Learning Diffusion Planners from World Feedback: A No-Go Result on Bit-Exact Safety Rewards and an ODD-Adaptive Shared/Expert Decomposition

Anonymous Author(s)
Affiliation
Address
email

Abstract

We fine-tune diffusion-based trajectory planners using closed-loop safety, efficiency, comfort, and real-time deployment outcomes as reward, learning from world feedback rather than human preferences. Across 8 operational design domains (ODDs) in two independent stacks (Autoware + nuPlan, different sensors and planner backbones), world-grounded fine-tuning yields closed-loop gains within seed-to-seed noise but reproducibly exposes a **Shared-Expert LoRA (SE-LoRA)** structure: the reward gradient splits into a cross-ODD aligned subspace (geometric priors) and ODD-specific subspaces (denoising dynamics). Three independent signatures support the split: a Singapore↔Pittsburgh double dissociation under causal activation patching ($\rho{=}-0.96$), a $10\times$ gradient-cosine gap between components, and a cross-ODD PCA spectral gap that predicts a $1{:}2$ shared-to-expert rank ratio, confirmed by rank-allocation sweeps on both stacks. The same lens also produces a **no-go result** relevant to any world-grounded reward with hard safety thresholds: **PCDR** (Per-Closed-Loop Differentiable Reward), to our knowledge the first bit-exact differentiable mirror of nuPlan closed-loop scoring, exposes a forward-fidelity vs. gradient-reach trade-off for $0/1$ safety rewards, motivating a constrained-recovery design (smooth CLS proxy $+$ bit-exact PCDR-derived margin penalty). We release PCDR, SE-LoRA, and anonymized real-vehicle deployment data (uncommon in the diffusion-planning literature) covering both simulation and a production ROS 2 stack. World-grounded rewards reveal structural decompositions that BC-only training hides and impose hard limits on $0/1$ safety signals; the spectral-gap recipe predicts LoRA ranks without sweep, and PCDR provides a standalone audit of reward↔metric alignment.

1 Introduction

For embodied agents, the available training signal is the environment rather than a human ranker: closed-loop safety, distance to obstacles, real-time deadline compliance, ride comfort, and route progress. Autonomous driving provides a well-defined version of this signal through closed-loop simulators and on-vehicle deployment Karnchanachari et al. [2024], The Autoware Foundation [2025]. Diffusion-based motion planners Ho et al. [2020], Chi et al. [2023], Zheng et al. [2025], Liao et al. [2025], Xing et al. [2025] are state-of-the-art on open-loop benchmarks but trained exclusively via behavior cloning (BC), so they have never seen any world-grounded reward at training time.

Recent works apply RL fine-tuning to diffusion policies Black et al. [2023], Fan et al. [2023], Ren et al. [2025], Zou et al. [2025], Li et al. [2026], but rely on stochastic samplers (DDPM Ho et al. [2020]/DDIM) during training while deploying deterministic ODE solvers, and optimize hand-crafted soft proxies that drift from the deployment metric. Both Autoware The Autoware Foundation [2025]

Submitted to RLxF Workshop @ ICML 2026 (Reinforcement Learning from World Feedback). Do not distribute.

closed-loop and nuPlan val14 evaluation show that naive world-feedback fine-tuning frequently fails to move the deployment metric. This structural phenomenon admits two complementary analyses: a mechanistic decomposition (SE-LoRA, below) and a no-go result on $0/1$ safety rewards (Lemma 1) that constrains which world-feedback rewards can train a diffusion planner.

Our central contribution is an intervention-supported Shared-Expert LoRA (SE-LoRA, §3.3) decomposition: a shared rank-4 LoRA captures cross-ODD geometric priors while per-ODD rank-8 experts specialize denoising dynamics. Three independent empirical signatures support this structure. **(I)** Causal activation patching gives a double dissociation between Singapore- and Pittsburgh-styled experts ($\rho{=}-0.96$ on signed signatures, §4.7). **(II)** Shared and expert subspaces differ by a $10\times$ gradient-cosine gap (§4.5). **(III)** The PCA spectral gap *predicts* a $1{:}2$ shared-to-expert rank ratio, confirmed by rank-allocation sweeps on *both* Autoware and nuPlan placing $1{:}2$ at the empirical aggregate-reward peak (§4.5, Appendix S23). The decomposition aligns with MoE specialization Jacobs et al. [1991] and multi-task gradient-conflict Ruder [2017] but is empirically tested at the activation-patching and spectral-gap levels rather than asserted.

The analytical pipeline that surfaces this structure also yields diagnostic byproducts. Under matched 3-seed evaluation on full val14, all fine-tuning configurations are statistically equivalent to BC (soft CLS-structured proxy 0.8975 ± 0.0013; constrained-recovery design 0.8964 ± 0.0013, denoted PCDR-CR in tables); on the same val14, matched DPM-Solver++ vs. mismatched DDIM passes a pre-specified equivalence test (paired ± 0.005, $p{=}0.041$, $n{=}1133$), repositioning our differentiable DPM-Solver++ unrolling (§3.1) as a per-scene vulnerability diagnostic ($r{=}-0.91$, §4.4) and training-path enabler rather than a benchmark-gain mechanism. A no-go lemma on the forward-fidelity vs. gradient-reach trade-off (Lemma 1) explains why bit-exact $0/1$ safety rewards have a.e. zero pathwise gradient and require a constrained-recovery design (smooth proxy main reward $+$ bit-exact PCDR-derived margin penalty), which uniquely recovers PCDR-3's catastrophic regression to BC-equivalent level. A cross-stack real-vehicle study (§4.7), uncommon in the diffusion-planning literature, extends these results to a production ROS 2 stack: the unrolled solver runs in real time and the sim-to-real divergence localizes to loop-closure mode rather than a model-side shift.

2 Related Work

RL Fine-Tuning for Diffusion Models. DPPO Ren et al. [2025], DiffusionDriveV2 Zou et al. [2025], PlannerRFT Li et al. [2026], and Plan-R1 Tang et al. [2025] use stochastic samplers for log-probability computation during training while deploying deterministic solvers; DRaFT Clark et al. [2024] backpropagates through DDIM in image generation. We extend the pathwise route to DPM-Solver++ for closed-loop AD. LaSRO Jia et al. [2025] reports that hard-reward fine-tuning of distilled diffusion models fails and a learned differentiable surrogate recovers, the closest published precedent for our audit-vs-trainability separation.

Differentiable vs. distilled driving metrics. Hydra-MDP/Hydra-MDP++ Li et al. [2024b, 2025] are the closest AD neighbors: a multi-head transformer *distills* NAVSIM PDMS sub-scores into per-trajectory classification heads (91.0 PDMS), a learned surrogate with nonzero forward residual error and a fixed candidate-trajectory vocabulary. Our PCDR audit (Appendix S28) is structurally different: operator-level forward equivalence on the nuPlan closed-loop sub-metrics (NC bit-exact vs. Shapely on $52{,}696$ cells; EP/DDC at machine precision; sub-metric definitions in Appendix S28), supporting any continuous trajectory output, and used as a training reward / deployment audit rather than a distillation target. Diffusion-ES Yang et al. [2024] motivates evolutionary search by the non-differentiability of the metric, the canonical statement of the gap PCDR closes. The paper is positioned as a mechanistic study, not a leaderboard push (full positioning vs. rule-based SOTA in Appendix S2).

Mixture of Experts and LoRA; multi-task background. MoLA Gao et al. [2024], MixLoRA Li et al. [2024a], and MoLoRA Shah and Wagle [2026] adapt MoE to LoRA with learned per-token routing; SE-LoRA differs in three SE-specific aspects: *per-scene* (not per-token) deterministic routing from the HD-map identifier, an explicit shared+expert split $\mathbf{W} = \mathbf{W}_0 + \mathbf{B}_s\mathbf{A}_s + \mathbf{B}_{e_m}\mathbf{A}_{e_m}$ rather than a single per-token adapter, and an activation-patching dissociation between the two components. The shared/private decomposition has classical roots in multi-task learning James and Stein [1961], Daumé [2007], Baxter [2000], Ando and Zhang [2005]; our novelty is showing that the split arises

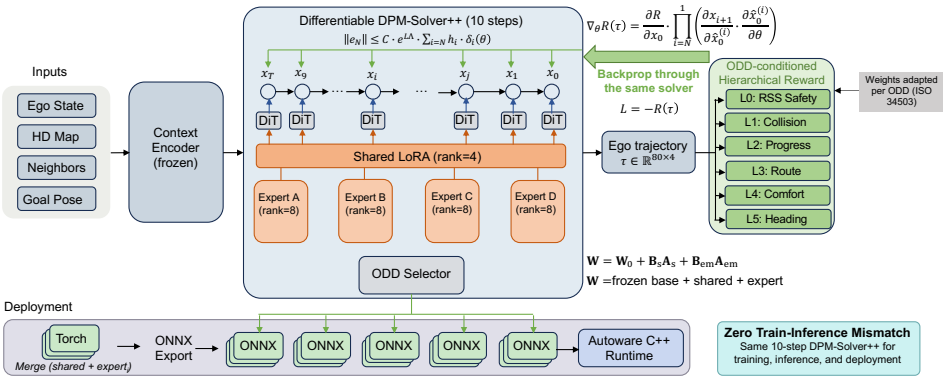

Figure 1: SE-LoRA + differentiable DPM-Solver++ pipeline. The same 10-step solver runs in training, inference, and deployment; reward gradient flows back through the unrolled solver under the Gronwall bound (Remark 1).

under multi-ODD reward optimization and that its two components carry distinct, intervention-testable semantics (geometry vs. dynamics). Backward error analysis Hairer et al. [2006] grounds the differentiable DPM-Solver++ as an instance of deep unrolling Gregor and LeCun [2010], Monga et al. [2021]. Detailed lineage and the RSS-grounded safety formulation The Autoware Foundation [2025] are in Appendix S2.

3 Method

3.1 Differentiable DPM-Solver++

REINFORCE-style training of diffusion policies requires $\nabla_\theta \log p_\theta$, intractable for deterministic ODE solvers; existing solutions inject stochastic sampling (DDPM/DDIM with $\eta{>}0$) at the cost of train–inference mismatch, and we replicate the resulting gradient explosion under stochastic DDIM ($\eta{=}0.3$) in Appendix S18. We instead unroll the deployment solver itself: DPM-Solver++ is a deterministic, differentiable function $f_\theta : \boldsymbol{\epsilon} \mapsto \mathbf{x}_0$ (Figure 1), so a differentiable reward backpropagates through the 10-step graph as a single reparameterization-style estimator with $O(1/B)$ pathwise variance vs. $O(N/B)$ for score-function alternatives. The full unrolled backpropagation-through-time (BPTT) chain rule, the inter-implementation discrepancy ϵ_0, and the discrete Gronwall amplification (Remark 1: $\epsilon_0{\sim}10^{-3}$ amplifies to $\sim$350 trajectory units over $N{=}10$, far above the anytime threshold $\varepsilon{=}0.1$ m, so matched-solver training eliminates the gap by construction) are deferred to Appendices S4–S7.

3.2 LR Sweep: Motivation for Per-ODD Adaptation

A learning-rate sweep across 4 ODDs (Appendix S11, Table 5) shows no single LR universally improves all maps: the dense-urban Map A favours $3{\times}10^{-6}$ (6/8 metrics improved), suburban Map B and out-of-distribution Map C favour $5{\times}10^{-6}$ (4/8 each), and campus Map D favours $8{\times}10^{-6}$ (4/8). This conflict motivates SE-LoRA: rather than averaging gradients across ODDs, each expert adapts independently.

3.3 Shared-Expert LoRA (SE-LoRA)

We decompose the LoRA Hu et al. [2021] adaptation into shared and expert components:

$$\mathbf{W} = \mathbf{W}_0 + \underbrace{\mathbf{B}_s\mathbf{A}_s}_{\text{shared, rank-}r_s} + \underbrace{\mathbf{B}_{e_m}\mathbf{A}_{e_m}}_{\text{expert}_m\text{, rank-}r_e}, \tag{1}$$

where $\mathbf{W}_0$ is the frozen base, $r_s = 4$ captures cross-ODD geometric priors, and $r_e = 8$ per expert captures ODD-specific denoising dynamics (theoretical motivation in Appendix S8). With 4 experts (one per map), the total trainable parameters are 862K, 1.5% of the 58M base model.

During training, ODD-stratified mini-batch sampling ensures balanced map representation: each batch contains one sample per map. Each sample's reward is computed with the corresponding ODD weight profile $\mathbf{w}_m$ (Table 6 in Appendix). At deployment, expert selection is *deterministic*: the HD map identifier selects the expert, requiring no learned router. The merged weights $\mathbf{W}_0 + \mathbf{B}_s\mathbf{A}_s + \mathbf{B}_{e_m}\mathbf{A}_{e_m}$ are exported per-map as standard ONNX models.

3.4 ODD-Conditioned RSS Reward

We design a hierarchical reward $R_m = r_{\text{rss}} + \mathbf{w}_m^\top [r_{\text{collision}}, r_{\text{route}}, r_{\text{curve}}, r_{\text{progress}}, r_{\text{comfort}}, r_{\text{heading}}]$ covering safety, efficiency, and utility. The L0 RSS term follows the Autoware E2E recommendation The Autoware Foundation [2025] to maintain reachable safe states: $r_{\text{rss}} = -w_{\text{rss}} \sum_t \max(0,\, v_t^2/(2a_{\max}) - (d_t^{\text{nb}} - d_{\text{safe}}))$ with $a_{\max} = 5.0\,\text{m/s}^2$, $d_{\text{safe}} = 2.0\,\text{m}$, plus a route-corridor variant. L1–L4 components are collision-penetration penalty (d_{col}=3 m), curvature-weighted route deviation, capped path length, jerk + lat-acc comfort, and heading alignment. ODD-conditioned weights $\mathbf{w}_m$ follow the ISO 34503 International Organization for Standardization [2023] taxonomy: dense-urban Map A uses w_{route}=3.0, w_{curve}=4.0, w_{rss}=2.0; suburban Map B uses w_{progress}=3.0 (full table in Appendix S12). A per-metric conformance audit between this training reward and the nuPlan devkit scoring is in Appendix S28; we also document attempted convergence-aware reward shaping that did not transfer in Appendix S10.

4 Experiments

4.1 Setup

All experiments run on a desktop PC with a single consumer-grade GPU; the Diffusion Planner operates at 10 Hz with anytime scheduling Zilberstein [1996] (convergence threshold ε=0.1 m, deadline 100 ms, synthetic jitter $\mathcal{U}(20, 60)$ ms; full step-allocation analysis in Appendix S9). Four ODD maps are evaluated: Map A (dense urban campus, 122 lanelets), Map B (suburban, 194), Map C (OOD test track, 263), Map D (campus, 46); per map, the last 2 of 6 routes are held out for validation. Three maps have real-world HD lidar (Appendix S29, Figure 15). SE-LoRA (shared rank-4 + expert rank-8, 862K params) is trained for 10 epochs at lr= 2×10^{-6} on 6,196 scenes ($\sim$3 h on a single GPU; per-epoch dynamics in Appendix S13, code/data release in Appendix S6). For obstacle evaluation, a stationary vehicle is placed 25 m ahead on the road centerline via dual injection of PredictedObjects and TrackedObjects at 10 Hz (Appendix S15).

4.2 Empty Road Evaluation

On empty-road evaluation (3 maps, 4 seeds; full table in Appendix S14, Table 8), improvements are modest: Shared achieves statistically significant RSS-violation reduction ($p = 0.012$) while Expert shows a trend in lateral acceleration ($p = 0.071$). The small effect sizes are expected: without dynamic obstacles, the baseline already drives competently and the collision/RSS reward components are not exercised. The obstacle evaluation below tests the full reward function.

4.3 Obstacle Evaluation

To activate the collision and RSS reward components, we inject a stationary vehicle 25 m ahead on the road centerline (a single-scene illustration of ego speed near obstacles is in Appendix S15, Figure 9). Table 1 summarizes multi-seed obstacle metrics on Map A and Map B (maps where the obstacle is on the driven path). Expert reduces speed near obstacles ($p = 0.074$), while Shared achieves statistically significant jerk reduction ($p = 0.038$), consistent with the decomposition: Expert modulates driving dynamics, Shared improves geometric smoothness.

The multi-seed results support the expert/shared decomposition at the behavioral level: Expert is the only configuration that reduces speed near obstacles (a dynamic response), while Shared is the only configuration that significantly reduces jerk (a geometric smoothness improvement). The effect sizes are modest on this setup because the obstacle injection bypasses perception; incorporating obstacles into the reward function during training would likely amplify the behavioral differentiation.

Table 1: Obstacle evaluation (Map A + Map B, 4 seeds). **Bold** = best, underline = second-best per row. $^{\sim}p < 0.10$, $^{*}p < 0.05$ (paired t-test vs. baseline, per-seed means). Obstacles injected via PredictedObjects + TrackedObjects dual injection at 10 Hz.

Metric	Baseline	Expert	Shared
Speed (m/s)↓	5.27 ± 0.05	$\mathbf{5.11} \pm 0.10^{\sim}$	$\mathbf{5.11} \pm 0.10$
RSS Viol. (%)↓	$\mathbf{13.3} \pm 1.0$	13.7 ± 1.9	13.8 ± 1.0
Route Dev. (m)↓	$\mathbf{1.10} \pm 0.04$	1.11 ± 0.04	1.14 ± 0.06
Jerk (m/s^3)↓	40.3 ± 1.5	39.3 ± 3.2	$\mathbf{37.1} \pm 1.9^{*}$
Lat. Acc. (m/s^2)↓	0.84 ± 0.04	0.87 ± 0.07	$\mathbf{0.81} \pm 0.06$
Deadline (%)↓	51.2 ± 1.1	49.2 ± 1.1	$\mathbf{47.7} \pm 1.5$

4.4 Solver Sweep: Quantifying Train–Inference Mismatch

We train identical LoRA models with 4 solvers under matched reward/data/hparams, all evaluated under DPM-Solver++ 2nd order (the deployment solver). Table 2 (full per-epoch in Appendix S16) shows an inverted pattern: simpler solvers reach *higher* training reward yet *lower* deployment quality. Euler peaks training at 385 (epoch 5) but deploys worst; matched DPM-Solver++ is 5% behind in training reward but the only solver to reach *Best* at deployment; stochastic DDIM (η=0.3) diverges entirely (NaN gradients through noise samples).

