. The gains are largest at 6 m/s, where `OptCar` FT-RS reduces trajectory tracking error relative to AnyCar FT-RS by 30% on road and 19% on wet grass with slope. A more extensive analysis of the learned context vector, projected with PCA and LDA, examines its variation across simulated vehicles, speeds, and turn directions (Appendix E).

7 Conclusion

We present `OptCar`, a recipe that specializes a generalist FKD foundation model to a vehicle for real-time MPC that stays accurate as the terrain changes. Our experiments yield three lessons. First, a single model can be both specialized to one vehicle and robust across terrains. Starting from a generalist prior and conditioning it on recent motion makes the model accurate on the target vehicle, while a specialist trained on a single terrain fails once the terrain changes. Second, compressing recent history into a single dynamics context token and conditioning the rollout decoder through FiLM enables adaptation to changing vehicle-terrain dynamics, improving high-speed tracking without online optimization. Third, targeted synthetic data from per-terrain system identification helps address the coverage-relevance trade-off of fine-tuning. Together, with only 5 minutes of real data per terrain, `OptCar` roughly halves 6 m/s tracking error relative to AnyCar FT-R and is competitive with a 30-minute single-terrain specialist on its home terrain while remaining accurate elsewhere. More broadly, FiLM-based history conditioning paired with targeted system-identification data helps in deploying foundation dynamics models on a specific robot without collecting large per-terrain datasets.

Limitations and future work. First, `OptCar` adapts only through its feedforward context vector and never updates its weights online, so it assumes the encountered dynamics lie within the pre-training and fine-tuning distribution. Genuinely novel dynamics would instead call for online model adaptation, as in meta-learning that updates the dynamics model on the fly [20], and combining fast feedforward context with slower online weight updates is a promising middle ground. Second, real-time MPPI tightly constrains the rollout model. Even at about 300K parameters, it affords only a short $N = 50$ horizon, because lengthening it slows each rollout and reduces the action samples evaluated per step. This myopic horizon motivates a scaling-law study of the trade-off between model capacity, rollout horizon, and sample budget. Third, `OptCar` adapts from proprioceptive history alone, and adding exteroceptive inputs such as RGB, geometric terrain (point clouds or elevation maps), or satellite-derived costmaps [32] could supply richer context and anticipate terrain changes before they appear in recent motion. Finally, planar DBM rollouts omit vertical motion, roll/pitch dynamics, and slope-dependent gravity, limiting synthetic coverage on steep or uneven terrain.

Acknowledgments

This work has taken place in the Autonomous Mobile Robotics Laboratory (AMRL) at UT Austin. AMRL research is supported in part by the National Science Foundation (OIA-2535195, CAREER-2046955, OIA-2219236, DGE-2125858, CCF-2319471). Any opinions, findings, and conclusions expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors. Joydeep Biswas holds a position as Visiting Professor with NVIDIA and receives financial compensation for this work. The terms of this arrangement have been reviewed and approved by the University of Texas at Austin in accordance with its policy on objectivity in research.
We thank Adam Uccello for his contributions to building the experimental platform.

References

[1] R. Rajamani. *Vehicle Dynamics and Control*. Mechanical Engineering Series. Springer, New York, NY, 2nd edition, 2012. ISBN 978-1-4614-1433-9. doi:10.1007/978-1-4614-1433-9. URL https://doi.org/10.1007/978-1-4614-1433-9.

[2] A. Raji, N. Musiu, A. Toschi, F. Prignoli, E. Mascaro, P. Musso, F. Amerotti, A. Liniger, S. Sorrentino, and M. 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. IEEE, Dec. 2023. doi:10.1109/ICSC58660.2023.10449744. URL https://doi.org/10.1109/ICSC58660.2023.10449744.

[3] J. Kabzan, L. Hewing, A. Liniger, and M. N. Zeilinger. Learning-based model predictive control for autonomous racing. *IEEE Robotics and Automation Letters*, 4(4):3363–3370, Oct. 2019. ISSN 2377-3774. doi:10.1109/LRA.2019.2926677. URL https://doi.org/10.1109/LRA.2019.2926677.

[4] G. Williams, A. Aldrich, and E. A. Theodorou. Model predictive path integral control: From theory to parallel computation. *Journal of Guidance, Control, and Dynamics*, 40(2):344–357, Feb. 2017. ISSN 1533-3884. doi:10.2514/1.G001921. URL https://doi.org/10.2514/1.G001921.

