A-FLOW: PERTURBATION CAN BE CORRECTION FOR FROZEN FLOW PLANNERS IN END-TO-END DRIVING

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent end-to-end driving methods use generative models for trajectory planning. They apply reinforcement learning (RL) fine-tuning to improve driving safety and comfort. However, such fine-tuning can be costly and sensitive to training data. Parameter updates may also disrupt pretrained driving behavior. Inspired by non-reversible Langevin dynamics, we design a lightweight skew-symmetric matrix, denoted by A, to parameterize a structured perturbation that is learned as a trajectory correction. Building on this design, we propose A-Flow, a driving framework with three components. First, Temporal BEV Mapping (TBM) combines camera history to represent road geometry and traffic dynamics for trajectory generation. Second, we use GRPO to search for high-reward trajectories around the uncorrected trajectory. We optimize only A while keeping the pretrained planner frozen. The learned matrices form a correction library for reuse. Third, Field-Guided Correction Transfer (FCT) retrieves relevant matrices to generate corrected candidates in unseen scenes. It uses TBM features to evaluate these candidates and select the final trajectory. A-Flow achieves 90.7 PDMS in NAVSIM v1 and an HD-Score of 37.49 in HUGSIM. HUGSIM evaluation uses zero-shot transfer without target-scene reward optimization. Experiments on a ReCogDrive-based planner further demonstrate the adaptability of our correction method.

1 INTRODUCTION

End-to-end autonomous driving integrates perception, prediction, and planning into a single pipeline for joint optimization. It learns to map sensor observations to planned trajectories. However, predicting a single trajectory may not fully capture the diversity of driving behaviors. Generative planners, including DiffusionDrive, GoalFlow, and MeanFuser, learn trajectory distributions through imitation learning (IL) to capture this diversity (Liao et al., 2025; Xing et al., 2025; Wang et al., 2026b). IL trains these planners to imitate expert behavior, but does not directly optimize the driving performance of their generated trajectories. Gen-Drive and ReCogDrive therefore introduce reinforcement learning (RL) fine-tuning. They use driving rewards to improve safety and comfort beyond imitation learning (Huang et al., 2025; Li et al., 2026b).

RL fine-tuning updates pretrained planner parameters and can be costly and sensitive to training data, including the types of driving scenes and their relative proportions (Yuan et al., 2025; Li et al., 2026a; Liu et al., 2025a). PlannerRFT reports that fine-tuning only on collision and road-departure cases degrades performance (Li et al., 2026a). R2SE also finds that relaxing the threshold for selecting difficult scenarios can reduce planning performance. Its ablations further show that refinement without an IL objective can impair previously learned behavior (Liu et al., 2025a). These observations motivate learning trajectory corrections while keeping the pretrained planner frozen. Policy Decorator explores this direction in robotics through a separate residual policy (Yuan et al., 2025). We explore whether driving rewards can similarly train reusable trajectory corrections across scenes.

Non-reversible Langevin dynamics uses skew-symmetric drift perturbations to improve sampling. These perturbations can preserve the Langevin target distribution while altering probability transport (Duncan et al., 2016; Futami et al., 2020). Our key observation is that a small structured perturbation can be optimized into a useful trajectory correction. Hence, we design a lightweight skew-symmetric matrix, denoted by A, to parameterize trajectory corrections. We learn A from driving rewards to explore high-reward trajectories while keeping the pretrained planner frozen. The

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

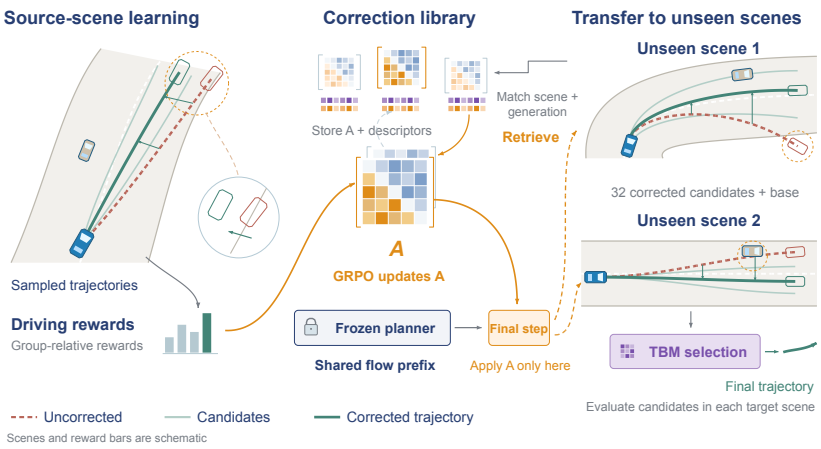

Figure 1: **Learning and reusing adaptive trajectory corrections.** We learn final-step correction matrices from driving rewards for a frozen planner. The same matrix can induce different corrections across unseen scenes through target-dependent flow representations.