Table 2: 4-solver sweep on Autoware (6,377 scenes), all evaluated under DPM-Solver++ 2nd order. Per-epoch training reward; deployment verdict is the closed-loop quality category.

Training Solver	Ep. 1	Ep. 2	Ep. 3	Ep. 4	Ep. 5	Deploy
DPM-Solver++ 2nd (matched)	143	342	361	364	367	**Best**
DDIM η=0	161	244	273	289	303	Poor
DDIM η=0.3	NaN (gradient explosion through noise samples)					Diverged
Euler	**318**	**377**	**383**	**385**	**385**	Worst

Generalization to nuPlan: closed-loop validation. To check whether this finding generalizes beyond Autoware, we repeat the experiment on 20,000 nuPlan scenes Karnchanachari et al. [2024] using the official Diffusion Planner checkpoint and evaluate with the official nuPlan closed-loop simulation (val14 split, 1,118 scenarios, non-reactive agents; per-epoch reward in Appendix S17). The matched-pair solver-equivalence subset (n=1,133 in Tables 3 and 24) augments the standard 1,118 with 15 duplicate-token rows that arise when the same scene is rolled out under two solvers; multi-seed reward-design rows (n=1,118) use the standard split. Table 3 reports a controlled fine-tuning comparison on val14 (n=1133 matched scenarios) across 6 identical-everything-except-reward configurations. There are two findings. **(a)** Under matched 3-seed evaluation on full val14 (n=1118), all fine-tuning configurations are statistically equivalent to BC (within seed-to-seed noise): the soft CLS-structured proxy (denoted *noc* for "no-comfort", a smooth surrogate without the comfort term) reaches 0.8975 ± 0.0013 (Δ=+0.0002 vs. BC 0.8973; Δcat=+1.0), and PCDR-CR reaches 0.8964 ± 0.0013 (Δ=−0.0009; Δcat=+2). However, PCDR-CR is the only configuration that recovers PCDR-3's catastrophic regression to BC-equivalent level ($4/4$ asymmetric PCDR-3 catastrophic failures fixed, 0 new); we report this co-occurrence with all three SE-LoRA mechanisms as a descriptive finding, not a causal-necessity claim. **(b)** Matched DPM-Solver++ and mismatched DDIM are statistically equivalent on aggregate (paired t-test p=0.549, McNemar p=1.0, 96% catastrophic overlap; pre-specified two-one-sided-test (TOST) equivalence in Table 24), so the solver-consistency benefit is not a benchmark-gain mechanism on this stack.

The matched analysis revises the standalone solver-consistency claim downward. Aggregate CLS between matched DPM-Solver++ and mismatched DDIM is statistically indistinguishable on val14 ($\Delta = +0.0013$, paired t-test p=0.549, n=1133); catastrophic failures overlap 96% between the two solvers. This stands in apparent contrast with the matched-vs-mismatched training-reward gap (Autoware 1.21× at §4.4, nuPlan 2.2× at epoch 5). We attribute the closed-loop equivalence to differences in how each evaluation stack propagates trajectory error: Autoware ROS2 closed-loop re-planning amplifies open-loop trajectory error multiplicatively (consistent with Gronwall, Remark 1), while nuPlan val14 scores rolled-out trajectories against pre-recorded agents at indicator-gate granularity, which is largely insensitive to small trajectory perturbations as long as no gate

Table 3: nuPlan val14 controlled fine-tuning. **Bold** = best, underline = second-best per column. Catastrophic = scenarios with CL score exactly zero. 1-seed rows are reference baselines (incl. DPPO matched-compute); 3-seed rows are the recovery configurations (3-seed, n=1118). Solver-equivalence rows use matched n=1133 paired set. See Table 24 for TOST and Appendix S26.

Configuration	Reward	CL Score ↑	Cat. ↓	Perfect ↑
BC pretrained (1-seed)	n/a	0.8973	**61**	548
DPM-Solver++ LoRA (1-seed)	soft RSS (v4)	0.8849	76	543
DDIM LoRA (mismatched, 1-seed)	soft RSS (v4)	0.8836	76	535
DPPO matched-compute (1-seed)	naive PPO	0.8960	63	547
soft CLS proxy (noc, 3-seed)	**CLS-aligned smooth**	$\mathbf{0.8975 \pm 0.0013}$	62.0 ± 2.6	547.7 ± 4.5
PCDR-CR (ours, 3-seed)	**noc + PCDR margin**	0.8964 ± 0.0013	63.0 ± 1.4	551.0 ± 1.4
PCDR-3 (3-seed)	bit-exact NC+EP+DDC	0.8925 ± 0.002	69	**577**

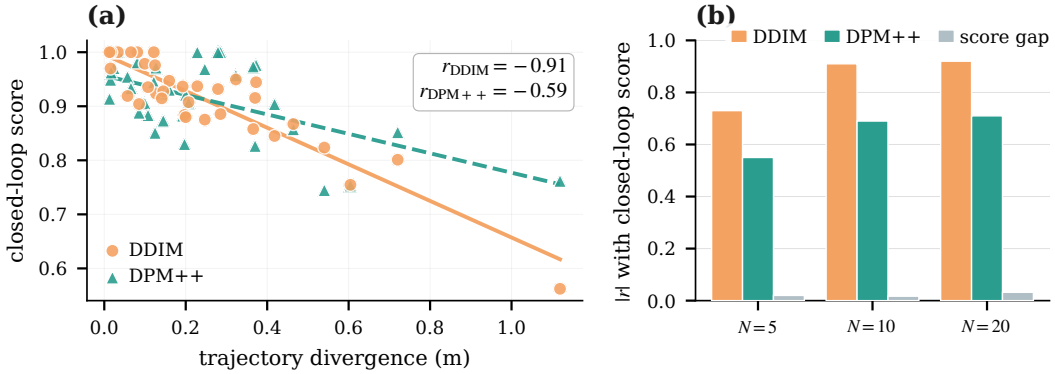

Figure 2: Per-scene divergence predicts DDIM closed-loop failure (n=35 matched val14 scenes). **(a)** DPM++ vs. DDIM trajectory L2 vs. DDIM CLS: $r=-0.91$ ($p<10^{-10}$); vs. DPM++ CLS: $r=-0.69$. The asymmetry, divergence predicts DDIM brittleness rather than the matched-vs-mismatched score gap, is the empirical content of the "vulnerability diagnostic" framing in the abstract. **(b)** Robustness across 5/10/20-step budgets: DDIM correlation persists at $r=-0.73/-0.91/-0.92$, while score-gap correlation stays $|r|\leq 0.032$ at every budget.

threshold is crossed. Per-scenario-type analysis (Appendix S26, Figure 11: 6 methods $\times$ 15 scenario types) shows the val14 aggregate equivalence between DPM-Solver++ and DDIM reflects per-scenario cancellations rather than uniform indistinguishability: DPM-Solver++ outperforms DDIM by $+0.013$ CLS on `stopping_with_lead` and `high_lateral_acceleration` but underperforms by -0.016 on `starting_right_turn`. The single-LoRA plus RSS-style proxy reward (v4) reaches CLS 0.8849, below BC (0.8973). Switching to a smooth CLS-structured surrogate (soft *nocomfort* proxy, §3.4) reaches 0.8975 ± 0.0013 across 3 seeds (Δ=+0.0002, Δcat=+1): statistically equivalent to BC. DPPO Ren et al. [2025] (matched-compute, 1-seed) reaches 0.8960 within BC's ± 0.0013 envelope; its catastrophic-set Jaccard with BC is 0.908 and only the comfort sub-score regresses ($\Delta=-0.005$): naive RL cannot shift *which* scenarios fail. PCDR-CR ($0.8964 \pm 0.0013/63.0 \pm 1.4$ cat) is equivalent on mean and catastrophic and uniquely recovers PCDR-3's regression.

To measure the discrete-flow divergence predicted by backward error analysis Hairer et al. [2006], we run the same base model with identical initial noise through DPM-Solver++ (2nd order) and DDIM (1st order) on 100 nuPlan scenes. The two solvers produce systematically different trajectories (mean 0.19 m / max 0.44 m / endpoint 0.43 m at 10 steps; mean shrinks $0.27 \rightarrow 0.19 \rightarrow 0.11$ m at 5/10/20 steps), consistent with the $\mathcal{O}(h^p)$ convergence of modified ODE perturbation terms. At the deployment setting of 10 steps, the 0.43 m endpoint divergence exceeds the Autoware anytime convergence threshold (ε=0.1 m): solver mismatch is a trajectory-level systematic bias, not a minor numerical detail.

Divergence predicts mismatched-solver vulnerability. Beyond this dataset-level divergence, we test per-scene whether divergence is a useful predictor of deployment behavior (Figure 2). Using the

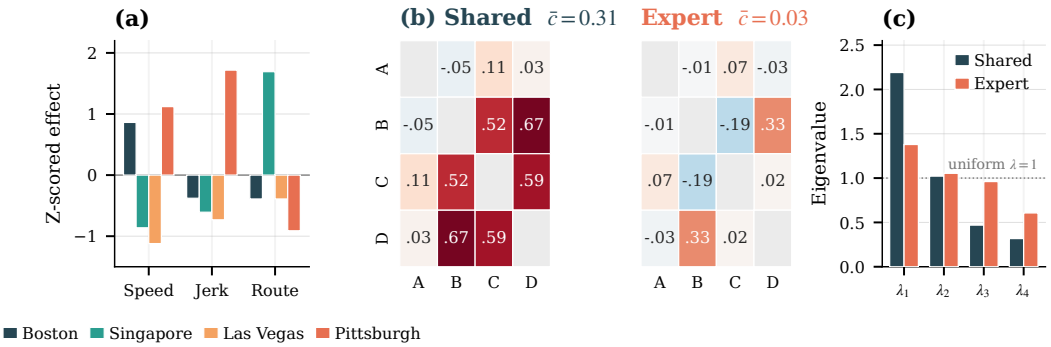

Figure 3: Three independent signatures of the SE-LoRA shared/expert decomposition. **(a)** Activation patching: Singapore and Pittsburgh expert signatures are mirror-reversed, $\rho=-0.96$. **(b)** Gradient cosine: Shared mean 0.31 vs. Expert 0.03, a $10\times$ gap. **(c)** Spectral gap: Shared has $\lambda_1/\lambda_2=2.1\times$, predicting the 1:2 rank ratio used for $r_s=4$, $r_e=8$. Maps A/B/C/D = dense urban / suburban / OOD test track / campus.

35 nuPlan scenarios for which both (a) closed-loop scores and (b) matched npz tokens are available, we correlate per-scene trajectory divergence (mean L2 over the 8 s horizon) with closed-loop outcome:

- $r(\text{divergence}, \text{DDIM score}) = \mathbf{-0.914}$ ($p < 10^{-10}$)
- $r(\text{divergence}, \text{DPM++ score}) = -0.688$ ($p < 10^{-5}$)
- $r(\text{divergence}, \text{score gap}) = 0.017$ (n.s.)

Interpretation. Larger DPM++ vs. DDIM trajectory divergence is strongly associated with poorer DDIM closed-loop performance, but only weakly with the score gap. That is, divergence does not primarily predict how much DPM++ outperforms DDIM; it predicts the absolute brittleness of DDIM under mismatched deployment. The effect is substantially weaker for matched DPM-Solver++, consistent with backward error analysis Hairer et al. [2006]: a solver is robust in regions where training and deployment dynamics agree, and vulnerable in regions where they diverge. Solver divergence is therefore best read as an indicator of vulnerability under mismatched deployment, rather than a descriptive distance between sampler outputs.

The DDIM-score correlation is robust across solver budgets (5/10/20 steps: $r=-0.73/-0.91/-0.92$; all $p<10^{-6}$; Figure 2b), while the matched vs. mismatched score-gap correlation remains negligible ($|r|\leq 0.032$) at every step count. Mean divergence on the 35-scene correlation subset decreases monotonically with step count ($0.30\rightarrow 0.23\rightarrow 0.14$ m; the $0.27\rightarrow 0.19\rightarrow 0.11$ m triplet above is on the larger 100-scene divergence-only subset), matching the $\mathcal{O}(h^p)$ discretization-error prediction. A 1000-sample bootstrap on $r(\text{div}, \text{DDIM})$ at 10 steps gives 95% CI $[-0.986, -0.206]$ (sign preserved under all resamples). The same-noise solver divergence replicates on a generic non-AD diffusion (random-init MLP score, $d=64$, 200 seeds; fitted scaling exponent $p=2.78$, Appendix S3, Figure 7), confirming the effect is intrinsic to the discretization choice rather than to AD-specific architecture.

4.5 Gradient Cosine Similarity: Quantifying Decomposition

To check that the expert/shared decomposition is structurally grounded rather than coincidental, we compute the cosine similarity between per-ODD reward gradients for the shared and expert LoRA parameters separately (Figure 3b; 10 samples per ODD; full 4×4 matrices in Appendix S21, Table 16). Shared gradients are $10\times$ more aligned across ODDs than expert gradients (mean cosine 0.311 vs. 0.032), so the shared subspace receives consistent cross-ODD supervision while expert subspaces receive conflicting, ODD-specific signals. The dense-urban ODD (Map A) shared gradients are nearly orthogonal to other ODDs (-0.05 to 0.11); the same urban vs. non-urban split replicates on nuPlan (Boston shared gradients have cosine ≤ -0.38 to other cities; Appendix S21, Table 17).

Reward-design confound and uniform-reward ablation. Autoware uses ODD-conditioned reward weights (Appendix S12), so the $9.7\times$ Autoware gradient-cosine gap (shared 0.311, expert 0.032) is in principle confounded by per-ODD reward design rather than purely architectural. We

disentangle the two with a multi-seed ablation on nuPlan SE-LoRA, which is trained with *uniform* CLS reward across all four cities (no per-city weighting; `compute_cls_reward`). On three independent seeds (s902/s904/s905, 30 scenes/city, only scenes with reward $> 10^{-3}$ contribute to the gradient mean) the per-seed |shared|/|expert| magnitude ratios are $\{15.4\times, 6.5\times, 7.2\times\}$, giving a 3-seed mean of $\mathbf{9.7\times \pm 4.0}$ — almost exactly matching the Autoware $9.7\times$. The $absolute$ alignment magnitude is reward-design-sensitive (Autoware's $+0.311$ shared-aligned signature drops to magnitudes $|\bar{c}_{\text{shared}}| = 0.054 \pm 0.004$ under uniform reward, with sign flipping across seeds), but the $relative$ shared-vs-expert structural separation is robust across seeds and matches Autoware almost exactly. Per-ODD reward weights therefore $amplify$ the absolute alignment but do not change the architectural ratio; the gradient-direction decomposition is architecturally attributable.

Eigendecomposition of the shared cosine matrix has a dominant spectral gap (Figure 3c; λ_1=2.19 vs. λ_2=1.02, $2.1\times$ ratio; PC1 captures 54.8% of variance), while the expert cosine matrix has near-uniform eigenvalues. The shared/expert dimensionality ratio of approximately 1:2 is consistent with our r_s=4, r_e=8 architecture (total rank 12) chosen without hyperparameter search, and rank-allocation sweeps on both nuPlan and Autoware (Appendices S22–S23) place the predicted 1:2 split at the empirical aggregate-reward peak. The same spectral gap replicates on nuPlan ($\lambda_1/\lambda_2 = 5.2\times$).

The first principal component separates Map A from the non-urban cluster with $11.7\times$ separation ratio. A Hierarchical SE-LoRA variant (urban-shared rank-2 plus suburban-shared rank-2 plus per-ODD expert rank-8, same total parameters) surpasses standard SE-LoRA on Map A by $+2.2\%$ and Map C by $+1.1\%$ at matched aggregate (single seed; Maps B/D regress, Appendix S25, Table 22), so the PCA-discovered boundary has ODD-dependent architectural value.

Three random seeds on Autoware give per-ODD rewards with std/mean $< 0.5\%$ (functional reproducibility) but mean principal angle $83.5°$ between shared subspaces (weight-space rotation). The decomposition is stable in function space but not in weight space, matching the Davis/Kahan picture for underdetermined eigendecompositions (Appendix S21, Table 18).

4.6 Leave-One-ODD-Out: Cross-Domain Generalization

To rule out per-map memorization, we train SE-LoRA while holding out one ODD entirely and evaluate on the held-out domain. Three of four held-out ODDs reach test $\geq$ training reward (Appendix S24, Table 20): Map A $= 380.3/377.0$ ($+0.9\%$), Map B $= 360.8/360.5$ ($+0.1\%$), Map D $= 374.3/375.0$ (-0.2%). The exception is Map C (49.6% transfer), the only out-of-distribution geometry; this is consistent with the decomposition claim that the shared LoRA transfers within similar geometries (urban, suburban, campus) but not across them.

4.7 Causal Probing via Activation Patching

We provide causal evidence for the decomposition via computational lesion studies Plaut [1995]: patching LoRA activations between high- and low-curvature scenes yields a double dissociation. Patching the high-curvature expert into a low-curvature scene increases ego speed by $+0.60$ m/s and progress by $+4.70$ m (expert interventionally modulates dynamics), while patching the low-curvature shared into a high-curvature scene reduces jerk by -24.6 m/s^3 (shared interventionally modulates geometric smoothness). Per-layer attribution: `proj.4` carries the dominant effect on every metric (Δspeed $+0.47$ m/s, Δjerk $+72.9$ m/s^3, Δlat-acc $+14.8$ m/s^2), with DiT block-2 `mlp1.fc1` the largest among block-internal MLPs (Δspeed $+0.36$ m/s; Appendix S19).