[5] W. Xiao, H. Xue, T. Tao, D. Kalaria, J. M. Dolan, and G. Shi. AnyCar to anywhere: Learning universal dynamics model for agile and adaptive mobility. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8819–8825. IEEE, May 2025. doi:10.1109/ICRA55743.2025.11128396. URL https://doi.org/10.1109/ICRA55743.2025.11128396.

[6] X. Xiao, J. Biswas, and P. Stone. Learning inverse kinodynamics for accurate high-speed off-road navigation on unstructured terrain. *IEEE Robotics and Automation Letters*, 6(3):6054–6060, July 2021. ISSN 2377-3774. doi:10.1109/LRA.2021.3090023. URL https://doi.org/10.1109/LRA.2021.3090023.

[7] H. Karnan, K. S. Sikand, P. Atreya, S. Rabiee, X. Xiao, G. Warnell, P. Stone, and J. Biswas. VI-IKD: High-speed accurate off-road navigation using learned visual-inertial inverse kinodynamics. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3294–3301. IEEE, Oct. 2022. doi:10.1109/IROS47612.2022.9982060. URL https://doi.org/10.1109/IROS47612.2022.9982060.

[8] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou. Information-theoretic model predictive control: Theory and applications to autonomous driving. *IEEE Transactions on Robotics*, 34(6):1603–1622, Dec. 2018. ISSN 1941-0468. doi:10.1109/TRO.2018.2865891. URL https://doi.org/10.1109/TRO.2018.2865891.

[9] F. Djeumou, J. Y. M. Goh, U. Topcu, and A. Balachandran. Autonomous drifting with 3 minutes of data via learned tire models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 968–974. IEEE, May 2023. doi:10.1109/ICRA48891.2023.10161370. URL https://doi.org/10.1109/ICRA48891.2023.10161370.

[10] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba. End to end learning for self-driving cars, 2016. URL https://arxiv.org/abs/1604.07316. arXiv preprint arXiv:1604.07316.

[11] M. S. Gandhi, B. Vlahov, J. Gibson, G. Williams, and E. A. Theodorou. Robust model predictive path integral control: Analysis and performance guarantees. *IEEE Robotics and Automation Letters*, 6(2):1423–1430, Apr. 2021. ISSN 2377-3774. doi:10.1109/LRA.2021.3057563. URL https://doi.org/10.1109/LRA.2021.3057563.

[12] J. Gibson, B. Vlahov, D. Fan, P. Spieler, D. Pastor, A.-a. Agha-mohammadi, and E. A. Theodorou. A multi-step dynamics modeling framework for autonomous driving in multiple environments. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7959–7965. IEEE, May 2023. doi:10.1109/ICRA48891.2023.10161330. URL https://doi.org/10.1109/ICRA48891.2023.10161330.

[13] H. Lee, T. Kim, J. Mun, and W. Lee. Learning terrain-aware kinodynamic model for autonomous off-road rally driving with model predictive path integral control. *IEEE Robotics and Automation Letters*, 8(11):7663–7670, Nov. 2023. ISSN 2377-3774. doi:10.1109/LRA.2023.3318190. URL https://doi.org/10.1109/LRA.2023.3318190.

[14] H. Xue, E. L. Zhu, J. M. Dolan, and F. Borrelli. Learning model predictive control with error dynamics regression for autonomous racing. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13250–13256. IEEE, May 2024. doi:10.1109/ICRA57147.2024.10611628. URL https://doi.org/10.1109/ICRA57147.2024.10611628.

[15] D. Kalaria, Q. Lin, and J. M. Dolan. Adaptive planning and control with time-varying tire models for autonomous racing using extreme learning machine. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10443–10449. IEEE, May 2024. doi:10.1109/ICRA57147.2024.10610848. URL https://doi.org/10.1109/ICRA57147.2024.10610848.

[16] W. Yu, J. Tan, C. K. Liu, and G. Turk. Preparing for the unknown: Learning a universal policy with online system identification. In *Proceedings of Robotics: Science and Systems*, Cambridge, Massachusetts, July 2017. doi:10.15607/RSS.2017.XIII.048. URL https://roboticsproceedings.org/rss13/p48.html.

[17] A. Kumar, Z. Fu, D. Pathak, and J. Malik. RMA: Rapid motor adaptation for legged robots. In *Proceedings of Robotics: Science and Systems*, Virtual, July 2021. doi:10.15607/RSS.2021.XVII.011. URL https://roboticsproceedings.org/rss17/p011.html.