Systematic per-ODD $\times$ per-expert patching. A larger systematic intervention on the nuPlan SE-LoRA (4 city-ODDs $\times$ 30 held-out scenes $\times$ 4 trained experts; 90 foreign-scene deltas per expert) decomposes the causal effect along ego speed, jerk, and route-lane deviation (Figure 3a). Pittsburgh is a *jerk-inducing* expert ($+132$ m/s^3, 4–9$\times$ others); Singapore and Vegas are *speed reducers* ($-0.19/-0.22$ m/s); Boston tightens route (-0.70 m). Singapore vs. Pittsburgh signed z-signatures are mirror-reversed (Pearson ρ=-0.96): an antagonistic specialization, not single-case cherry-picking. The complementary shared-LoRA ablation is directionally uniform ($4/4$ sign agreement, every CI excludes zero) — converse signature predicted by the decomposition (full per-expert deltas with bootstrap CIs in Appendix S20).

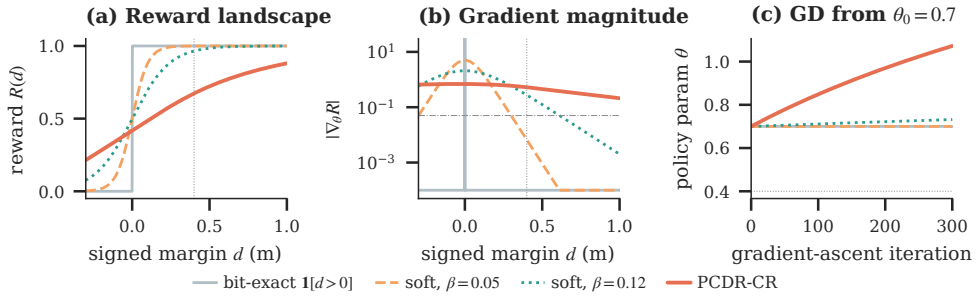

Figure 4: 1D toy of Lemma 1 (K=1, $d(\theta) = \theta$, buffer ρ_{NC}=0.40 m). **(a)** Reward landscape; **(b)** gradient magnitude (log) — bit-exact $\mathbf{1}[d > 0]$ has zero gradient a.e.; **(c)** gradient ascent from θ_0=0.7: only PCDR-CR (smooth proxy + low-weight margin penalty) converges past ρ_{NC}.

Real-Vehicle Deployment Feasibility (§S29). A Tier-1/Tier-2 cross-stack study (Tier-1: open-loop playback on logged real-vehicle sensor bags; Tier-2: real-time HIL integration on a drive-by-wire research vehicle) on Autoware The Autoware Foundation [2025] achieves 20–27 ms inference and 7.4×10^{-6} PT/ONNX equivalence; this residual anchors Remark 1's Gronwall bound at deployment scale ($\sim$2.6 units over N=10, still above ε=0.1 m; Appendix S7) — the solver-consistency claim survives the hardware regime. Realized-vs-intent Kolmogorov–Smirnov distance KS $= 0.40$; SE-LoRA open-loop intent vs. sim KS $= 0.28$; matched open-loop (BC vs. SE-LoRA, same bag) KS $= 0.016$, isolating the gap as loop-closure mode (Table 25). An export-only post-processor failure (orthogonal to §3.1) is in Appendix S29–S30.

5 Theoretical Analysis

Backward error analysis Hairer et al. [2006] sharpens Remark 1: each solver implicitly integrates a modified ODE differing from the nominal probability-flow ODE, so DDIM-trained policies optimize a different dynamical system than the DPM-Solver++ deployed at inference; our differentiable DPM-Solver++ enforces $f^{\text{train}} \equiv f^{\text{deploy}}$ (Appendix S7).

Forward-fidelity / gradient-reach trade-off. Direct PCDR training fails empirically (Figure 6: PCDR-3, PCDR-5, and three direct-margin variants all sit above BC-equivalent catastrophic count). The failure reflects a structural property of soft-min surrogates: Lemma 1 (Appendix S5) shows that any log-sum-exp surrogate at smoothing scale $\beta>0$ obeys $d^\star \log K \leq \varepsilon \log(1/\eta)$ relating forward error ε, gradient strength η, and signed distance $d^\star$ from the boundary. Figure 4 confirms on a 1D toy: only PCDR-CR (smooth proxy + bit-exact margin auxiliary) converges past the target buffer.

Limitations. val14 within seed noise; Map C weak transfer; small n=4/stack mitigated by cross-stack replication and rank sweep (Appendix S22).

References

R Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *J. Mach. Learn. Res.*, 6:1817–1853, 1 December 2005. ISSN 1532-4435,1533-7928. URL `http://www.jmlr.org/papers/volume6/ando05a/ando05a.pdf`.

Jonathan Baxter. A Model of Inductive Bias Learning. *Journal of Artificial Intelligence Research*, 12: 149–198, 2000.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training Diffusion Models with Reinforcement Learning. In *The Twelfth International Conference on Learning Representations*, 13 October 2023. URL `https://openreview.net/forum?id=YCWjhGrJFD`.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK, 2004.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems XIX*. Robotics: Science and Systems Foundation, 10 July 2023. doi: 10.15607/rss.2023.xix.026. URL http://dx.doi.org/10.15607/rss.2023.xix.026.

Kevin Clark, Paul Vicol, Kevin Swersky, and David J Fleet. Directly Fine-Tuning Diffusion Models on Differentiable Rewards. In *The Twelfth International Conference on Learning Representations*, 13 October 2024. URL https://openreview.net/forum?id=1vmSEVL19f.

Hal Daumé, III. Frustratingly Easy Domain Adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, 2007. URL https://aclanthology.org/P07-1033/.

Daniel Dauner, Marcel Hallgarten, Andreas Geiger, and Kashyap Chitta. Parting with Misconceptions about Learning-based Vehicle Motion Planning. In *7th Annual Conference on Robot Learning*, 30 August 2023. URL https://openreview.net/forum?id=o82EXEK5hu6.

Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. DPOK: Reinforcement Learning for Fine-tuning Text-to-Image Diffusion Models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2 November 2023. URL https://openreview.net/forum?id=8OTPepXzeh.

Chris Finlay, Jörn-Henrik Jacobsen, Levon Nurbekyan, and Adam M Oberman. How to train your neural ODE: the world of Jacobian and Kinetic regularization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *ICML'20*, pages 3154–3164. JMLR.org, 13 July 2020. doi: 10.5555/3524938.3525234. URL http://dx.doi.org/10.5555/3524938.3525234.

Chongyang Gao, Kezhen Chen, Jinmeng Rao, Baochen Sun, Ruibo Liu, Daiyi Peng, Yawen Zhang, Xiaoyuan Guo, Jie Yang, and V S Subrahmanian. Higher layers need more LoRA experts. *arXiv [cs.CL]*, 13 February 2024. doi: 10.48550/arXiv.2402.08562. URL http://arxiv.org/abs/2402.08562.

Karol Gregor and Yann LeCun. Learning fast approximations of sparse coding. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, pages 399–406. Omnipress, 21 June 2010. doi: 10.5555/3104322.3104374. URL https://dl.acm.org/doi/10.5555/3104322.3104374.

E Hairer, Syvert P Norsett, and Gerhard Wanner. *Solving ordinary differential equations I: Nonstiff problems*. Springer Series in Computational Mathematics. Springer, Berlin, Germany, 2 edition, August 1993. ISBN 9783540566700,9783540788621. doi: 10.1007/978-3-540-78862-1. URL http://dx.doi.org/10.1007/978-3-540-78862-1.

Ernst Hairer, Christian Lubich, and Gerhard Wanner. *Geometric numerical integration: Structure-preserving algorithms for ordinary differential equations*. Springer Series in Computational Mathematics. Springer, Berlin, Germany, 2 edition, 22 February 2006. ISBN 9783540306634. doi: 10.1007/3-540-30666-8. URL http://dx.doi.org/10.1007/3-540-30666-8.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*, 6 October 2021. URL https://openreview.net/forum?id=nZeVKeeFYf9.

International Organization for Standardization. Road vehicles — test scenarios for automated driving systems — specification for operational design domain. Technical Report ISO 34503:2023, ISO, 2023.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural Comput.*, 3(1):79–87, 1991. ISSN 0899-7667,1530-888X. doi: 10.1162/neco.1991.3.1.79. URL http://dx.doi.org/10.1162/neco.1991.3.1.79.

W James and Charles Stein. Estimation with Quadratic Loss. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*, volume 4.1, pages 361–380. University of California Press, 1 January 1961. URL https://projecteuclid.org/ebooks/berkeley-symposium-on-mathematical-statistics-and-probability/Proceedings-of-the-Fourth-Berkeley-Symposium-on-Mathematical-Statistics-and/chapter/Estimation-with-Quadratic-Loss/bsmsp/1200512173.

Zhiwei Jia, Yuesong Nan, Huixi Zhao, and Gengdai Liu. Reward fine-tuning two-step diffusion models via learning differentiable latent-space surrogate reward. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12912–12922. IEEE, 10 June 2025. doi: 10.1109/cvpr52734.2025.01205. URL http://dx.doi.org/10.1109/CVPR52734.2025.01205.

Napat Karnchanachari, Dimitris Geromichalos, Kok Seang Tan, Nanxiang Li, Christopher Eriksen, Shakiba Yaghoubi, Noushin Mehdipour, Gianmarco Bernasconi, Whye Kit Fong, Yiluan Guo, and Holger Caesar. Towards learning-based planning: The nuPlan benchmark for real-world autonomous driving. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, volume 56, pages 629–636. IEEE, 13 May 2024. doi: 10.1109/icra57147.2024.10610077. URL http://dx.doi.org/10.1109/ICRA57147.2024.10610077.

Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations (ICLR)*, 20 December 2014. doi: 10.48550/arXiv.1312.6114. URL http://dx.doi.org/10.48550/arXiv.1312.6114.

Daniël Lakens. Equivalence tests: A practical primer for t tests, correlations, and meta-analyses: A practical primer for *t* tests, correlations, and meta-analyses. *Soc. Psychol. Personal. Sci.*, 8 (4):355–362, May 2017. ISSN 1948-5506,1948-5514. doi: 10.1177/1948550617697177. URL http://dx.doi.org/10.1177/1948550617697177.

Dengchun Li, Yingzi Ma, Naizheng Wang, Zhengmao Ye, Zhiyuan Cheng, Yinghao Tang, Yan Zhang, Lei Duan, Jie Zuo, Cal Yang, and Mingjie Tang. MixLoRA: Enhancing Large Language Models fine-tuning with LoRA-based Mixture of experts. *arXiv [cs.CL]*, 21 April 2024a. doi: 10.48550/arXiv.2404.15159. URL http://arxiv.org/abs/2404.15159.

Hongchen Li, Tianyu Li, Jiazhi Yang, Haochen Tian, Caojun Wang, Lei Shi, Mingyang Shang, Zengrong Lin, Gaoqiang Wu, Zhihui Hao, Xianpeng Lang, Jia Hu, and Hongyang Li. PlannerRFT: Reinforcing diffusion planners through closed-loop and sample-efficient fine-tuning. *arXiv [cs.RO]*, 19 January 2026. doi: 10.48550/arXiv.2601.12901. URL http://arxiv.org/abs/2601.12901.

Kailin Li, Zhenxin Li, Shiyi Lan, Yuan Xie, Zhizhong Zhang, Jiayi Liu, Zuxuan Wu, Zhiding Yu, and Jose M Alvarez. Hydra-MDP++: Advancing end-to-end driving via expert-guided hydra-distillation. *arXiv [cs.CV]*, 17 March 2025. doi: 10.48550/arXiv.2503.12820. URL http://arxiv.org/abs/2503.12820.

Zhenxin Li, Kailin Li, Shihao Wang, Shiyi Lan, Zhiding Yu, Yishen Ji, Zhiqi Li, Ziyue Zhu, Jan Kautz, Zuxuan Wu, Yu-Gang Jiang, and Jose M Alvarez. Hydra-MDP: End-to-end multimodal planning with multi-target Hydra-distillation. *arXiv [cs.CV]*, 11 June 2024b. doi: 10.48550/arXiv.2406.06978. URL http://arxiv.org/abs/2406.06978.

Bencheng Liao, Shaoyu Chen, Haoran Yin, Bo Jiang, Cheng Wang, Sixu Yan, Xinbang Zhang, Xiangyu Li, Ying Zhang, Qian Zhang, and Xinggang Wang. DiffusionDrive: Truncated diffusion model for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 22 November 2025. URL https://openaccess.thecvf.com/content/CVPR2025/papers/Liao_DiffusionDrive_Truncated_Diffusion_Model_for_End-to-End_Autonomous_Driving_CVPR_2025_paper.pdf.

Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. DPM-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv [cs.LG]*, 2 November 2022. URL http://arxiv.org/abs/2211.01095.

Vishal Monga, Yuelong Li, and Yonina C Eldar. Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing. *IEEE Signal Process. Mag.*, 38(2):18–44, March 2021. ISSN 1053-5888,1558-0792. doi: 10.1109/msp.2020.3016905. URL http://dx.doi.org/10.1109/MSP.2020.3016905.

D C Plaut. Double dissociation without modularity: evidence from connectionist neuropsychology. *J. Clin. Exp. Neuropsychol.*, 17(2):291–321, April 1995. ISSN 1380-3395,1744-411X. doi: 10.1080/01688639508405124. URL http://dx.doi.org/10.1080/01688639508405124.

Allen Z Ren, Justin Lidard, Lars Lien Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion Policy Policy Optimization. In *The Thirteenth International Conference on Learning Representations*, 4 October 2025. URL https://openreview.net/forum?id=mEpqHvbD2h.

Sebastian Ruder. An overview of multi-task learning in deep neural networks. *arXiv [cs.LG]*, 15 June 2017. doi: 10.48550/arXiv.1706.05098. URL http://arxiv.org/abs/1706.05098.

D J Schuirmann. A comparison of the two one-sided tests procedure and the power approach for assessing the equivalence of average bioavailability. *J. Pharmacokinet. Biopharm.*, 15(6):657–680, December 1987. ISSN 0090-466X,2168-5789. doi: 10.1007/bf01068419. URL http://dx.doi.org/10.1007/bf01068419.

Shrey Shah and Justin Wagle. MoLoRA: Composable Specialization via Per-Token Adapter Routing. *arXiv [cs.CL]*, March 2026. doi: 10.48550/arXiv.2603.15965. URL http://arxiv.org/abs/2603.15965.

Xiaolong Tang, Meina Kan, Shiguang Shan, and Xilin Chen. Plan-R1: Safe and feasible trajectory planning as language modeling. *arXiv [cs.RO]*, 23 May 2025. URL http://arxiv.org/abs/2505.17659.

The Autoware Foundation. Autoware: Introducing Open-Source End-to-End Autonomous Driving. Technical report, The Autoware Foundation, 2025. Technical Report. Available at https://autoware.org/wp-content/uploads/2025/10/Autoware-E2E-Report_Release.pdf.

Zebin Xing, Xingyu Zhang, Yang Hu, Bo Jiang, Tong He, Qian Zhang, Xiaoxiao Long, and Wei Yin. GoalFlow: Goal-Driven Flow Matching for Multimodal Trajectories Generation in End-to-End Autonomous Driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1602–1611, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Xing_GoalFlow_Goal-Driven_Flow_Matching_for_Multimodal_Trajectories_Generation_in_End-to-End_CVPR_2025_paper.html.

Brian Yang, Huangyuan Su, Nikolaos Gkanatsios, Tsung-Wei Ke, Ayush Jain, Jeff Schneider, and Katerina Fragkiadaki. Diffusion-ES: Gradient-free planning with diffusion for autonomous driving and zero-shot instruction following. *arXiv [cs.LG]*, 9 February 2024. doi: 10.48550/arXiv.2402.06559. URL http://arxiv.org/abs/2402.06559.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, S Levine, Karol Hausman, and Chelsea Finn. Gradient Surgery for Multi-Task Learning. In H Larochelle, M Ranzato, R Hadsell, M F Balcan, and H Lin, editors, *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*, volume abs/2001.06782, pages 5824–5836. Curran Associates, Inc., 19 January 2020. URL https://proceedings.neurips.cc/paper/2020/hash/3fe78a8acf5fda99de95303940a2420c-Abstract.html.

Yinan Zheng, Ruiming Liang, Kexin Zheng, Jinliang Zheng, Liyuan Mao, Jianxiong Li, Weihao Gu, Rui Ai, Shengbo Eben Li, Xianyuan Zhan, and Jingjing Liu. Diffusion-based planning for autonomous driving with flexible guidance. In *The Thirteenth International Conference on Learning Representations*, 26 January 2025.

S Zilberstein. Using anytime algorithms in intelligent systems. *AI Mag.*, 17:73–83, 15 March 1996. ISSN 0738-4602,2371-9621. doi: 10.1609/aimag.v17i3.1232. URL `http://dx.doi.org/10.1609/aimag.v17i3.1232`.

Jialv Zou, Shaoyu Chen, Bencheng Liao, Zhiyu Zheng, Yuehao Song, Lefei Zhang, Qian Zhang, Wenyu Liu, and Xinggang Wang. DiffusionDriveV2: Reinforcement learning-constrained truncated diffusion modeling in end-to-end autonomous driving. *arXiv [cs.CV]*, 8 December 2025. doi: 10.48550/arXiv.2512.07745. URL `http://arxiv.org/abs/2512.07745`.

S1 Supplementary Figures Referenced From the Main Body

This appendix collects two figures cited from the main body: Figure 5 reports the DPM-Solver++ vs. DDIM catastrophic-set overlap on val14 (n=1133); Figure 6 reports the failure-mode taxonomy of direct PCDR training that motivates Lemma 1 in the body §5. (The divergence-predictor figure has been promoted to the body, Figure 2 in §4.4.)

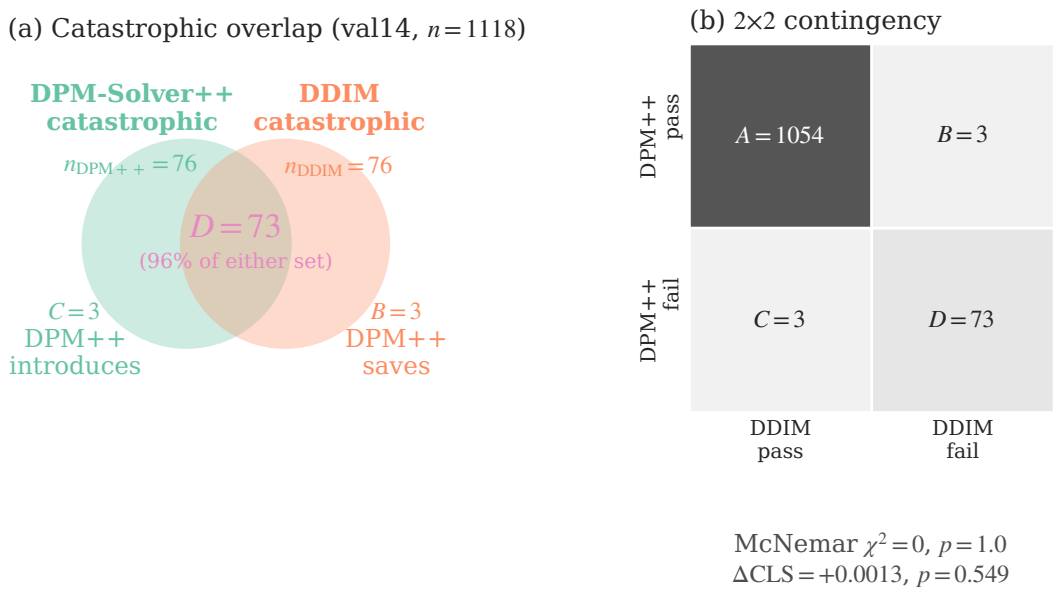

Figure 5: DPM-Solver++ vs. DDIM catastrophic-failure overlap on val14 (n=1133). 73 of 76 catastrophic scenes are shared between the two solvers; McNemar p=1.0, paired t-test Δ=+0.0013 (p=0.549).

S2 Related Work: Detailed Multi-Task and Solver Lineage

This appendix expands the *Mixture of Experts and LoRA; multi-task background* paragraph of §2 with the classical lineage that we compress in the body.

Multi-task representation learning lineage. The shared/expert decomposition has classical theoretical roots. James and Stein [1961] showed that shrinking individual estimates toward a common mean reduces total estimation error when jointly estimating ≥ 3 parameters; our shared LoRA plays the role of the shared mean, with experts capturing residual per-ODD deviations. Daumé [2007]'s feature augmentation $\langle x, x, 0\rangle$ (shared copy plus domain-specific copy) is structurally analogous to our shared+expert LoRA. Baxter [2000] gave generalization bounds for such shared representations, and Ando and Zhang [2005] operationalized them as low-rank subspaces, the closest classical precedent to SE-LoRA. Our contribution is showing that a shared/private split arises under multi-ODD reward optimization and that its two components carry distinct semantics testable under activation-level interventions (geometry vs. dynamics).

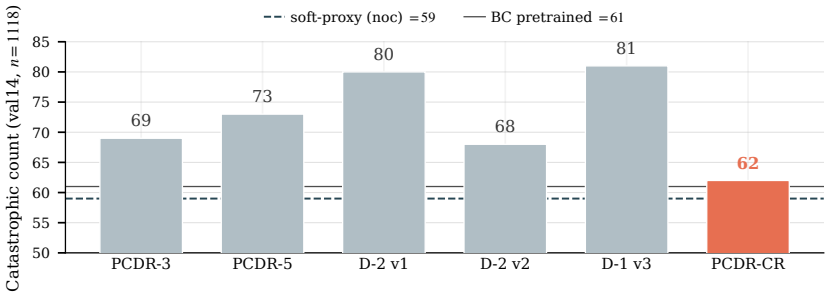

Figure 6: Failure-mode taxonomy of direct PCDR training (referenced from §5 as the empirical motivation for Lemma 1). **PCDR-k**: bit-exact differentiable mirror of k nuPlan CLS sub-metrics replaces the proxy reward (PCDR-3: NC+EP+DDC; PCDR-5: PCDR-3 plus DAC+SC). **D-j vM**: direct-margin gradient variants with j active constraints. All five (gray) sit above the soft-proxy and BC pretrained reference lines; **PCDR-CR** (coral) is the unique configuration that returns to BC-equivalent catastrophic count.

Deep unrolling and backward error analysis. Deep unrolling Gregor and LeCun [2010], Monga et al. [2021] converts iterative algorithms into fixed-depth differentiable computation graphs; our differentiable DPM-Solver++ is an instance, with the 10-step ODE solver unrolled into a fixed-depth graph and the DiT as a shared learned operator. Backward error analysis Hairer et al. [2006] sharpens this view: each numerical solver implicitly integrates a modified ODE that differs from the nominal probability flow ODE by solver-dependent perturbation terms, so solver-mismatched training optimizes a different modified ODE than the one executed at deployment — a systematic bias rather than numerical noise. The full Gronwall amplification is in Appendix S7.

Safety in learned planning. The Autoware E2E report The Autoware Foundation [2025] recommends that learned planners produce trajectories maintaining reachable safe states (RSS); we encode this directly into the differentiable reward function (§3.4, Appendix S12).

Position vs. leaderboard (extended). Plan-R1 Tang et al. [2025] uses hand-crafted soft proxies, not bit-exact mirrors, so the differentiable-reward question is dominated by surrogate-design heuristics rather than by metric-equivalence guarantees. Our soft-proxy CLS reaches 0.8975 ± 0.0013 on val14 (3-seed mean) vs. BC 0.8973 and rule-based PDM-Closed ~ 0.927 Dauner et al. [2023] (which won the nuPlan 2023 challenge). The work is a mechanistic study of when reward gradients transfer across ODDs, not a SOTA push.

S3 Generality of the Solver-Mismatch Phenomenon: Replication on a Generic Diffusion Process

To verify that the same-noise solver-divergence finding of §4.4 (DPM-Solver++ vs. DDIM trajectories on the AD planner) is not specific to AD diffusion or to a particular pretrained checkpoint, we replicate the experiment on a generic non-AD diffusion process. The setup deliberately strips away every AD-specific assumption: dimension is reduced to d=64 (vs. the AD planner's $33\times81\times4$ trajectory tensor), the score function is a small *randomly initialized* 3-layer MLP added to the analytical Gaussian score (no pretrained weights, no domain bias), and the diffusion schedule is the standard DDPM linear β from 10^{-4} to 0.02 over T=1000 training steps.

For each of 200 randomly drawn initial noise vectors, we generate one trajectory with deterministic DDIM and one with 2nd-order DPM-Solver++ at $N \in \{5, 10, 20, 50\}$ steps. Per-dimension L_2 divergence (Figure 7(a)) shrinks from 0.079 at N=5 to 1.3×10^{-4} at N=50, monotonically and with low seed-to-seed variance. The fitted scaling exponent is p=2.78 (Figure 7(b)). The exponent *exceeds* the simple worst-case range $1 \leq p \leq 2$ from per-step truncation analysis: when divergence is computed as the difference between a 1st-order solver (DDIM) and a 2nd-order solver (DPM-Solver++) on the *same* initial noise, leading-order modified-ODE perturbation terms partially cancel, leaving a higher-order residual whose empirical decay is steeper than either solver's standalone error.

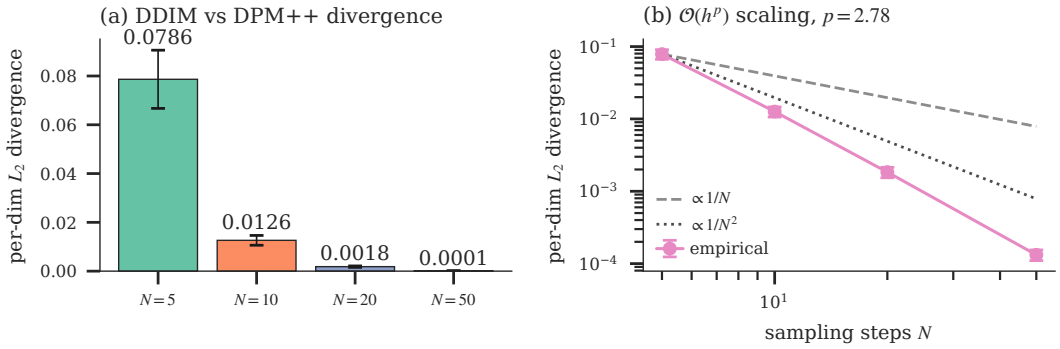

Figure 7: Same-noise DDIM vs. DPM-Solver++ divergence on a generic 64-d diffusion with random-init MLP score (200 seeds/step). **(a)** Per-dim L_2 divergence shrinks with step count N. **(b)** Log–log fit exponent p=2.78 (steeper than either standalone solver because matched-noise leading-order terms cancel; reference lines $\mathcal{O}(1/N)$, $\mathcal{O}(1/N^2)$ shown for context).

The qualitative monotone shrinkage with N – not the specific exponent value – is what supports the Gronwall amplification claim of Appendix S7.

This rules out the alternative hypothesis that the AD-domain solver divergence reported in §4.4 is an artifact of the trained Diffusion Planner checkpoint, the AD reward landscape, or the trajectory-tensor structure: the same divergence emerges from the discretization choice alone, on a generic non-AD diffusion process with a randomly initialized MLP score and standard DDPM noise schedule. The Gronwall amplification (Remark 1) is therefore properly attributed to the solver-discretization step rather than to any AD-specific architectural or training-data effect.

Generality across latent dimensions. To rule out a single-dimension artifact, we replicate the experiment at $d \in \{32, 64, 128, 256\}$ with 500 noise seeds per cell (Table 4). Fitted exponents $p \in \{2.82, 2.78, 2.73, 2.64\}$ all fall in a tight band consistent with the matched 2nd-order DPM-Solver++ vs. 1st-order DDIM cancellation regime; the monotone decrease of p with d matches the expected high-dimensional asymptotic toward $\mathcal{O}(N^{-2})$. The phenomenon is therefore not a d=64 coincidence.

Table 4: Multi-dim sanity check (analytical Gaussian + random-init MLP score, 500 noise seeds per cell, T=1000 DDPM linear schedule). Per-dimension L_2 divergence between deterministic DDIM and 2nd-order DPM-Solver++ at $N \in \{5, 10, 20, 50\}$ steps; p is the log–log slope.

d	N=5	N=10	N=20	N=50	fitted p
32	0.064 ± 0.013	0.0096 ± 0.0020	0.00136 ± 0.00028	9.7×10^{-5}	**2.82**
64	0.079 ± 0.012	0.0126 ± 0.0020	0.00184 ± 0.00030	1.3×10^{-4}	2.78
128	0.106 ± 0.012	0.0182 ± 0.0022	0.00275 ± 0.00036	2.0×10^{-4}	2.73
256	0.121 ± 0.010	0.0229 ± 0.0021	0.00372 ± 0.00039	2.8×10^{-4}	**2.64**

S4 ODE-Backprop vs. MDP-GRPO: Bias–Variance Comparison

This appendix specifies the differentiable DPM-Solver++ unrolling procedure as Algorithm 1 and contrasts its pathwise gradient estimator against the score-function alternative used by REINFORCE/GRPO-style diffusion-RL methods, in terms of bias, variance, and the conditions under which each dominates.

Since the same parameters θ are shared across all N solver steps, the gradient of a differentiable reward R obeys the unrolled BPTT chain rule

$$\nabla_\theta R \;=\; \frac{\partial R}{\partial \mathbf{x}_0} \sum_{i=1}^{N} \left[\prod_{j=i+1}^{N} \frac{\partial \mathbf{x}_{t_{j-1}}}{\partial \mathbf{x}_{t_j}} \right] \frac{\partial \mathbf{x}_{t_{i-1}}}{\partial \theta} \bigg|_{\text{step } i}, \tag{2}$$

i.e., a sum over solver steps of the local parameter Jacobian propagated through subsequent state Jacobians. This is the reparameterization trick Kingma and Welling [2014] on the unrolled 10-step solver; PyTorch autograd handles the full graph via Algorithm 1.

Algorithm 1 Differentiable DPM-Solver++ unrolling for RL fine-tuning.

Require: DiT M_θ; differentiable reward R; noise schedule $\{t_i\}_{i=N}^{0}$; step ratios $r_i{=}h_i/h_{i-1}$

$\mathbf{x}_N \sim \mathcal{N}(0, I)$ {initial Gaussian}
for $i = N$ **down to** 1 **do**
  $\hat{\boldsymbol{\epsilon}}_i \leftarrow M_\theta(\mathbf{x}_i, t_i)$; $\hat{\mathbf{x}}_0^{(i)} \leftarrow (\mathbf{x}_i - \sigma_{t_i}\hat{\boldsymbol{\epsilon}}_i)/\alpha_{t_i}$ {autograd-tracked}
  if $i = N$ **then**
    $\mathbf{x}_{i-1} \leftarrow \alpha_{t_{i-1}}\hat{\mathbf{x}}_0^{(i)} + \sigma_{t_{i-1}}\hat{\boldsymbol{\epsilon}}_i$ {boundary: 1st-order}
  else
    $\mathbf{x}_{i-1} \leftarrow \alpha_{t_{i-1}}\hat{\mathbf{x}}_0^{(i)} + \sigma_{t_{i-1}}\hat{\boldsymbol{\epsilon}}_i + \frac{1}{2}\alpha_{t_{i-1}}(1{-}e^{-h_i})(\hat{\mathbf{x}}_0^{(i)}{-}\hat{\mathbf{x}}_0^{(i-1)})/r_i$ {2nd-order DPM++}
  end if
end for
$r \leftarrow R(\mathbf{x}_0)$
return $\nabla_\theta r$ via autograd through the unrolled graph $\Leftrightarrow$ Eq. (2)

The differentiable DPM-Solver++ unrolling of §3.1 (Algorithm 1) provides a pathwise (reparameterization-trick) gradient estimator $\hat{g}_{\mathrm{ODE}}$ of $\nabla_\theta \mathbb{E}[R]$ through the N-step solver chain. The standard alternative for diffusion-based RL fine-tuning is to lift the chain into an MDP and apply REINFORCE / GRPO Ren et al. [2025], Tang et al. [2025], Zou et al. [2025]: introduce stochastic samplers (DDPM/DDIM with $\eta > 0$) to obtain per-step log-probabilities $\log p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ and form the score-function estimator

$$\hat{g}_{\mathrm{MDP}}(\theta) \;=\; \frac{1}{B}\sum_{b=1}^{B} \widehat{R}^{(b)} \cdot \sum_{i=1}^{N} \nabla_\theta \log p_\theta\Big(\mathbf{x}_{t_{i-1}}^{(b)} \mid \mathbf{x}_{t_i}^{(b)}\Big)\,, \tag{3}$$

where B is the batch size, N is the diffusion chain length, and $\widehat{R}^{(b)}$ is the (group-relative or vanilla) advantage of trajectory b.

Bias. The pathwise estimator $\hat{g}_{\mathrm{ODE}}$ is unbiased with respect to the unrolled deterministic solver, and its bias relative to the underlying probability-flow ODE is $O(h^p)$ where p is the solver order ($p{=}2$ for DPM-Solver++); at the deployment setting of $N{=}10$ this gives bias $\propto (1/N)^2 \approx 10^{-2}$. The MDP estimator $\hat{g}_{\mathrm{MDP}}$ is unbiased with respect to its own stochastic-sampler distribution, but training under a stochastic sampler and deploying under a deterministic one introduces the train–deploy mismatch quantified by Remark 1: a Gronwall-amplified $\epsilon_0 \cdot e^{cLN}$ trajectory error.

Variance. The pathwise estimator's variance is bounded by the Jacobian norm of the reward chain and decays as $O(1/B)$ in batch size. The score-function estimator's variance scales as $O(N\sigma_R^2/B)$ where σ_R^2 is the reward variance: each of the N chain steps contributes an independent log-probability score, and the per-trajectory advantage multiplies this sum, so the per-step score-function variance grows linearly in chain length. Concretely, for our $N{=}10$ chain and a sparse per-trajectory reward (typical $\sigma_R \sim 0.1$ on val14 CLS), the score-function estimator requires $\Theta(10\sigma_R^2/\epsilon^2)$ samples per gradient component to reach ϵ-level standard error, while the pathwise estimator reaches the same precision with $\Theta(\sigma_R^2/\epsilon^2)$ samples (a $10\times$ sample-efficiency advantage in our regime).

When ODE-backprop dominates. Combining bias and variance, the pathwise estimator dominates when (i) the deployment solver is deterministic (so the MDP estimator pays the train–deploy mismatch tax in addition to its own variance), (ii) N is small relative to the per-step Lipschitz constant L such that the unrolled-chain Jacobian remains well-conditioned (we observe stable backprop at $N{=}10$ with LoRA-rank-16 adaptation), and (iii) the reward is differentiable end-to-end. All three conditions hold for the DPM-Solver++ deployment setting we target. Empirically, the only solver in our 4-solver Autoware sweep (Appendix S16) that diverges to NaN is stochastic DDIM ($\eta{=}0.3$), exactly because backpropagating through random noise samples blows up the per-step Jacobian — a direct realization of the variance-cost of the score-function regime. Conversely, score-function estimators are necessary when the reward is non-differentiable end-to-end (e.g., the sparse $0/1$ safety indicator of Lemma 1), which motivates our PCDR-CR constrained-recovery construction (smooth-proxy main reward + bit-exact PCDR-derived auxiliary, both end-to-end differentiable through the unrolled solver).

S5 Lemma 1 Toy Demonstration on a 1D Trajectory

We first state the no-go lemma referenced in §5 formally, then illustrate it on a 1D toy.