[18] M. O'Connell, G. Shi, X. Shi, K. Azizzadenesheli, A. Anandkumar, Y. Yue, and S.-J. Chung. Neural-Fly enables rapid learning for agile flight in strong winds. *Science Robotics*, 7(66): eabm6597, May 2022. ISSN 2470-9476. doi:10.1126/scirobotics.abm6597. URL https://doi.org/10.1126/scirobotics.abm6597.

[19] K. Huang, R. Rana, A. Spitzer, G. Shi, and B. Boots. DATT: Deep adaptive trajectory tracking for quadrotor control. In J. Tan, M. Toussaint, and K. Darvish, editors, *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pages 326–340. PMLR, 06–09 Nov 2023. URL https://proceedings.mlr.press/v229/huang23a.html.

[20] J. Levy, J. Gibson, B. Vlahov, E. Tevere, E. Theodorou, D. Fridovich-Keil, and P. Spieler. Meta-learning online dynamics model adaptation in off-road autonomous driving. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025. doi:10.15607/RSS.2025.XXI.139. URL https://www.roboticsproceedings.org/rss21/p139.html.

[21] Y. Tsuchiya, T. Balch, P. Drews, and G. Rosman. Online adaptation of learned vehicle dynamics model with meta-learning approach. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 802–809. IEEE, Oct. 2024. doi:10.1109/IROS58592.2024.10801427. URL https://doi.org/10.1109/IROS58592.2024.10801427.

[22] K. Lee, Y. Seo, S. Lee, H. Lee, and J. Shin. Context-aware dynamics model for generalization in model-based reinforcement learning. In H. Daumé, III and A. Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5757–5766. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/lee20g.html.

[23] S. J. Wang, H. Zhu, and A. 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. IEEE, May 2024. doi:10.1109/ICRA57147.2024.10611059. URL https://doi.org/10.1109/ICRA57147.2024.10611059.

[24] J. Węgrzynowski, P. Kicki, G. Czechmanowski, M. P. Krupka, and K. Walas. Beyond constant parameters: Hyper prediction models and HyperMPC. In J. Lim, S. Song, and H.-W. Park, editors, *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pages 4885–4907. PMLR, 27–30 Sep 2025. URL https://proceedings.mlr.press/v305/wegrzynowski25a.html.

[25] S. Zhu, A. Kimmel, K. E. Bekris, and A. Boularias. Fast model identification via physics engines for data-efficient policy search. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 3249–3256. International Joint Conferences on Artificial Intelligence Organization, July 2018. doi:10.24963/ijcai.2018/451. URL https://doi.org/10.24963/ijcai.2018/451.

[26] F. Ramos, R. C. Possas, and D. Fox. BayesSim: Adaptive domain randomization via probabilistic inference for robotics simulators. In *Proceedings of Robotics: Science and Systems*, Freiburg im Breisgau, Germany, June 2019. doi:10.15607/RSS.2019.XV.029. URL https://doi.org/10.15607/RSS.2019.XV.029.

[27] Y. Chebotar, A. Handa, V. Makoviychuk, M. Macklin, J. Issac, N. Ratliff, and D. Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8973–8979. IEEE, May 2019. doi:10.1109/ICRA.2019.8793789. URL https://doi.org/10.1109/ICRA.2019.8793789.

[28] F. Muratore, C. Eilers, M. Gienger, and J. Peters. Data-efficient domain randomization with Bayesian optimization. *IEEE Robotics and Automation Letters*, 6(2):911–918, Apr. 2021. ISSN 2377-3774. doi:10.1109/LRA.2021.3052391. URL https://doi.org/10.1109/LRA.2021.3052391.

[29] O. Dikici, E. Ghignone, C. Hu, N. Baumann, L. Xie, A. Carron, M. Magno, and M. Corno. Learning-based on-track system identification for scaled autonomous racing in under a minute. *IEEE Robotics and Automation Letters*, 10(2):1984–1991, Feb. 2025. ISSN 2377-3774. doi:10.1109/LRA.2025.3527336. URL https://doi.org/10.1109/LRA.2025.3527336.

[30] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. Courville. FiLM: Visual reasoning with a general conditioning layer. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1):3942–3951, Apr. 2018. ISSN 2159-5399. doi:10.1609/aaai.v32i1.11671. URL https://doi.org/10.1609/aaai.v32i1.11671.

[31] S. Zhao, H. Zhang, P. Wang, L. Nogueira, and S. Scherer. Super odometry: IMU-centric LiDAR-visual-inertial estimator for challenging environments. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8729–8736. IEEE, Sept. 2021. doi:10.1109/IROS51168.2021.9635862. URL https://doi.org/10.1109/IROS51168.2021.9635862.