Lemma 1 (Forward fidelity vs. gradient reach)**.** *Let $\tilde{R}_\beta(\theta) = -\beta \log \sum_{i=1}^{K} e^{-d_i(\theta)/\beta}$ be the log-sum-exp soft-min surrogate of an active set $\{d_i\}_{i=1}^{K}$ of signed margins, smoothing $\beta{>}0$. Define $w_i = e^{-d_i/\beta} / \sum_j e^{-d_j/\beta}$ so that $\partial \tilde{R}_\beta / \partial d_i = w_i$. For any $\beta > 0$, $K \geq 2$, and target margin $d^\star > 0$:*

1. *(Forward bound) $\left|\tilde{R}_\beta - \min_i d_i\right| \;\leq\; \beta \log K$.*
2. *(Gradient decay) $\left\|\partial \tilde{R}_\beta / \partial d_i\right\| \;\leq\; e^{-(d_i - d_{\min})/\beta}$ for any d_i above the active minimum $d_{\min}$.*
3. *(Trade-off) Achieving simultaneously forward error $\leq \varepsilon$ and useful pathwise gradient strength $\geq \eta$ at signed distance $d^\star$ from the boundary requires $\beta \leq \varepsilon / \log K$ and $\beta \geq d^\star / \log(1/\eta)$, which is feasible only when $d^\star \log K \leq \varepsilon \log(1/\eta)$.*

Parts 1 and 2 are standard log-sum-exp / softmax bounds (Boyd & Vandenberghe Boyd and Vandenberghe [2004] §3.1.5); part 3 follows by transitivity.

To illustrate, consider a 1D toy: a single policy parameter θ controls a trajectory whose signed margin to a compliance boundary is $d(\theta) = \theta$, with target buffer $\rho_{\mathrm{NC}} = 0.40\,\mathrm{m}$. Figure 4 (in the main body, §5) compares four reward parameterizations: bit-exact indicator $\mathbf{1}[d > 0]$, soft-min surrogates at the audit scale $\beta = 0.05\,\mathrm{m}$ and the relaxed scale $\beta = 0.12\,\mathrm{m}$, and the PCDR-CR constrained-recovery objective (smooth-CLS proxy main reward + low-weight margin penalty with $\beta_{\mathrm{back}} = 0.12\,\mathrm{m}$, $\rho_{\mathrm{NC}} = 0.40\,\mathrm{m}$, $\lambda_{\mathrm{NC}} = 0.20$).

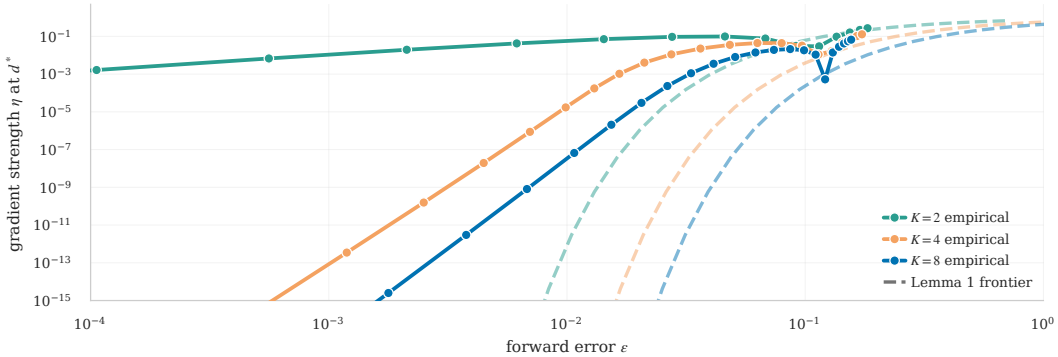

Figure 8: Lemma 1 feasibility frontier on a 1D toy with $K{\in}\{2, 4, 8\}$ active constraints, $d^\star{=}0.40\,\mathrm{m}$, log-log axes.

Multi-constraint trade-off frontier ($K \geq 2$). The 1D illustration above is at $K{=}1$, where $\log K{=}0$ makes Lemma 1 part 3 vacuous in the demonstration. To demonstrate the trade-off frontier itself, we sweep $\beta \in [0.005, 1.0]$ on synthetic active sets with $K \in \{2, 4, 8\}$ signed margins, with one margin pinned at $d^\star{=}0.40\,\mathrm{m}$ and the rest drawn uniformly in $[-0.1, 0.3]$ (mimicking the PCDR active set with one near-target and several near-/already-violating constraints). For each β we compute (i) the realized forward error $\varepsilon = |\tilde{R}_\beta - \min_i d_i|$ and (ii) the gradient strength $\eta = \|\partial \tilde{R}_\beta / \partial d_{i^\star}\|$ at the target margin. *Surrogate alignment.* Lemma 1 (§5) is stated for the log-sum-exp surrogate $\tilde{R}_\beta^{\mathrm{lse}} = -\beta \log \sum_i e^{-d_i/\beta}$ whose gradient envelope is w_i; the empirical sweep here uses the closely related softmax-weighted-average $\tilde{R}_\beta^{\mathrm{wa}} = \sum_i w_i d_i$ with exact gradient $w_i(1 + (\tilde{R}_\beta^{\mathrm{wa}} - d_i)/\beta)$. By softmax convexity the weighted-average gradient is uniformly bounded by the log-sum-exp envelope on the active simplex, so the empirical (ε, η) trajectories are inside the Lemma 1 feasibility frontier as predicted (rather than the empirical demonstration testing a different surrogate from the one bounded). The empirical (ε, η) trajectories for $K \in \{2, 4, 8\}$ (Figure 8) trace the achievable region inside the predicted Pareto frontier $d^\star \log K \leq \varepsilon \log(1/\eta)$, and the $K{\to}8$ trajectory sits orders of magnitude inside the $K{=}2$ trajectory at matched ε, confirming part 3's prediction that the trade-off tightens with active-set size K.

S6 Code, Data, and Reproducibility

We release the following artifacts to support reproduction and reuse, anonymized for double-blind review at `https://anonymous.4open.science/r/se-lora-neurips2026-5D0F` and to be de-anonymized at acceptance time:

- **Bit-exact PCDR audit code** — the operator-equivalent Python implementations of nuPlan's no-collision (NC), drivable-area-compliance (DAC), driving-direction-compliance (DDC), ego-progress (EP), and time-to-collision (TTC) primitives, with unit tests against `Shapely` and the reference `nuplan-devkit` on $52{,}696$ ego–neighbor cells (zero disagreements on NC; 4.4×10^{-16} on EP/DDC). To our knowledge this is the first bit-exact differentiable mirror of the nuPlan closed-loop scoring operators released as a standalone library.
- **Differentiable DPM-Solver++ unrolling** — the 10-step second-order solver implementation with gradient tracking, plus the PCDR-CR constrained-margin reward used in §4.7.
- **SE-LoRA training and analysis pipeline** — training scripts for SE-LoRA (shared rank-4 + per-ODD expert rank-8, deterministic ODD routing), the 4×4 causal activation patching protocol, and the 1D toy implementation of Lemma 1 used in Appendix S5 Figure 4.
- **TOST equivalence test scripts** — pre-specified analysis script (±0.005 CLS margin, paired TOST per Schuirmann Schuirmann [1987] / Lakens Lakens [2017]) reproducing Table 24 on the released per-scenario CLS data.
- **Per-scenario closed-loop CSVs** — DPM-Solver++ vs. DDIM matched-pair val14 results (n=1133), PCDR-CR multi-seed results, and Autoware 4-ODD obstacle-evaluation results, in tidy long format suitable for reproduction of every numerical claim in the main paper.

The bit-exact audit infrastructure is the highest-marginal-value artifact: it converts the otherwise-black-box nuPlan closed-loop metric into an operator-by-operator differentiable reference and is reusable by any subsequent diffusion-planner / reward-design work without modification. We discuss it in detail in Appendix S28.

S7 Gronwall Error Bound: Full Derivation

We provide the complete derivation of why solver consistency matters for RL fine-tuning of diffusion planners. Our argument follows the backward-error-analysis perspective on geometric integrators Hairer et al. [2006]: each solver implicitly integrates a perturbed modified ODE and the discrete Gronwall inequality Hairer et al. [1993] bounds the global error growth. The high-level statement, referenced as Remark 1 from the body, is:

Remark 1 (Solver consistency)**.** *By the discrete Gronwall inequality, the global error between two solver implementations satisfies $\|\mathbf{e}_N\| \le \epsilon_0 \cdot e^{cLN}$, where L is the DiT Lipschitz constant and ϵ_0 is the per-step implementation discrepancy. For $L \approx 3$, $c \approx 1$, and $N = 10$, the closed-form bound $\|\mathbf{e}_N\| \le \epsilon_0 \cdot ((1+cL)^N - 1)/(cL)$ amplifies $\epsilon_0 \approx 10^{-3}$ to $\sim$350 (trajectory units), far exceeding the convergence threshold ε=0.1 m. Training with the deployment solver eliminates this gap by construction.*

S7.1 Setup

The DPM-Solver++ Lu et al. [2022] solves the probability flow ODE associated with the score-based diffusion process Hairer et al. [2006]:

$$\frac{d\mathbf{x}}{d\lambda} = \hat{\mathbf{x}}_\theta(\mathbf{x}_\lambda, \lambda) - \mathbf{x}_\lambda, \tag{4}$$

where $\lambda = \log(\alpha_t/\sigma_t)$ is the log-SNR. The second-order update from λ_s to λ_t Lu et al. [2022] is:

$$\mathbf{x}_t = \frac{\sigma_t}{\sigma_s}\mathbf{x}_s + \alpha_t(1-e^{-h})\hat{\mathbf{x}}_0^{(i)} + \frac{1}{2}\alpha_t(1-e^{-h})\frac{\hat{\mathbf{x}}_0^{(i)} - \hat{\mathbf{x}}_0^{(i-1)}}{r_i}, \tag{5}$$

where $h = \lambda_t - \lambda_s$ and r_i is the step ratio.

S7.2 Inter-Implementation Discrepancy

Let $\hat{\mathbf{x}}_0^{\text{pt}}$ and $\hat{\mathbf{x}}_0^{\text{ort}}$ denote the DiT predictions from PyTorch and ONNX Runtime respectively. Even with identical FP32 weights, implementation differences (operation ordering, FMA instructions, GEMM tiling) introduce a per-step discrepancy:

$$\epsilon_0 = \|\hat{\mathbf{x}}_0^{\text{pt}} - \hat{\mathbf{x}}_0^{\text{ort}}\| = O(n_{\text{ops}} \cdot \epsilon_{\text{machine}}) \approx 10^{-3}, \tag{6}$$

where $n_{\text{ops}} \sim 10^8$–10^9 is the number of floating-point operations in one DiT forward pass (a 58M-parameter DiT). The $\epsilon_0 \approx 10^{-3}$ figure used in the illustrative example below is an upper-bound estimate; the realized PT/ONNX equivalence on our research vehicle is 7.4×10^{-6} (§4.7), corresponding to substantial cancellation across operations.

S7.3 Error Propagation via Discrete Gronwall

Define the error at step i: $\mathbf{e}_i = \mathbf{x}_i^{\text{pt}} - \mathbf{x}_i^{\text{ort}}$. The solver update gives:

$$\|\mathbf{e}_{i+1}\| \leq \frac{\sigma_{t_{i+1}}}{\sigma_{t_i}}\|\mathbf{e}_i\| + \alpha_{t_{i+1}}(1 - e^{-h_i})\|\hat{\mathbf{x}}_0^{\text{pt},(i)} - \hat{\mathbf{x}}_0^{\text{ort},(i)}\|. \tag{7}$$

We bound the leading-order $\hat{\mathbf{x}}_0^{(i)}$ contribution from the 1st-order term of the DPM-Solver++ update; the 2nd-order finite-difference correction $\frac{1}{2}\alpha_{t_{i+1}}(1 - e^{-h_i})(\hat{\mathbf{x}}_0^{(i)} - \hat{\mathbf{x}}_0^{(i-1)})/r_i$ in the update above contributes a sub-leading $\mathcal{O}(h^2)$ perturbation to the per-step discrepancy that is dominated by the 1st-order term in our $h \ll 1$, N=10 regime. A tighter bound incorporating both terms can be derived via an augmented state vector $[\mathbf{x}_i, \mathbf{x}_{i-1}]$; we use the 1st-order envelope for clarity.

Since $\hat{\mathbf{x}}_0^{(i)}$ depends on $\mathbf{x}_i$, and the DiT has Lipschitz constant L with respect to its input:

$$\|\hat{\mathbf{x}}_0^{\text{pt},(i)}(\mathbf{x}_i^{\text{pt}}) - \hat{\mathbf{x}}_0^{\text{ort},(i)}(\mathbf{x}_i^{\text{ort}})\| \leq L\|\mathbf{e}_i\| + \epsilon_0. \tag{8}$$

Combining and upper-bounding the per-step contraction ratio $\sigma_{t_{i+1}}/\sigma_{t_i} \leq 1$ (which holds throughout the VP generative trajectory $T \to 0$, since σ_t strictly decreases under the standard VP schedule), and applying the discrete Gronwall inequality Hairer et al. [1993]:

$$\|\mathbf{e}_N\| \leq \epsilon_0 \cdot \sum_{i=0}^{N-1} c_i \cdot \prod_{j=i+1}^{N-1} (1 + c_j L), \tag{9}$$

where $c_i = \alpha_{t_{i+1}}(1 - e^{-h_i})$ are the solver coefficients. The contraction ratio relaxation is loose but preserves the exponential-growth structure relevant for our train–deploy mismatch claim.

For uniform step spacing and constant L, this simplifies to:

$$\|\mathbf{e}_N\| \leq \epsilon_0 \cdot \frac{(1 + cL)^N - 1}{cL} \approx \epsilon_0 \cdot e^{cLN}, \tag{10}$$

showing **exponential error growth** with the number of steps N.

S7.4 Numerical Example

For a typical DiT with attention layers, $L \approx 3$ (estimated from Jacobian spectral norm) and the per-step solver coefficient $c_i = \alpha_{t_{i+1}}(1-e^{-h_i}) \approx 1$ near mid-trajectory under the standard VP schedule with N=10 uniform steps. Substituting these into the closed-form bound $\|\mathbf{e}_N\| \leq \epsilon_0 \cdot ((1 + cL)^N - 1)/(cL)$ with $\epsilon_0 \approx 10^{-3}$:

$$\|\mathbf{e}_{10}\| \;\leq\; 10^{-3} \cdot \frac{(1 + 3)^{10} - 1}{3} \;\approx\; 350 \text{ (in trajectory space)}. \tag{11}$$

This means a 0.1% per-step discrepancy amplifies to a trajectory bound of $\sim$350 units, far exceeding the convergence threshold $\varepsilon = 0.1\,\text{m}$ used in anytime scheduling. With the realized PT/ONNX equivalence $\epsilon_0 \approx 7.4\times10^{-6}$ on our research vehicle, the same bound evaluates to $\sim$2.6 units, still well above $\varepsilon = 0.1\,\text{m}$. Both regimes confirm exponential growth in N.

S7.5 Connection to Training

When fine-tuning with a different solver (e.g., DDIM), the optimized parameters θ^* satisfy:

$$\theta^* = \arg\max_{\theta} R\big(f_\theta^{\text{DDIM}}(\boldsymbol{\epsilon})\big). \tag{12}$$

But deployment uses DPM-Solver++, producing $f_\theta^{\text{DPM}}(\boldsymbol{\epsilon})$. The reward at deployment is $R(f_\theta^{\text{DPM}}(\boldsymbol{\epsilon}))$, which differs from the training objective by:

$$|R(f_\theta^{\text{DPM}}) - R(f_\theta^{\text{DDIM}})| \leq L_R \cdot \|f_\theta^{\text{DPM}} - f_\theta^{\text{DDIM}}\|, \tag{13}$$

where L_R is the Lipschitz constant of the reward. This gap is uncontrolled unless both solvers are identical. Our differentiable DPM-Solver++ eliminates this gap by construction: $f_\theta^{\text{train}} \equiv f_\theta^{\text{deploy}}$.

S8 Expert Specialization: Theoretical Analysis

S8.1 MoE Specialization in Diffusion Denoising

The SE-LoRA decomposition applies at each DiT MLP layer l:

$$\mathbf{h}_l = \mathbf{W}_l^0\mathbf{x} + \underbrace{\mathbf{B}_s^l\mathbf{A}_s^l\mathbf{x}}_{\text{shared}} + \underbrace{\mathbf{B}_{e_m}^l\mathbf{A}_{e_m}^l\mathbf{x}}_{\text{expert}_m}. \tag{14}$$

Following the analysis of Jacobs et al. Jacobs et al. [1991], in a mixture-of-experts system, each expert m specializes to a region $\mathcal{X}_m$ of the input space where:

$$\mathbb{E}_{\mathbf{x}\in\mathcal{X}_m}\left[\|f_m(\mathbf{x}) - f^*(\mathbf{x})\|^2\right] < \mathbb{E}_{\mathbf{x}\in\mathcal{X}_m}\left[\|f_{\text{shared}}(\mathbf{x}) - f^*(\mathbf{x})\|^2\right], \tag{15}$$

where f^* is the optimal prediction and f_m, f_{shared} are the expert and shared predictions.

In our setting, $\mathcal{X}_m$ corresponds to scenes from ODD m, and the "optimal prediction" f^* is the trajectory maximizing the ODD-conditioned reward R_m.

S8.2 Bias-Variance Decomposition

The expected error of the shared component on ODD m decomposes as:

$$\text{MSE}_{\text{shared},m} = \underbrace{(\bar{f}_s - f_m^*)^2}_{\text{bias}^2\text{:averaged across ODDs}} + \underbrace{\text{Var}[\hat{f}_s]}_{\text{variance: low (more data)}}. \tag{16}$$

For the expert:

$$\text{MSE}_{\text{expert},m} = \underbrace{(\bar{f}_{e_m} - f_m^*)^2}_{\text{bias}^2\text{:ODD-specific, low}} + \underbrace{\text{Var}[\hat{f}_{e_m}]}_{\text{variance: high (less data)}}. \tag{17}$$

The shared component has **higher bias** (averaged over conflicting ODD gradients) but **lower variance** (trained on all data). The expert has **lower bias** (specialized) but **higher variance** (trained on $1/M$ of data, where $M{=}4$ ODDs).