[32] R. Rana, J. Quattrociocchi, D. Lee, C. Ellis, A. Adkins, A. Uccello, G. Warnell, and J. Biswas. OVerSeeC: Open-vocabulary costmap generation from satellite images and natural language. In *2026 IEEE International Conference on Robotics and Automation (ICRA)*, pages 17233–17240. IEEE, 2026. doi:10.1109/ICRA57385.2026.11696225. URL https://doi.org/10.1109/ICRA57385.2026.11696225.

A Architecture and Training Details

Context extraction and FiLM conditioning. The history spans indices $t - M : t$, including the current step. Each historical state feature contains a position delta and relative quaternion expressed with respect to the preceding pose, together with linear and angular velocities in the instantaneous body frame (13D total). Actions are 2D, and current-state features comprise world-frame orientation and linear/angular velocities (10D), excluding absolute position. Separate MLPs embed these streams into $d = 64$ features; the historical state and action token sequences and the single current-state token are concatenated along the sequence dimension, with learned positional encodings added to form $\mathbf{M}_t$. A learnable summary token `[CTX]` is prepended before the bidirectional transformer encoder g_ϕ. Candidate future actions use the same action MLP and, with future positional encodings, form the decoder input. Context extraction is given by

$$\Big[\, \mathbf{c}_t;\ \tilde{\mathbf{M}}_t \,\Big] = \mathrm{Enc}_\phi([\,\texttt{[CTX]};\ \mathbf{M}_t\,])\,. \tag{7}$$

Only the summary output $\mathbf{c}_t$ is passed to the rollout decoder; the refined per-step tokens $\tilde{\mathbf{M}}_t$ are discarded. Each decoder block applies Feature-wise Linear Modulation at two sites:

$$\tilde{\mathbf{h}}_j^{(l)} = \boldsymbol{\gamma}_j^{(l)}(\mathbf{c}_t) \odot \mathbf{h}_j^{(l)} + \boldsymbol{\beta}_j^{(l)}(\mathbf{c}_t), \qquad j \in \{\mathrm{attn}, \mathrm{ffn}\}, \tag{8}$$

where $\mathbf{h}_j^{(l)}$ denotes decoder features within block l after self-attention ($j = \mathrm{attn}$) or after the feed-forward sublayer at the block output ($j = \mathrm{ffn}$). The two sites use separate learned affine projections $\boldsymbol{\gamma}_j^{(l)}, \boldsymbol{\beta}_j^{(l)} : \mathbb{R}^d \to \mathbb{R}^d$, initialized so that $\boldsymbol{\gamma}_j^{(l)} \approx \mathbf{1}$ and $\boldsymbol{\beta}_j^{(l)} \approx \mathbf{0}$.

Body-frame rollout integration. The decoder head outputs body-frame pose increments $\Delta\hat{\mathbf{s}}_{t+k} = (\Delta\hat{\mathbf{p}}_{t+k}, \Delta\hat{\mathbf{q}}_{t+k})$. The rotational output $\Delta\hat{\mathbf{q}}_{t+k}$ is L_2-normalized to a unit quaternion before rollout integration and geodesic loss evaluation. The rollout used by MPC is recovered by integrating these increments from the current pose, taken as the local origin $(\hat{\mathbf{p}}_t, \hat{\mathbf{q}}_t) = (\mathbf{0}, \mathbf{q}_{\mathrm{id}})$:

$$\hat{\mathbf{q}}_{t+k} = \hat{\mathbf{q}}_{t+k-1} \otimes \Delta\hat{\mathbf{q}}_{t+k}, \qquad \hat{\mathbf{p}}_{t+k} = \hat{\mathbf{p}}_{t+k-1} + \mathbf{R}(\hat{\mathbf{q}}_{t+k-1})\,\Delta\hat{\mathbf{p}}_{t+k}, \tag{9}$$

for $k = 1, \ldots, N$, where $\mathbf{R}(\cdot)$ is the rotation matrix of a unit quaternion, $\otimes$ denotes quaternion multiplication, and $\mathbf{q}_{\mathrm{id}}$ is the identity quaternion. Linear and angular velocities are recovered from the predicted pose sequence using finite differences with the rollout timestep $\Delta t = 0.02$ s; the resulting predicted velocity is used in the MPC cost.