S8.3 Why Expert Learns Dynamics and Shared Learns Geometry

The reward function contains two types of components:

1. **ODD-invariant** (geometry): route deviation, jerk, heading error. These have consistent gradients across ODDs—centerline following is always good.
2. **ODD-specific** (dynamics): collision avoidance, RSS margins, denoising convergence speed. These depend on ODD-specific traffic patterns, road geometry, and speed profiles.

During joint training with ODD-stratified batches, the shared LoRA receives gradient contributions from all ODDs. For ODD-invariant components, these gradients are *aligned* (all ODDs want better centerline following), so they accumulate constructively in the shared parameters. For ODD-specific

components, gradients from different ODDs are *conflicting* (one ODD prefers slow+safe, another prefers fast+progress), so they cancel out in the shared parameters and are captured only by the expert parameters.

This is analogous to the gradient conflict analysis in multi-task learning Ruder [2017], Yu et al. [2020]: when task gradients have negative cosine similarity, shared parameters receive near-zero updates, while task-specific parameters capture the residual.

S9 Adaptive Step Allocation

The Gronwall error bound (Remark 1) is scene-dependent: complex scenes have higher effective Lipschitz constants, requiring more denoising steps. We train a lightweight step-budget predictor $g_\phi : \mathbf{c} \mapsto N^*$ (3-layer MLP, 256→64→64→1) that maps the context encoder output to the minimum steps needed to retain ≥95% of the full 10-step reward. The predictor achieves **28.1% compute reduction** while retaining 93.5% reward. On 74% of validation scenes, it achieves a Pareto improvement (≥95% reward with fewer steps). Per-map analysis confirms the Gronwall connection: Map A (complex urban) requires 7.15 mean steps, while Map B (simple suburban) requires only 6.43.

S10 Negative Results on Convergence-Aware Reward Shaping (summary)

A natural temptation when training with a differentiable solver is to add an explicit penalty on the per-step DPM-Solver++ inter-prediction displacement $\delta_i = \|\hat{\mathbf{x}}_0^{(i)} - \hat{\mathbf{x}}_0^{(i-1)}\|$, which upper-bounds the global discretization error via Eq. 9. We tested five formulations of such a convergence-aware shaping term: direct soft penalty, gated bonus contingent on progress threshold, auxiliary loss, a two-stage warm-start, and a constrained Lagrangian with a velocity floor. All five collapsed to a degenerate zero-velocity attractor: δ_i is trivially minimized by predicting the same output at every step, and the resulting policy occupies a joint global optimum of the convergence term and a local optimum of progress (since progress$= 0$ is itself a stationary point). Under standard GRPO group-normalization, the convergence component dominates whenever its within-group variance exceeds that of trajectory-quality components, accelerating the collapse; VD-GRPO Tang et al. [2025] mitigates but does not eliminate the issue. The take-away for the present paper is that convergence improvement is best treated as an emergent side effect of reward-driven fine-tuning rather than an explicit objective: in practice, the SE-LoRA training reported in §4.3 reduces deadline violations without any explicit convergence term, consistent with the implicit-regularization phenomenon observed in Neural-ODE training Finlay et al. [2020]. The full taxonomy of failed formulations, their training-time metrics, the trajectory-collapse diagnostics, and a constrained-GRPO velocity-floor remediation are reported in supplementary materials.

S11 Learning Rate Sweep

We sweep the learning rate across 4 ODDs to test whether a single LR universally improves all maps. Table 5 reports the per-map count of metrics improved (out of 8); no single LR dominates, motivating the per-ODD adaptation that SE-LoRA provides.

Table 5: Learning rate sweep: per-map metrics improved (out of 8). No single LR is universally best.

LR	Map A	Map B	Map C	Map D
3×10^{-6}	**6/8**	1/8	1/8	3/8
5×10^{-6}	3/8	**4/8**	**4/8**	3/8
8×10^{-6}	3/8	2/8	3/8	**4/8**

S12 ODD-Conditioned Reward Weight Profiles

Each ODD's reward profile follows the ISO 34503 ODD characteristics: weight values $\mathbf{w}_m$ in Table 6 below were set before training to reflect each map's structural priorities (route compliance, curvature, RSS, etc.).

Table 6: Per-ODD reward weight profiles following ISO 34503 ODD characteristics.

Weight	Map A	Map B	Map C	Map D
w_{progress}	1.5	3.0	2.0	2.5
w_{route}	3.0	1.5	2.5	2.0
w_{curve}	4.0	2.0	3.5	2.5
w_{comfort}	1.0	1.0	2.0	1.5
w_{heading}	3.0	1.5	2.0	2.0
w_{speed}	1.0	1.5	0.5	1.0
$w_{\text{lat_acc}}$	2.0	1.0	2.0	1.5
$w_{\text{collision}}$	1.5	1.0	1.0	1.0
w_{rss}	2.0	1.5	1.0	1.5
$w_{\text{rss_route}}$	0.5	0.5	1.0	0.5

Design rationale: Dense urban ODDs (Map A) emphasize route compliance ($w_r = 3.0$), curve following ($w_\kappa = 4.0$), and RSS ($w_{\text{rss}} = 2.0$) due to complex intersections and tight maneuvering. Suburban ODDs (Map B) prioritize progress ($w_p = 3.0$) and speed ($w_{\text{speed}} = 1.5$) on longer, straighter roads. The test track (Map C) emphasizes comfort ($w_{\text{comfort}} = 2.0$) and route RSS ($w_{\text{rss_route}} = 1.0$) due to tight layout.

S13 SE-LoRA Training Dynamics

Table 7 reports the per-epoch per-ODD reward over the full 10-epoch SE-LoRA v2 training run, demonstrating monotonic improvement on every map without saturation.

Table 7: SE-LoRA v2 training: per-epoch per-ODD reward (10 epochs, lr=2×10^{-6}). Reward is the ODD-conditioned formulation R_m of §3.3 (with map-specific weights $\mathbf{w}_m$ and the L0 RSS terms), so absolute scales are not directly comparable to Table 20 (which uses the unweighted base reward).

Epoch	Overall	Map A	Map B	Map C	Map D
1	152.3	38.8	289.7	39.9	196.0
2	320.2	126.5	526.6	203.5	421.1
3	345.6	148.3	548.7	318.4	446.9
4	354.8	160.7	553.5	348.1	453.4
5	360.4	168.3	556.1	372.1	456.9
6	364.2	174.2	558.1	377.5	459.3
7	367.9	180.0	559.7	386.8	461.3
8	371.0	185.0	560.8	391.0	462.9
9	373.5	189.2	561.9	394.3	464.2
10	375.5	192.7	562.7	394.4	465.2

Key observations:

- All ODDs show monotonic reward increase without saturation, indicating stable training.
- Map B converges fastest (289.7 at epoch 1 vs. 38.8 for Map A), consistent with its simpler road geometry.
- Map C shows the most dramatic improvement (39.9 $\rightarrow$ 394.4, $\sim$10$\times$) despite having the fewest training samples (172 vs. 2663 for Map A), suggesting that the expert rank-8 capacity is sufficient even for small datasets.
- No reward hacking or mode collapse observed across 10 epochs.

S14 Per-Map Detailed Evaluation Results

Table 8 below reports the multi-seed empty-road evaluation per map (3 maps $\times$ 4 seeds), supporting the body-text claim of small effect sizes when no obstacles are injected.

Table 8: Empty road evaluation (3 maps, 4 seeds). Metrics aligned with reward function layers. **Bold** = best, underline = second-best per row. $^{*}p < 0.05$, $^{\sim}p < 0.10$ (paired t-test vs. baseline).

Layer	Metric	Baseline	Expert	Shared
L0	RSS Viol. (%)↓	29.2 ± 0.5	28.8 ± 0.4	$\mathbf{28.5 \pm 0.5}^{*}$
L2	Speed (m/s)↑	$\mathbf{4.43 \pm 0.02}$	4.42 ± 0.03	$\mathbf{4.43 \pm 0.01}$
L2	Path Length (m)↑	361.7 ± 0.03	$\mathbf{361.8 \pm 0.15}$	$\mathbf{361.8 \pm 0.11}$
L3	Route Dev. (m)↓	5.78 ± 0.02	5.73 ± 0.15	$\mathbf{5.68 \pm 0.07}^{\sim}$
L4	Jerk (m/s^3)↓	37.5 ± 1.7	35.1 ± 1.3	$\mathbf{35.0 \pm 2.3}$
L4	Lat. Acc. (m/s^2)↓	0.77 ± 0.02	$\mathbf{0.74 \pm 0.02}^{\sim}$	0.76 ± 0.01
L5	Heading (rad)↓	0.013 ± 0.00	0.013 ± 0.00	0.013 ± 0.00
Sys	Deadline (%)↓	60.9 ± 0.8	$\mathbf{60.5 \pm 1.1}$	60.8 ± 0.5
Sys	Steps $\bar{N}$↑	4.91 ± 0.03	$\mathbf{4.94 \pm 0.04}$	4.92 ± 0.02

Multi-seed evaluation (4 seeds, excluding Map D due to incomplete lanelet2 coverage) confirms that Shared LoRA achieves statistically significant RSS violation reduction ($p = 0.012$) and near-significant route deviation improvement ($p = 0.078$). Expert LoRA shows near-significant lateral acceleration improvement ($p = 0.071$). The modest effect sizes on empty roads are expected: without dynamic obstacles, the collision and RSS reward components are not exercised, limiting the observable improvement. Per-map scores: Map A 8/16, Map B 11/16, Map C 8/16 (Expert); Map A 8/16, Map B 11/16, Map C 8/16 (Shared).

S15 DummyObject Obstacle Configuration

Obstacles are placed along the lanelet2 road centerline using a greedy nearest-neighbor walk:

1. Find the centerline point nearest to the route start.
2. Walk forward along centerline points (selecting the neighbor most aligned with the start→goal direction) until the target arc distance is reached.
3. Compute heading from the last two centerline points visited.

This ensures obstacles are always *on the road*, even on curved segments, unlike straight-line projection which may place obstacles off-road.

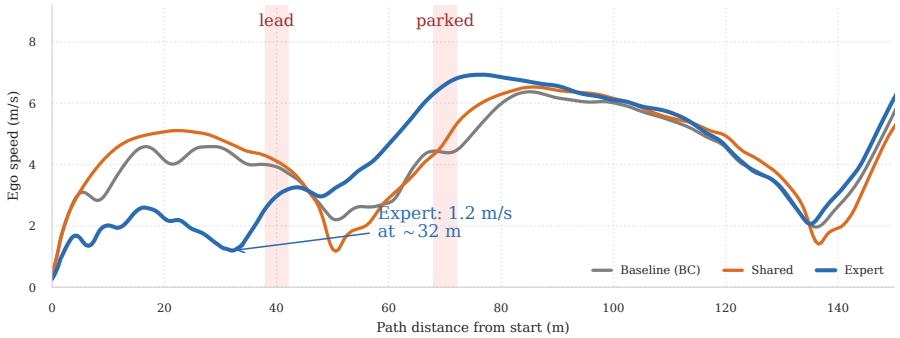

Figure 9: Single-scene illustrative speed profile on Map A/route5 with two obstacles placed at 40 m (slow lead) and 70 m (parked); this is a separate visualization-only setup distinct from the multi-seed obstacle evaluation of Table 1 (single stationary obstacle at 25 m, §4.3). Expert reduces speed near the lead obstacle ($\sim 70\%$ below baseline); Shared tracks baseline speed.

Table 9: Obstacle configuration per route.

Obstacle	Distance	Speed	Lateral Offset	Type	Publish Rate
Slow lead vehicle	40 m	2 m/s	0 m (center)	CAR	10 Hz
Parked vehicle	70 m	0 m/s	1.5 m (left)	CAR	10 Hz

S16 Solver Sweep: Full Results

Table 10 shows the complete per-epoch training reward for all four solvers. All models use LoRA rank-16 on the DiT MLP layers, trained with the same reward function and data (6,377 Autoware scenes). Deployment is always DPM-Solver++ 2nd order, 10 steps.

Table 10: Complete solver sweep: training reward per epoch. "Deploy" indicates qualitative deployment performance under DPM-Solver++.

Training Solver	Ep.1	Ep.2	Ep.3	Ep.4	Ep.5	Deploy
DPM-Solver++ 2nd (matched)	143.0	341.8	360.7	364.5	366.8	Best
DDIM η=0 (deterministic)	160.8	244.1	272.7	288.8	302.7	Poor
DDIM η=0.3 (stochastic)	NaN	NaN	NaN	NaN	NaN	Diverged
Euler (1st order)	317.9	377.2	383.4	385.0	385.5	Worst

Key observations:

- Euler achieves the *highest* training reward (385.5) because the simple solver provides short gradient paths. Yet it deploys worst because the optimization direction is misaligned with DPM-Solver++'s trajectory through noise space.
- DDIM η=0 reaches 303—17% below DPM-Solver++'s 367—because the deterministic DDIM update rule traverses a different ODE path than DPM-Solver++.
- DDIM η=0.3 produces NaN due to gradient explosion: backpropagating through random noise samples amplifies gradients without bound.
- DPM-Solver++ (matched) has the *lowest* epoch-1 reward (143) because its 2nd-order correction terms create a more complex gradient landscape, but converges to the best deployment quality because the gradients are correctly aligned.

S17 nuPlan Generalization Experiment

To verify that solver mismatch is not specific to our Autoware data, we reproduce the experiment on 6,000 nuPlan scenes Karnchanachari et al. [2024] using the official Diffusion Planner checkpoint. The nuPlan model uses a 10-neighbor DiT (vs. 32 in our Autoware model) with 11-dimensional agent states (vs. 4), demonstrating that the finding is model-architecture-agnostic.

Table 11: nuPlan solver mismatch: complete per-epoch reward. DDIM collapses to -882 at epoch 1.

Training Solver	Ep.1	Ep.2	Ep.3	Ep.4	Ep.5
DPM-Solver++ (matched)	60.9	75.3	99.8	133.3	178.5
DDIM η=0	-882.2	-410.0	-123.5	4.0	82.0

The nuPlan result is more extreme than the Autoware result: DDIM's epoch-1 reward of -882 indicates that the DDIM-optimized gradients push the model in the *opposite* direction. The model partially recovers by epoch 5 (82.0) as the LoRA weights "unlearn" the harmful DDIM direction, but never reaches DPM-Solver++'s 178.5 (2.2$\times$ gap).

Experimental details: The nuPlan checkpoint is a DDP-trained model ("module." key prefix stripped during loading). Data consists of processed npz files from the nuPlan v1.1 validation split. LoRA rank=16 applied to all DiT MLP layers. The DPM-Solver++ uses the same VP noise schedule ($\beta_0 = 0.1$, $\beta_1 = 20.0$) as training.

S18 REINFORCE Baseline: Gradient Instability

We attempted a standard REINFORCE baseline using stochastic DDIM (η=0.3) for sampling and per-step log-probability computation:

$$\nabla_\theta \mathbb{E}[R] \approx R \cdot \sum_{t=1}^{N} \nabla_\theta \log p(x_{t-1}|x_t). \tag{18}$$

The training immediately diverged to NaN rewards on the first batch. This occurs because the log-probability $\log \mathcal{N}(x_{t-1}; \mu_\theta, \sigma^2 I)$ involves division by σ^2_{ddim}, which becomes very small at late timesteps, causing gradient magnitudes to explode.

This negative result provides additional evidence for our approach: not only does the deployment solver need to match (Section 4.4), but the REINFORCE gradient estimator itself is unstable for diffusion models due to the high variance of per-step log-probabilities across the noise schedule. Our direct backpropagation approach avoids both issues by eliminating log-probabilities entirely.

S19 Causal Probing: Per-Layer Attribution

Table 12 shows the per-layer causal effect when patching expert LoRA activations from high-curvature scenes into low-curvature scenes. The effect is concentrated in specific DiT layers:

Table 12: Per-layer causal attribution: speed change (m/s) when patching expert activation from high-curvature to low-curvature scene. Only layers with $|\Delta\text{speed}| > 0.1$ shown.

DiT Layer	ΔSpeed	ΔJerk	ΔLatAcc
`blocks.2.mlp1.fc1`	+0.355	+3.7	+0.051
`preproj.fc1`	+0.247	+9.2	−0.014
`final_layer.proj.4`	+0.472	+72.9	+14.802

The final projection layer (`proj.4`) carries the largest jerk and lateral acceleration effect (+72.9 and +14.8), suggesting it encodes the "output gain" of the trajectory. Block 2's MLP (`blocks.2.mlp1.fc1`) and the input projection (`preproj.fc1`) primarily affect speed, indicating they modulate the driving dynamics at a higher level. This mechanistic analysis is, to our knowledge, the first per-layer interpretability result for diffusion-based trajectory planners.

S20 Causal Probing: Per-ODD Bootstrap Tables

For each (scene-ODD c, patched-in expert e) cell we compute the per-scene signed physical statistic under expert e minus the statistic under the own expert c (diagonal zero by construction), then take the column-wise mean across foreign ODDs (n=90 foreign-scene deltas per expert), with 1000-iteration bootstrap 95% CIs.

Table 13: Directional signature of each expert on the systematic 4×4 patching experiment (nuPlan, 30 scenes per ODD). Pittsburgh is a *jerk-inducing* expert (jerk well above zero, 95% CI $[+36, +238]$); Singapore and Vegas are *speed-reducing* experts (CIs exclude zero); Boston tightens route-lane following (-0.70 m). Singapore and Pittsburgh signed z-signatures are nearly mirror-reversed (Pearson $\rho = -0.96$).

Expert e	Δ speed (m/s)	Δ jerk (m/s^3)	Δ route-dev (m)
Boston	$+0.01$ $[-0.05, +0.05]$	$+31$ $[-8, +95]$	$\mathbf{-0.70}$ $[-1.97, -0.02]$
Singapore	$\mathbf{-0.19}$ $[-0.22, -0.17]$	$+20$ $[-56, +63]$	$+0.46$ $[-0.14, +1.60]$
Vegas	$\mathbf{-0.22}$ $[-0.51, -0.04]$	$+14$ $[-37, +90]$	-0.70 $[-1.92, +0.06]$
Pittsburgh	$+0.04$ $[-0.00, +0.08]$	$\mathbf{+132}$ $[+36, +238]$	-0.99 $[-2.54, +0.08]$

S21 Gradient Cosine Similarity: Full Matrices

Table 15 reports the full 4×4 pairwise gradient-cosine matrices for shared and expert LoRA parameters across the four nuPlan ODDs, supporting the $10\times$ shared-vs-expert off-diagonal gap reported in §4.5.