Training loss. The pretraining and fine-tuning loss combines position MSE with a geodesic quaternion loss:

$$\mathcal{L} = \lambda_p \sum_{k=1}^{N} \left\| \Delta\hat{\mathbf{p}}_{t+k} - \Delta\mathbf{p}_{t+k} \right\|^2 + \lambda_q \sum_{k=1}^{N} d_{\mathrm{geo}}(\Delta\hat{\mathbf{q}}_{t+k},\, \Delta\mathbf{q}_{t+k})\,, \tag{10}$$

with $\lambda_p = 1.0$ and $\lambda_q = 0.3$. The geodesic quaternion distance is

$$d_{\mathrm{geo}}(\mathbf{q}_1, \mathbf{q}_2) = 2\,\mathrm{arctan2}\big(\big\|\mathrm{Im}(\mathbf{q}_1^{-1} \otimes \mathbf{q}_2)\big\|\,,\ \big|\mathrm{Re}(\mathbf{q}_1^{-1} \otimes \mathbf{q}_2)\big|\big)\,, \tag{11}$$

which is invariant to the double-cover ambiguity $\mathbf{q} \sim -\mathbf{q}$.

B Reference Trajectory

The figure-8 reference trajectory is parameterized by $\theta \in [-\pi, \pi]$ as

$$x(\theta) = \frac{A\cos\theta}{1 + \sin^2\theta}, \qquad y(\theta) = \frac{wA\sin\theta\cos\theta}{1 + \sin^2\theta}, \tag{12}$$

with width factor $w = 1.5$ and heading given by the tangent direction $\psi(\theta) = \mathrm{atan2}(dy/d\theta, dx/d\theta)$. We use amplitude $A = 8.3$ for the road trajectory, giving an approximately 17 m horizontal span, and $A = 15$ for the lower-traction terrain trajectories, giving a 30 m horizontal span.

C Dynamic Bicycle Model with Wheel-Velocity Actuation

Each per-terrain simulator F_{ψ_e} in Section 4.3 is a modified dynamic bicycle model (DBM) with driven wheels, in which throttle is a desired wheel-edge linear velocity command d_w with a first-order wheel-speed state v_w [1]. The model is planar, with yaw-only rotation represented internally and exported as a quaternion. Table 2 lists the parameters; those marked ID form the per-terrain vector ψ_e identified by nonlinear least squares on real one-step transitions (Eq. 4).

Symbol	Description	Src.
m	Vehicle mass	Meas.
I	Yaw inertia ($\frac{1}{12}m(l^2+b^2)$)	Approx.
l_f, l_r	COM to front/rear axle distance	Assumed
$\psi_{\max}$	Maximum steering angle	Limit
$v_{w,\min}$, $v_{w,\max}$	Min/max wheel linear velocity	Limit
T_w	Wheel-speed time constant	ID
C_a	Drive force constant	ID
$C_{t,f}$, $C_{t,r}$	Front/rear tire cornering stiffness	ID
C_r	Rolling resistance coefficient	ID
C_d	Quadratic drag coefficient	ID

Table 2: Dynamic bicycle model parameters. In the inertia approximation, l and b denote vehicle length and width, respectively, distinct from the COM-to-axle distances l_f and l_r. **Src.** denotes the parameter source: *Meas.* = directly measured from the platform; *Approx.* = approximated from vehicle geometry; *Assumed* = assumed by model symmetry; *Limit* = platform-imposed limit; *ID* = identified by least-squares system identification on real-world data.

State and inputs. The DBM evolves the planar state

$$x = [p_x \quad p_y \quad \theta \quad v_{\text{long}} \quad v_{\text{lat}} \quad r \quad v_w]^\top, \tag{13}$$

where (p_x, p_y) is planar position, θ is yaw, $(v_{\text{long}}, v_{\text{lat}})$ are body-frame velocities, r is the yaw rate, and v_w is the wheel-edge linear velocity. The control input is $u = [\, d_w \quad \psi \,]^\top$, clamped to physical limits,

$$\tilde{d}_w = \text{clip}(d_w, v_{w,\min}, v_{w,\max}), \qquad \tilde{\psi} = \text{clip}(\psi, -\psi_{\max}, \psi_{\max}). \tag{14}$$

For compatibility with the learned FKD model, the planar state is exported in the same 13D pose-velocity format used throughout the paper, with the out-of-plane components set to zero and yaw encoded as a quaternion.

Dynamics. Planar kinematics are

$$\dot{p}_x = v_{\text{long}} \cos\theta - v_{\text{lat}} \sin\theta, \tag{15}$$
$$\dot{p}_y = v_{\text{long}} \sin\theta + v_{\text{lat}} \cos\theta, \tag{16}$$
$$\dot{\theta} = r. \tag{17}$$

Front and rear slip angles are

$$\alpha_f = \tilde{\psi} - \text{atan2}(v_{\text{lat}} + r l_f,\ v_{\text{long}}), \tag{18}$$
$$\alpha_r = -\text{atan2}(v_{\text{lat}} - r l_r,\ |v_{\text{long}}|). \tag{19}$$

Drive, tire, and resistive forces are

$$F_f^{\text{drive}} = F_r^{\text{drive}} = \tfrac{1}{2} C_a (v_w - v_{\text{long}}), \tag{20}$$
$$F_f^{\text{tire}} = C_{t,f}\, \alpha_f, \qquad F_r^{\text{tire}} = C_{t,r}\, \alpha_r, \tag{21}$$
$$F_{\text{roll}} = C_r \,\text{sign}(v_{\text{long}}), \qquad F_{\text{drag}} = C_d\, v_{\text{long}} |v_{\text{long}}|. \tag{22}$$

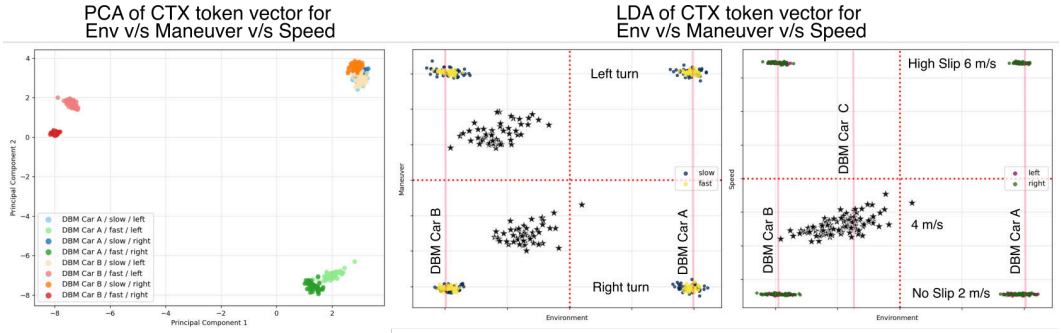


Figure 7: **Context-vector representation analysis.** Context vectors $\mathbf{c}$ extracted during simulated closed-loop operation are projected to two dimensions. *Left:* PCA gives an unsupervised view of the embedding and separates the dominant dynamics regimes across vehicles, speeds, and turning directions. *Middle:* LDA using maneuver labels separates left- and right-turn histories while preserving vehicle-specific structure along the environment axis. *Right:* LDA using speed labels separates low-slip 2 m/s, moderate 4 m/s, and high-slip 6 m/s regimes, showing that the learned context encodes speed-dependent slip effects.

The net body-frame forces and yaw moment are

$$F_{\text{long}} = F_f^{\text{drive}} \cos\tilde{\psi} + F_r^{\text{drive}} - F_f^{\text{tire}} \sin\tilde{\psi} - F_{\text{roll}} - F_{\text{drag}}, \tag{23}$$

$$F_{\text{lat}} = F_f^{\text{drive}} \sin\tilde{\psi} + F_f^{\text{tire}} \cos\tilde{\psi} + F_r^{\text{tire}}, \tag{24}$$

$$\tau_z = F_f^{\text{drive}} l_f \sin\tilde{\psi} + F_f^{\text{tire}} l_f \cos\tilde{\psi} - F_r^{\text{tire}} l_r, \tag{25}$$

yielding the body-frame and wheel-speed dynamics

$$\dot{v}_{\text{long}} = \tfrac{1}{m} F_{\text{long}} + v_{\text{lat}} r, \qquad \dot{v}_{\text{lat}} = \tfrac{1}{m} F_{\text{lat}} - v_{\text{long}} r, \tag{26}$$

$$\dot{r} = \tfrac{1}{I} \tau_z, \qquad \dot{v}_w = \frac{\tilde{d}_w - v_w}{T_w}. \tag{27}$$

The continuous-time system is integrated with RK4 to form the discrete transition F_{ψ_e} used for system identification and synthetic rollout generation.