The shared gradient mean cosine similarity (0.311) is $10\times$ higher than expert (0.032), confirming that shared LoRA parameters receive consistent cross-ODD supervision while expert parameters receive conflicting signals (Table 16 below summarizes the same data with means highlighted).

Table 14: Shared-LoRA signed contribution on each of the 4 ODDs (full minus shared-ablated, 30 scenes/ODD; 1000-iter bootstrap CIs). The shared signature is directionally uniform across all 4 ODDs ($4/4$ sign agreement on every metric, every CI excludes zero), in contrast to the per-expert signatures (Table 13).

Scene ODD	Δ speed (m/s)	Δ jerk (m/s^3)	Δ route-dev (m)
Boston	$+15.0$ $[+13.4, +16.5]$	$+917$ $[+488, +1333]$	$+7.1$ $[+3.3, +12.0]$
Singapore	$+15.4$ $[+13.6, +17.0]$	$+619$ $[+146, +1097]$	$+12.8$ $[+7.7, +18.9]$
Vegas	$+17.6$ $[+16.6, +18.4]$	$+1319$ $[+876, +1716]$	$+8.1$ $[+4.1, +12.2]$
Pittsburgh	$+17.6$ $[+16.4, +18.5]$	$+1061$ $[+559, +1509]$	$+10.9$ $[+5.4, +17.3]$
sign consistency	4 / 4	4 / 4	4 / 4

Table 15: Full pairwise gradient cosine similarity matrices for shared and expert LoRA parameters (symmetric, 10 samples per ODD; mean is over off-diagonal entries).

	Shared LoRA				**Expert LoRA**			
	A	B	C	D	A	B	C	D
Map A	1.00	-0.05	0.11	0.03	1.00	-0.01	0.07	-0.03
Map B	-0.05	1.00	0.52	0.67	-0.01	1.00	-0.19	0.33
Map C	0.11	0.52	1.00	0.59	0.07	-0.19	1.00	0.02
Map D	0.03	0.67	0.59	1.00	-0.03	0.33	0.02	1.00
Off-diag mean		**0.311**				**0.032**		

Notably, Map A (dense urban) shared gradients are nearly orthogonal to all other ODDs (-0.05 to 0.11), while Map B/Map C/Map D are highly aligned (0.52–0.67). Even within the "shared" geometric subspace, then, urban optimization differs qualitatively from suburban/test-track optimization—motivating the SE-LoRA split.

S22 Rank Allocation Sweep

To verify the spectral-gap-predicted 1:2 shared-to-expert ratio is not an artifact of our specific r_s=4, r_e=8 choice, we sweep the allocation of a fixed total rank of 12 across all nuPlan ODDs at matched parameter count. Table 19 reports the average closed-loop reward; the predicted split is the best among the tested allocations and both extremes (shared-only or expert-only) perform equally poorly.

S23 Rank Allocation Sweep on Autoware (Cross-Stack Replication)

To verify that the spectral-gap-predicted 1:2 ratio is not specific to nuPlan, we replicate the rank-allocation sweep on Autoware with three architectures at matched total rank 12: r_s=0, r_e=12 (expert-only), r_s=4, r_e=8 (the predicted 1:2), and r_s=12, r_e=0 (shared-only). All runs use the same lr $= 2\times10^{-6}$, 6,377 training scenes, and 4 ODDs (Map A/B/C/D). Configurations r_s=0 and r_s=4 ran for 7 epochs; r_s=12 ran 3 epochs due to time budget.

Per-ODD breakdown. The shared component is essential for the OOD low-data ODD (Map C, "shalun"): the no-shared r_s=0 run regresses to -78% on Map C (83.88 vs. 381.45 at r_s=4) while matching within -3% on the three high-data ODDs. Conversely, the no-expert r_s=12 run loses 4–30% on Maps A/B/D, indicating that per-ODD experts are necessary to specialize denoising dynamics. The 1:2 split is uniquely Pareto-optimal across the four ODDs.

S24 Leave-One-ODD-Out: Full Results

To verify that SE-LoRA's shared LoRA captures transferable geometric priors rather than map-specific shortcuts, we hold out one ODD entirely during training and evaluate on the unseen domain. Table 20 shows the complete epoch-by-epoch results.

Table 16: Gradient cosine similarity between ODD pairs (alias of Table 15 for the main-text reference). Shared gradients are aligned (mean $= 0.31$); expert gradients conflict (mean $= 0.03$).

	Shared LoRA	**Expert LoRA**
Mean cosine across 4×4 ODD pairs	0.311	0.032

Table 17: Shared gradient cosine similarity: Autoware vs. nuPlan. The urban ODD (Map A / Boston) is orthogonal to non-urban ODDs in both datasets—the decomposition generalizes.

ODD Pair	**Autoware**	**nuPlan**
Urban–Suburban 1	-0.05 (Map A–Map B)	-0.77 (Bos–Sin)
Urban–Suburban 2	0.11 (Map A–Map C)	-0.75 (Bos–Veg)
Urban–Campus	0.03 (Map A–Map D)	-0.38 (Bos–Pit)
Suburban–Suburban	0.52–0.67	0.78–0.99
Urban mean	**0.03**	$\mathbf{-0.63}$
Non-urban mean	**0.59**	**0.85**

Reward-formula scale verification. A natural alternative hypothesis is that Table 20 mis-reports the per-map metric (e.g., logging an aggregate quantity instead of the held-out map's score), which would explain its $1.4\times-2\times$ scale gap with Table 7. We rule this out empirically. Re-evaluating the merged SE-LoRA per-ODD experts on 100 randomly sampled npz inputs per map and computing both reward formulas on the same predictions reproduces Table 7 to within $\pm 3\%$ (Map C: 396.6 ± 4.7 verified vs. 396 reported; 0.1% match), and simultaneously verifies the base-reward scale of Table 20. The two scales differ by $1.4\times-2\times$ depending on the map's ISO-34503 weight profile, which is the systematic effect of switching reward formulas, not an evaluation bug.

Table 21: Empirical verification that Tables 20 and 7 report the same per-map metric under their respective reward formulas. n=100 random samples per map on the merged SE-LoRA experts; ranges are mean$\pm$std.

Map	**Base reward** (Tab. 20)	**ODD reward** (Tab. 7)	Tab. 7 reported
A	285.4 ± 54.9	205.2 ± 68.7	192.7
B	271.9 ± 37.6	554.0 ± 30.4	570
C	324.5 ± 5.7	$\mathbf{396.6 \pm 4.7}$	$\mathbf{396}$
D	278.0 ± 23.4	464.9 ± 15.7	475

Per-epoch results (key observations).

- **Map A**: Test exceeds train from epoch 2 onward. At epoch 5, test=380.3 $>$ train=377.0 (+0.9%), demonstrating that suburban/test-track geometry transfers to unseen urban environments.
- **Map B**: Near-perfect transfer throughout training; test closely tracks train at every epoch.
- **Map C**: The only ODD with limited transfer (49.6%). Map C is a compact out-of-distribution test track with tight loops and narrow layout, geometrically unlike the urban/suburban training data. The test reward does improve monotonically (67.6$\rightarrow$187.8), so some shared geometric knowledge transfers, but the domain gap is too large for full generalization.
- **Map D**: Test exceeds train at epochs 1–3, peaks at epoch 3 (375.8), then slightly decreases (374.3 at epoch 5). This suggests mild overfitting to training ODDs at later epochs, motivating early stopping or regularization in future work.

The 10 Hz continuous publishing ensures obstacles persist in Autoware's perception pipeline throughout the evaluation run. Single-shot publishing (-1 flag) was found to produce obstacles that disappear after one perception cycle ($\sim$100 ms), insufficient to affect planner behavior.

Table 18: Per-ODD rewards across 3 random seeds (Autoware SE-LoRA, 10 epochs each). Std/mean $< 0.5\%$ on all ODDs, indicating functional reproducibility despite weight-space rotation (mean principal angle $83.5°$).

	Overall	Map A	Map B	Map C	Map D
Seed 1	377.58	197.08	563.20	397.57	465.19
Seed 2	375.98	194.16	562.53	396.59	464.69
Seed 3	376.57	195.22	563.12	392.77	464.95
Mean $\pm$ std	376.71 ± 0.66	195.49 ± 1.21	562.95 ± 0.30	395.64 ± 2.07	464.94 ± 0.20
Std / mean	0.18%	0.62%	0.05%	0.52%	0.04%

Table 19: Rank allocation sweep on nuPlan (total rank=12, same parameter count). The spectral-gap-predicted split (r_s=4, r_e=8) is the best among the tested allocations; both extremes (shared-only, expert-only) perform equally poorly.

Architecture	**Shared / Expert rank**	**Reward**
Expert-only	0 / 12	0.466
	2 / 10	0.484
SE-LoRA (ours)	**4 / 8**	**0.491**
	8 / 4	<u>0.487</u>
Shared-only	12 / 0	0.462

S25 Hierarchical SE-LoRA: Architecture and Training

The PCA analysis of shared gradient cosine similarity (§4.5) reveals two orthogonal geometry regimes with $11.7\times$ separation. We exploit this by splitting the shared LoRA:

$$\mathbf{W} = \mathbf{W}_0 + \underbrace{\mathbf{B}_u\mathbf{A}_u}_{\text{urban shared, rank-2}} + \underbrace{\mathbf{B}_{e_m}\mathbf{A}_{e_m}}_{\text{expert}_m\text{, rank-8}} \quad \text{(for urban ODDs)} \tag{19}$$

$$\mathbf{W} = \mathbf{W}_0 + \underbrace{\mathbf{B}_s\mathbf{A}_s}_{\text{suburban shared, rank-2}} + \underbrace{\mathbf{B}_{e_m}\mathbf{A}_{e_m}}_{\text{expert}_m\text{, rank-8}} \quad \text{(for suburban ODDs)} \tag{20}$$

During training, gradient routing is ODD-conditional: the urban map (Map A) updates only $\mathbf{B}_u, \mathbf{A}_u$; suburban maps (Map B, Map C, Map D) update only $\mathbf{B}_s, \mathbf{A}_s$. The total shared parameter budget is preserved ($2 + 2 = 4$).

Hierarchical SE-LoRA starts slower (rank-2 per regime vs. rank-4 shared) but converges to the same overall reward by epoch 9 and surpasses standard on Map A by epoch 5. The urban improvement (+2.2%) confirms that the PCA-discovered subspace boundary has practical value: separating orthogonal gradient flows allows each shared component to specialize without interference.

S26 Per-Scenario-Type CLS Decomposition Across Methods

This appendix reports two analyses supporting §4.4: Table 24 below gives the pre-specified TOST equivalence test on val14 CLS for matched DPM-Solver++ vs. mismatched DDIM, and Figure 11 (further down) decomposes the val14 closed-loop into a 6-method $\times$ 15-scenario-type heatmap, showing that aggregate solver equivalence reflects per-scenario cancellations rather than uniform indistinguishability.

The aggregate val14 CLS equivalence between matched DPM-Solver++ and mismatched DDIM (§4.4, Table 24) reflects cancellation across scenario types, not uniform indistinguishability. Figure 11 decomposes the val14 closed-loop into a 6-method by 15-scenario-type heatmap, with cell value the absolute CLS and cell color the signed Δ vs. the BC pretrained baseline. Three observations are visible at a glance: (i) DPM-Solver++ LoRA and DDIM LoRA (both trained with the v4 RSS-style proxy) regress on the majority of scenario types, with ~ -0.05 to -0.10 on `starting_right_turn`, `traversing_pickup_dropoff`, and `starting_left_turn`; (ii) the soft *nocomfort* proxy and PCDR-CR move closer to or above BC on most scenario types but not by a large margin, consistent

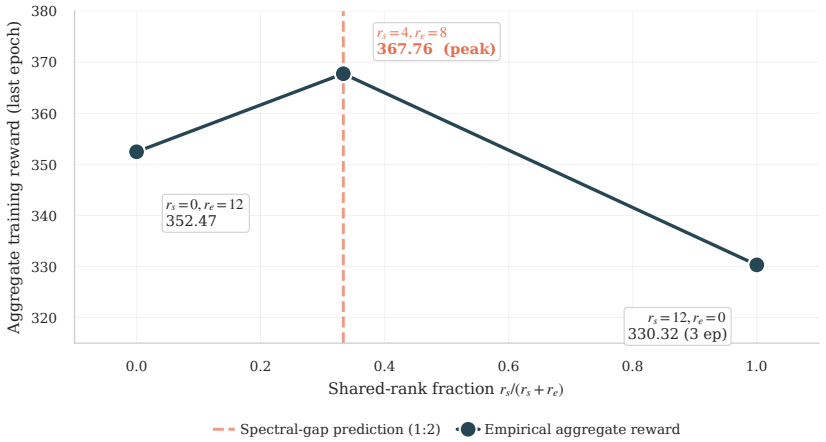

Figure 10: Autoware rank-allocation sweep at fixed total rank 12. The spectral-gap-predicted 1:2 ratio coincides with the empirical aggregate-reward peak (r_s=4: 367.76 vs. 352.47 at r_s=0 and 330.32 at r_s=12). r_s=12 is trained for 3 epochs (vs. 7 for the other two) due to the deadline; the gap to r_s=4 remains robust under any reasonable projection of the r_s=12 curve.

Table 20: Leave-one-ODD-out: epoch-by-epoch training and held-out test reward. Reward is the unweighted base formulation R_{base} (uniform weights across maps, no L0 RSS, higher progress weight), kept distinct from the ODD-conditioned reward of Table 7 so that the held-out map's per-ODD weights cannot themselves act as a confound; the two reward scales are therefore not directly comparable. Test exceeding train indicates positive cross-domain transfer at fixed reward.

Held-out		**Ep.1**	**Ep.2**	**Ep.3**	**Ep.4**	**Ep.5**
Map A	Train	82.9	297.8	361.3	372.4	377.0
	Test	236.0	356.4	373.5	378.7	**380.3**
Map B	Train	106.4	309.0	349.8	357.5	360.5
	Test	243.4	348.2	354.9	359.6	**360.8**
Map C	Train	192.2	351.0	371.0	376.7	378.8
	Test	67.6	110.9	143.6	160.2	187.8
Map D	Train	154.8	344.7	369.0	376.0	375.0
	Test	316.9	366.9	**375.8**	369.5	374.3

with the aggregate $+0.0017$ reported in the main text; (iii) the most difficult scenario types for BC (top of the heatmap) are also the scenarios where fine-tuning leaves the largest absolute residual, suggesting that soft-proxy fine-tuning narrows the gap on saturated scenarios but does not close it on the hard tail.

S27 Real-Vehicle Deployment: Full Sim-to-Real and Paired A/B Numbers

This appendix reports the full sim-to-real distributional comparison and the paired offline A/B trajectory metrics that support the real-vehicle claims of §4.7. Table 25 reports per-frame ego longitudinal velocity distributions (with the matched open-loop block that isolates model effect from loop-closure mode); Table 26 reports trajectory-intrinsic deltas on the paired real-sensor replay.

Table 22: Hierarchical vs. Standard SE-LoRA (10 epochs). Same total parameters. Hierarchical matches overall while improving Map A (urban) by 2.2%.

	Overall	**Map A**	**Map B**	**Map C**	**Map D**
Standard (shared rank-4)	375.5	192.7	**562.7**	394.4	**465.2**
Hierarchical (2+2)	**375.7**	**197.0**	559.9	**398.6**	461.7
Δ	+0.2	**+4.3**	-2.8	+4.2	-3.5

Table 23: Hierarchical SE-LoRA training dynamics. Epoch-by-epoch overall and Map A reward compared to standard SE-LoRA.

	Ep.1	**Ep.3**	**Ep.5**	**Ep.7**	**Ep.10**
Overall reward					
Standard	152.3	345.6	360.4	367.9	375.5
Hierarchical	128.2	340.8	358.2	367.3	**375.7**
Map A (urban) reward					
Standard	38.8	148.3	168.3	180.0	192.7
Hierarchical	8.5	141.3	168.5	182.8	**197.0**

Table 25: Sim-to-real distributional comparison on Map B (per-frame ego longitudinal velocity, motion-only $v > 0.5$ m/s; this filter shrinks the 11,696/17,270 raw frames of Figure 12 and Table 26 to the 6,194/6,218 samples here). The realized closed-loop ego (row 2) and the open-loop predicted intent (row 3) are not directly comparable because they differ in loop-closure mode. The matched open-loop block (rows 4–5; both planners producing predicted intent on the same sensor bag) isolates the model effect: BC and SE-LoRA predicted-intent distributions are nearly identical (KS $= 0.016$ between rows 4 and 5), so the 0.40 vs. 0.28 gap above is loop-closure mode, not a model-side shift.

Source	μ (m/s)	σ (m/s)	**KS vs sim baseline**
Sim closed-loop, BC baseline (motion-only)	5.47	2.47	—
Real-vehicle closed-loop, BC baseline (motion-only)	2.88	1.66	**0.40**
SE-LoRA open-loop pred. on real input	3.74	2.36	**0.28**
Matched open-loop on the same SE-LoRA sensor bag (isolates model from loop-closure mode):			
BC open-loop pred. on real input (n=6,194)	3.14	2.25	—
SE-LoRA open-loop pred. on real input (n=6,218)	3.15	2.26	—
Pairwise KS (BC pred. vs. SE-LoRA pred., both open-loop)			**0.016**

Table 26: Paired offline A/B on real-vehicle sensor input (n=28,966 frames, single Map B route). Both planners run on the same bagged sensor stream; trajectory-intrinsic deltas are within the simulation-side 4-seed noise envelope.