D Inverse Kinodynamic (IKD) Baseline

The **IKD FT-R** baseline mirrors `OptCar`'s history-conditioned transformer, including the history encoder, the `[CTX]` summary token, and FiLM conditioning, but reverses the input-output direction of prediction. Instead of predicting future states from a candidate action sequence, IKD takes a sequence of desired future states $\mathbf{s}^{\text{des}}_{t+1:t+N}$ and predicts the actions that realize them,

$$\hat{\mathbf{a}}_{t:t+N-1} = h_\xi\left(\mathbf{s}_{t-M:t},\ \mathbf{a}_{t-M:t},\ \mathbf{s}^{\text{des}}_{t+1:t+N}\right), \tag{28}$$

where the decoder consumes the candidate future-state stream and emits per-step throttle and steering commands. The desired future states are not free variables; they are produced by a separate planner that rolls out the per-terrain DBM $F_{\psi_e^*}$ (Appendix C) under MPPI to obtain a reference trajectory, which IKD then inverts into commands. Because execution depends on the planner's analytical model rather than on scoring action rollouts with the deployed learned dynamics, tracking accuracy is bounded by the validity of the DBM, which degrades in the high-slip regime and explains the sharp error growth of IKD FT-R at 6 m/s in Table 1.

E Context-Vector Representation Analysis

We analyze the learned context vectors $\mathbf{c}_t$ extracted during simulated closed-loop operation across vehicles, speeds, and turning maneuvers by projecting them with PCA and LDA (Figure 7). In the

unsupervised PCA projection, contexts cluster by dominant dynamics regime, including vehicle, speed, and turning direction. LDA projections using maneuver and speed labels show that these factors are also linearly recoverable from the embedding. Thus the context vector is not merely a trajectory identifier; it organizes recent history around factors that affect rollout prediction for MPC.

F Closed-Loop Tracking Results with Standard Deviations

Table 3 reports means and standard deviations of per-timestep tracking errors pooled across three runs for all methods.

		Easy (2 m/s)		Medium (4 m/s)		Hard (6 m/s)	
Env.	**Method**	TTE	VTE	TTE	VTE	TTE	VTE
Road	DBM	**0.011** $\pm$ 0.003	**0.004** $\pm$ 0.001	0.034 $\pm$ 0.007	**0.019** $\pm$ 0.004	0.562 $\pm$ 0.070	1.234 $\pm$ 0.281
	IKD FT-R	0.016 $\pm$ 0.003	0.032 $\pm$ 0.007	0.067 $\pm$ 0.014	0.189 $\pm$ 0.040	0.412 $\pm$ 0.052	1.849 $\pm$ 0.423
	AnyCar FT-R	0.013 $\pm$ 0.002	0.009 $\pm$ 0.002	0.025 $\pm$ 0.005	0.025 $\pm$ 0.005	0.248 $\pm$ 0.063	0.912 $\pm$ 0.197
	AnyCar FT-RS	0.014 $\pm$ 0.003	0.008 $\pm$ 0.002	0.028 $\pm$ 0.006	0.022 $\pm$ 0.005	0.152 $\pm$ 0.041	0.882 $\pm$ 0.192
	Concat FT-RS	0.016 $\pm$ 0.003	0.009 $\pm$ 0.002	0.027 $\pm$ 0.005	0.035 $\pm$ 0.008	0.176 $\pm$ 0.064	1.002 $\pm$ 0.230
	`OptCar` Specialist	0.012 $\pm$ 0.002	0.007 $\pm$ 0.002	**0.020** $\pm$ 0.004	0.020 $\pm$ 0.004	**0.093** $\pm$ 0.034	**0.711** $\pm$ 0.161
	`OptCar` FT-R	0.018 $\pm$ 0.004	**0.004** $\pm$ 0.001	0.034 $\pm$ 0.007	0.027 $\pm$ 0.006	0.196 $\pm$ 0.041	0.911 $\pm$ 0.209
	`OptCar` FT-RS	0.016 $\pm$ 0.003	0.013 $\pm$ 0.003	0.026 $\pm$ 0.005	0.023 $\pm$ 0.005	0.106 $\pm$ 0.042	0.901 $\pm$ 0.203
Wet Grass + Slope	DBM	0.012 $\pm$ 0.002	0.017 $\pm$ 0.004	0.156 $\pm$ 0.029	0.044 $\pm$ 0.010	1.356 $\pm$ 0.182	1.456 $\pm$ 0.330
	IKD FT-R	0.013 $\pm$ 0.003	0.029 $\pm$ 0.006	0.062 $\pm$ 0.012	0.255 $\pm$ 0.055	2.371 $\pm$ 0.421	1.004 $\pm$ 0.234
	AnyCar FT-R	**0.011** $\pm$ 0.002	0.025 $\pm$ 0.005	0.100 $\pm$ 0.019	0.083 $\pm$ 0.018	1.082 $\pm$ 0.173	**0.892** $\pm$ 0.202
	AnyCar FT-RS	0.014 $\pm$ 0.003	0.022 $\pm$ 0.005	0.052 $\pm$ 0.010	0.049 $\pm$ 0.011	0.771 $\pm$ 0.151	0.922 $\pm$ 0.215
	Concat FT-RS	0.015 $\pm$ 0.003	0.026 $\pm$ 0.006	0.058 $\pm$ 0.011	0.067 $\pm$ 0.015	0.785 $\pm$ 0.179	1.083 $\pm$ 0.250
	`OptCar` Specialist	0.013 $\pm$ 0.003	**0.016** $\pm$ 0.004	0.146 $\pm$ 0.028	0.051 $\pm$ 0.011	1.149 $\pm$ 0.211	1.056 $\pm$ 0.240
	`OptCar` FT-R	0.013 $\pm$ 0.003	0.027 $\pm$ 0.006	**0.031** $\pm$ 0.006	0.078 $\pm$ 0.017	0.751 $\pm$ 0.146	0.933 $\pm$ 0.207
	`OptCar` FT-RS	0.012 $\pm$ 0.002	0.025 $\pm$ 0.005	0.044 $\pm$ 0.009	**0.011** $\pm$ 0.003	**0.625** $\pm$ 0.125	0.940 $\pm$ 0.210
Vegetation + Dirt	DBM	0.061 $\pm$ 0.011	0.041 $\pm$ 0.009	0.276 $\pm$ 0.054	0.312 $\pm$ 0.070	0.824 $\pm$ 0.134	1.118 $\pm$ 0.255
	IKD FT-R	0.055 $\pm$ 0.010	0.033 $\pm$ 0.007	0.232 $\pm$ 0.045	0.452 $\pm$ 0.100	0.669 $\pm$ 0.127	1.452 $\pm$ 0.336
	AnyCar FT-R	**0.054** $\pm$ 0.010	**0.019** $\pm$ 0.004	0.207 $\pm$ 0.039	0.199 $\pm$ 0.045	0.776 $\pm$ 0.144	0.991 $\pm$ 0.221
	`OptCar` FT-R	0.055 $\pm$ 0.011	0.020 $\pm$ 0.004	0.171 $\pm$ 0.033	0.172 $\pm$ 0.040	0.519 $\pm$ 0.108	0.981 $\pm$ 0.220
	`OptCar` FT-RS	0.059 $\pm$ 0.012	0.023 $\pm$ 0.005	**0.157** $\pm$ 0.030	**0.072** $\pm$ 0.016	**0.348** $\pm$ 0.072	**0.937** $\pm$ 0.224
Cart Pulling	DBM	0.129 $\pm$ 0.025	0.014 $\pm$ 0.003	0.212 $\pm$ 0.041	0.197 $\pm$ 0.045	1.012 $\pm$ 0.163	1.881 $\pm$ 0.427
	IKD FT-R	0.067 $\pm$ 0.013	0.086 $\pm$ 0.019	0.244 $\pm$ 0.047	0.102 $\pm$ 0.023	0.711 $\pm$ 0.133	1.782 $\pm$ 0.403
	AnyCar FT-R	0.042 $\pm$ 0.008	0.031 $\pm$ 0.007	0.182 $\pm$ 0.035	0.049 $\pm$ 0.011	0.489 $\pm$ 0.095	1.338 $\pm$ 0.299
	`OptCar` FT-R	0.028 $\pm$ 0.006	0.028 $\pm$ 0.006	0.174 $\pm$ 0.034	0.062 $\pm$ 0.014	0.319 $\pm$ 0.062	1.290 $\pm$ 0.294
	`OptCar` FT-RS	**0.011** $\pm$ 0.002	**0.014** $\pm$ 0.003	**0.085** $\pm$ 0.017	**0.044** $\pm$ 0.010	**0.229** $\pm$ 0.054	**1.023** $\pm$ 0.232

Table 3: **Closed-loop figure-8 tracking with standard deviations.** Expanded version of Table 1, reporting TTE (m $\downarrow$) and VTE (m/s $\downarrow$) across terrains, speeds, and the unseen cart-pulling payload. Values report mean $\pm$ standard deviation of per-timestep tracking errors pooled across three runs.