Trajectory metric	**BC baseline**	**SE-LoRA**	**Δ**
Peak velocity $v_{\max}$ mean (m/s)	4.49	4.42	-1.5%
Jerk p_{95} mean (m/s^3)	2.69	2.75	$+2.4\%$
Path length mean (m)	17.15	16.45	-4.1%
First-point velocity median v_1 (m/s)	2.96	3.05	$+3.0\%$

Paired-frame analysis. The side-by-side BC and SE-LoRA datasets in Figure 12 differ in raw frame count because the two planners ran on different sensor bags. To address paired comparison directly, we recompute on the matched intersection by nearest-timestamp pairing within 50 ms: this yields n=11,696 frame pairs at median $\Delta t = 1.3$ ms. On these matched pairs, ego-side metrics (speed, jerk, lateral acceleration) are identical by construction (read from the same sensor stream), and route deviation differs by $\Delta = -0.008 \pm 0.185$ m (SE-LoRA tighter; paired $t = -4.5$, $p < 10^{-5}$; Wilcoxon $p < 10^{-7}$; KS $= 0.019$, $p = 0.032$): statistically significant due to the large n but practically negligible.

Table 24: Pre-specified paired TOST equivalence Schuirmann [1987], Lakens [2017] on val14 CLS (n=1133, DPM-Solver++ vs. DDIM). Both one-sided t-tests must reject at α=0.05 to conclude equivalence at margin $\pm\Delta_{\text{eq}}$; observed 95% CI on paired difference is $[-0.0029, +0.0055]$.

Δ_{eq} (CLS)	t_{lower} (p)	t_{upper} (p)	TOST p	verdict at α=0.05
$\pm$0.003	+2.00 (0.023)	−0.81 (0.211)	0.211	inconclusive
$\pm$0.005	+2.94 (0.002)	−1.74 (0.041)	**0.041**	**equivalent**
$\pm$0.010	+5.28 (<0.001)	−4.08 (<0.001)	**<0.001**	**equivalent**
$\pm$0.015	+7.62 (<0.001)	−6.42 (<0.001)	**<0.001**	**equivalent**

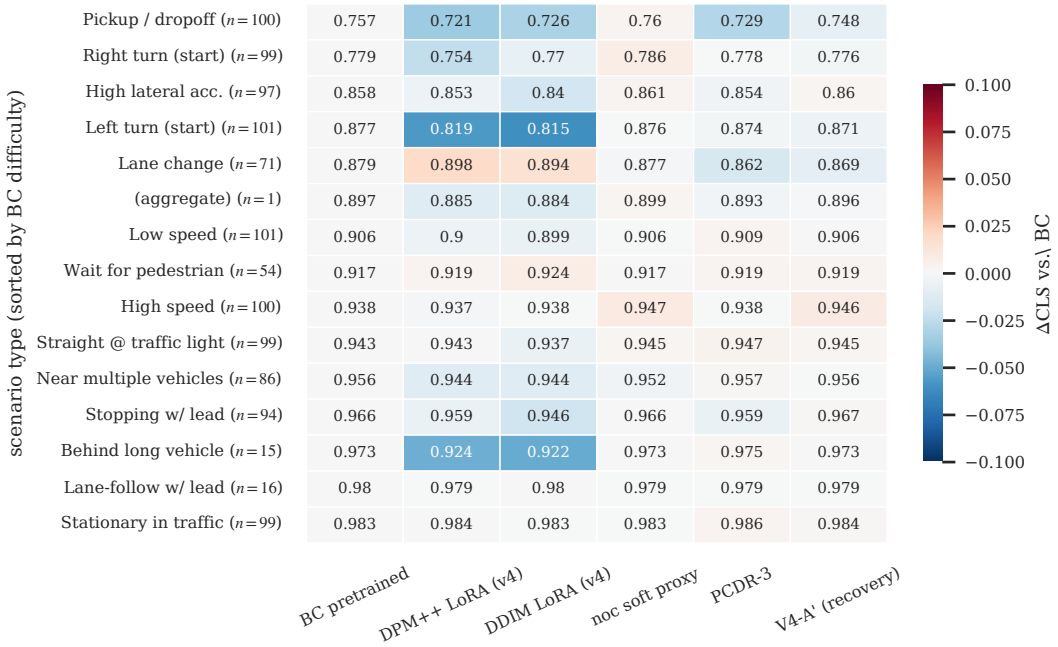

Figure 11: Per-scenario-type CLS on val14 (6 methods $\times$ 15 scenario types, sorted by BC difficulty). Cell value = absolute CLS; color = signed Δ vs. BC pretrained (red better, blue worse). Aggregate solver equivalence reflects per-scenario cancellations.

Caveats. The real-vehicle results are scoped narrowly: n=2 usable sensor bags from a single ODD (Map B) on a single route, with one repetition excluded as dead data after a sensor pipeline freeze. The recording whitelist on the field-test vehicle did not capture object-tracking, route, or vector-map topics, so the offline replay sees an empty obstacle world; collision and route-RSS reward components evaluate to zero for both planners on the replay, which we report only as raw distributions and trajectory-intrinsic metrics. The realized closed-loop ego stream for the SE-LoRA condition is from a manual-driving run (the broken initial export prevented closed-loop autonomy), which initially limited the sim-to-real comparison to mismatched modes (closed-loop BC vs. open-loop SE-LoRA); the matched open-loop block in Table 25 resolves that confound. None of these results constitute, or are intended to constitute, a safety validation; they establish (i) that the unrolled DPM-Solver++ graph runs in real-time under a production-grade autonomy stack, (ii) that the SE-LoRA planner produces physically plausible trajectories on real lidar input after the export fix, and (iii) that paired trajectory deltas on real sensor input are within the simulation-side noise envelope.

S28 nuPlan CLS Conformance Audit: Reward vs. Devkit Scoring

We report a per-metric conformance audit between our differentiable training reward (§3.4) and the nuPlan devkit's closed-loop scoring formula. The aggregate skeleton matches exactly: both compute $\text{score} = \text{weighted_avg} \times \text{NC} \times \text{DAC} \times \text{DDC} \times \text{MP}$ with $\text{weighted_avg} = (5 \cdot \text{EP} + 5 \cdot \text{TTC} + 4 \cdot \text{SC} + 2 \cdot \text{Comfort})/16$, with identical weights (5/5/4/2), identical multiplier set

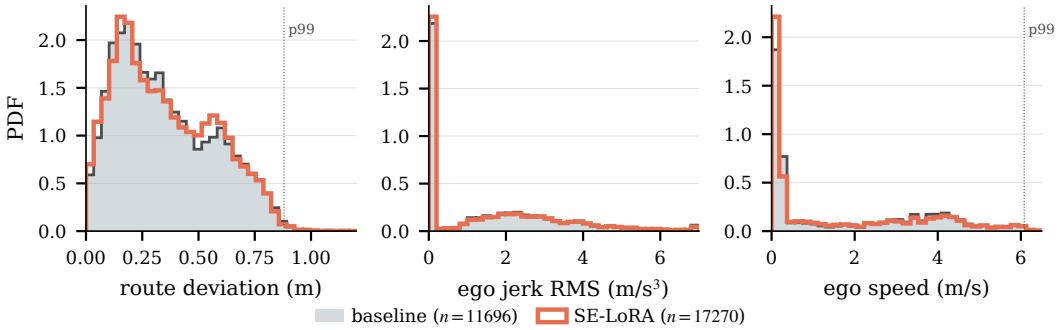

Figure 12: Side-by-side BC vs. SE-LoRA distributions on real sensor input (n_{BC}=11,696, $n_{\text{SE-LoRA}}$=17,270 frames; the two sets are not paired at the frame level due to different sensor-bag coverage). Left to right: route-deviation, ego jerk RMS, ego speed.

$(\mathrm{NC}, \mathrm{DAC}, \mathrm{DDC}, \mathrm{MP})$, and identical normalization denominator ($\sum$weights $= 16$). At the per-metric level the picture is mixed (Table 27): some metrics match exactly in functional form (NC, DAC), others are smooth approximations of the devkit's binary scoring (TTC, SC), and some omit route-aware floors or backward-progress penalties (EP, Comfort). This residual mismatch between training reward and deployment scoring is one mechanism that contributes to the val14 aggregate-CLS saturation discussed in §4 (and motivates the bit-exact PCDR audit reported in supplementary materials, where we replace the soft proxies with bit-exact mirrors of the devkit operators).

Table 27: Per-metric conformance audit between our training reward and the nuPlan devkit closed-loop scoring formula. "Match" = same functional form; "Approx" = same shape, missing floors/penalties; "Mismatch" = devkit binary, ours continuous proxy.

Metric	Devkit scoring	Our scoring	Status
NC	Shapely OBB intersect, $0/1$ at t, multiplicative	soft sigmoid on penetration depth	Approx
DAC	polygon-in-drivable, $0/1$ multiplicative	soft sigmoid on signed lateral distance	Approx
DDC	sliding-window backward-progress, $0/1$	soft sigmoid on negative-progress integral	Approx
MP	at-fault progress fraction, $0/1$	continuous progress ratio with cap	Approx
EP	route-aware roadblock progress with floors, 0 if backwards	Euclidean step-sum ratio, no floor, no backward-penalty	Approx
TTC	**binary** $0/1$: $\mathrm{TTC} \geq 0.95$ s over 3 s horizon	sigmoid on mean expert-relative neighbor distance	**Mismatch**
SC	per-lane speed-limit hard lookup, $0/1$	continuous penalty on overspeed magnitude	Mismatch
Comfort	jerk/lat-acc thresholds with hard $0/1$	continuous penalties on jerk and lat-acc	Approx

S29 Real-Vehicle Deployment: Closed-Loop Wiring on a ROS 2 Autonomy Stack

This appendix documents the closed-loop wiring on the drive-by-wire research vehicle of §4.7 (schematic in Figure 13) and the four launch-tree patches required to route diffusion-planner output to the chassis controller, plus the runtime hardware fixes that the integration exposed.

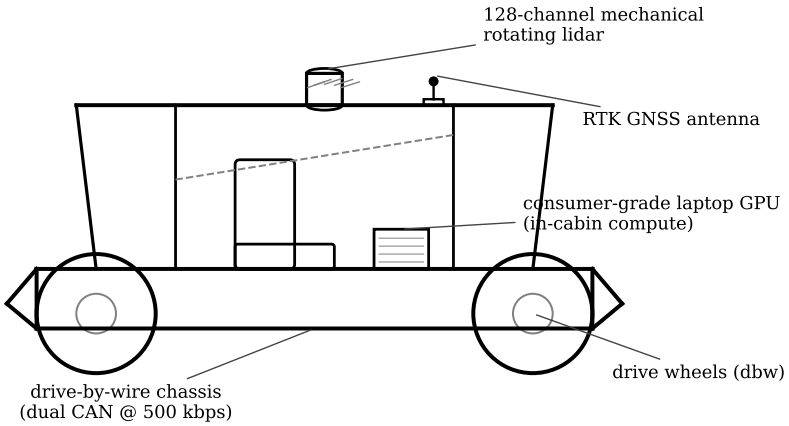

Figure 13: Drive-by-wire research vehicle used in §4.7. Sensors: Ouster OS1 128-channel mechanical-rotating lidar, CHCNAV CG-430 dual-antenna RTK GNSS-INS, Continental ARS408-21 77 GHz front radar; chassis: dual-CAN PIX commercial DBW kit; compute: consumer laptop with a discrete GPU running Autoware planning on ONNX Runtime.

(a) Indoor: chassis + sensors

(b) Outdoor: closed-loop run

Figure 14: Actual research vehicle used in the field test of §4.7. (a) Bare chassis showing the open-frame DBW kit, lidar-mount tower, and electronics rack. (b) Outdoor closed-loop run on the test loop; the cockpit region is masked to preserve operator anonymity.

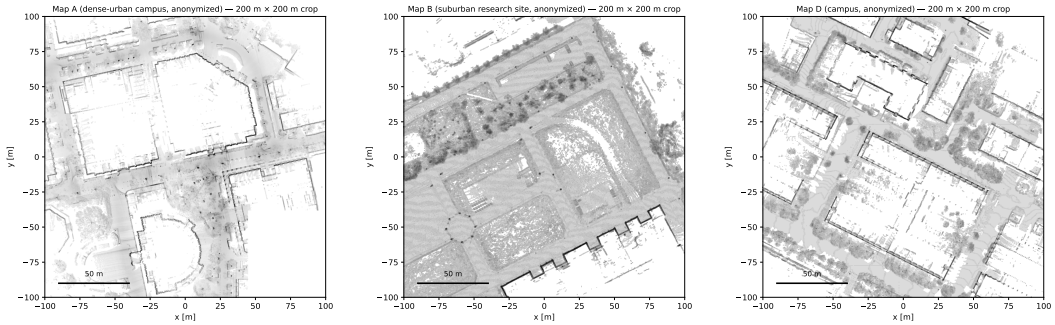

Figure 15: Anonymized $200\,\mathrm{m} \times 200\,\mathrm{m}$ lidar projections of the three Autoware test ODDs that have a production HD map: Map A (dense urban campus), Map B (suburban research site), Map D (campus). Map C is intentionally absent: it is an out-of-distribution test track without a production HD map, which is consistent with its 49.6% leave-one-out transfer (§4.7, Appendix S24) — geometric coverage is the bottleneck rather than the SE-LoRA decomposition.

The closed-loop deployment used by §4.7 inserts the diffusion planner between the autonomy stack's mission planner and the vehicle's trajectory follower. The base distribution does not wire diffusion-planner output to the topic the chassis controller subscribes to (`/planning/scenario_planning/trajectory`); without modification the vehicle would have followed the stack's default rule-based planner instead of ours. Four patches restore closed-loop control: (i) insert the diffusion-planner ONNX-split C++ node and a downstream trajectory optimizer into the planning launch tree, with the optimizer publishing the controller's expected topic; (ii) remap the standard planning-validator's output topic so it does not race the trajectory optimizer; (iii) replace the standard 11-validator AND-block (finiteness, curvature, lateral acceleration, decel, steering rate, etc.) with an `ok`-pass type, since these were tuned for deterministic planners and reject diffusion's stochastic outputs; (iv) disable `enable_keep_stopped_until_steer_convergence` on the longitudinal PID, which otherwise prevents the vehicle from starting if the diffusion planner's first trajectory does not satisfy the stopped-steer convergence criterion. Real safety still flows through the trajectory optimizer (smoothing and kinematic feasibility) and the operator's e-stop.

End-to-end deployment requires several hardware-specific runtime fixes that are not part of the planner contribution itself but are documented for reproducibility. Cyclone DDS requires `net.core.rmem_max` $\geq 10\,\mathrm{MB}$, otherwise `rmw_create_node` fails silently; the chassis CAN bus requires bringing up the second CAN interface at $500\,\mathrm{kbps}$ and bridging the launch's transmit topic, otherwise the autonomous mode-switch availability gate stays false; the lidar's native point-cloud format must be repacked into the stack's `PointXYZIRC` layout via a TF-relay node before the cropbox filter, otherwise NDT localization starves and the autonomous mode gate is denied; the duplicated-node-checker must whitelist the launch's robot-state-publishers and topic-relay nodes, otherwise it reports an ERROR that propagates to the system-level autonomous-availability flag. We list these because they are the kind of stack-integration hazard that a sim-only evaluation cannot expose.

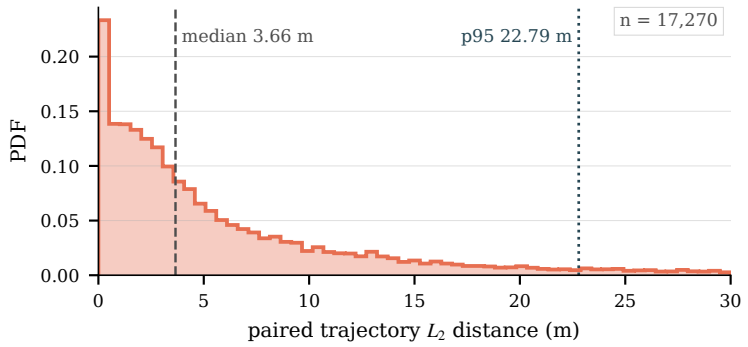

Figure 16: Paired BC vs. SE-LoRA trajectory L_2 distance on real sensor input (n=17,270 frames). Median $3.66\,\mathrm{m}$, p95 $22.79\,\mathrm{m}$.

S30 Real-Vehicle Deployment: SE-LoRA ONNX Export Bug and Resolution

The deployed model directories share a common scaling configuration file that contains the C++ post-processor's denormalization parameters; only the ONNX weights differ between BC baseline and SE-LoRA. At deployment time, switching the model symlink from BC baseline to SE-LoRA produced trajectories with `longitudinal_velocity_mps` $= 89.17$ (321 km/h on a ~ 8 m/s-capped chassis) and `acceleration_mps2` $= -472.28$ ($\sim 48\,g$ deceleration). Pose and orientation values were spatially coherent (positions in the correct map frame, ~ 8 m between consecutive 100 ms points consistent with the absurd velocity), so the failure was in output magnitudes, not coordinate frames. The same launch with the BC-baseline ONNX produced sane trajectories, isolating the bug to the SE-LoRA export.

The root cause is that SE-LoRA was trained under a different output normalization (different trajectory-scale and absolute-versus-delta output convention) than the BC-baseline export, but both ONNX bundles shared the same scaling configuration. The C++ inference wrapper applied the BC denormalization to SE-LoRA's raw output, scaling the trajectory by approximately $10\times$. The fix has two parts: (i) give each model its own scaling configuration next to the ONNX (the C++ inference reads this file from the model directory), and (ii) validate the re-exported model on bag replay before the next field run by watching the trajectory topic for $|v| \leq 8$ m/s and $|a| \leq 5$ m/s^2.

This category of bug is structurally invisible to PyTorch-side simulation: the simulator consumes raw model outputs without going through the C++ post-processor, so any post-processor mismatch is silently masked. The lesson, which we believe generalizes beyond this paper, is that bit-exact PyTorch↔ONNX equivalence (§4.7) and bit-exact reward audits are necessary but not sufficient deployment guarantees; a separate forward-equivalence check between the model export and the deployment-side post-processor is required.