learned corrections can repair failed trajectories. They can be transferred to unseen driving scenes without target-scene reward optimization. Building on this design, we propose **A-Flow** (Figure 1), a driving framework that learns and reuses trajectory corrections.

Specifically, A-Flow is composed of three components. First, Temporal Bird's-Eye-View Mapping (TBM) combines current and past camera observations. It provides scene geometry and motion cues for trajectory generation and candidate evaluation. Second, we use GRPO to learn correction matrices from source-scene driving rewards. We store the learned matrices with scene and flow-trajectory descriptors in a correction library. Third, Field-Guided Correction Transfer (FCT) matches corrections using scene and flow-trajectory descriptors. Within each target scene, all candidates share one flow prefix. We apply each retrieved matrix A only at the final flow sampling step. FCT reuses TBM features to evaluate the candidates and select the final trajectory. Applying different matrices requires no additional flow-network evaluations.

We evaluate A-Flow on NAVSIM v1 and HUGSIM. A-Flow achieves 90.7 PDMS on NAVSIM v1 and an HD-Score of 37.49 on HUGSIM. HUGSIM tests zero-shot transfer of the NAVSIM-trained system without target-scene reward optimization. We also retain the ReCogDrive Stage 2 backbone and replace diffusion with flow matching. A-Flow improves this frozen flow planner from 86.12 to 90.3 PDMS using a separately learned correction library.

Our main contributions are:

- We propose A-Flow, a driving framework with reusable trajectory corrections. Inspired by non-reversible Langevin dynamics, we parameterize structured perturbations with lightweight skew-symmetric matrices. We learn these perturbations as trajectory corrections from driving rewards while keeping the pretrained planner frozen.
- We propose Field-Guided Correction Transfer (FCT) to reuse corrections in unseen scenes. FCT uses Temporal Bird's-Eye-View Mapping (TBM) features to evaluate corrected candidates and select the final trajectory.
- Experiments assess planning on NAVSIM v1 and zero-shot transfer to HUGSIM, achieving 90.7 PDMS and 37.49 HD-Score, respectively. Further experiments on ReCogDrive evaluate adaptation of the correction method to another frozen planner.

2 Related Work

Generative Planning and Reinforcement Learning. DiffusionDrive, GoalFlow, and MeanFuser improve the diversity and efficiency of trajectory generation (Liao et al., 2025; Xing et al., 2025; Wang et al., 2026b). Their imitation objectives do not directly optimize driving outcomes. GenDrive and ReCogDrive use RL fine-tuning to address this gap (Huang et al., 2025; Li et al., 2026b).

Flow-GRPO and ReinFlow extend online RL to flow models (Liu et al., 2025b; Zhang et al., 2025b). Updating large policies can be expensive (Yuan et al., 2025). PlannerRFT reports degradation when fine-tuning only on failure cases; including low-score scenes improves performance (Li et al., 2026a). This motivates learning corrections for a frozen planner.

Policy Correction and Experience Reuse. Policy Decorator improves a frozen policy through residual learning (Yuan et al., 2025), while MPA combines a diffusion adapter with value-based trajectory selection (Lin et al., 2025). In both methods, correction experience is encoded in a trained network; matching individual source-scene corrections is outside their design. L2M and IsCiL study retaining and reusing learned skills (Schmied et al., 2023; Lee et al., 2024). For driving, scene-specific reuse requires matching corrections to the target scene and evaluating their effects under road and traffic constraints. A-Flow addresses these two operations through FCT.

Non-Reversible Langevin Dynamics. Non-reversible Langevin methods introduce structured perturbations to improve sampling efficiency. Duncan et al. (2016) analyze variance reduction through non-reversible perturbations, while Futami et al. (2020) use skew-symmetric interactions to accelerate ensemble sampling. These methods sample from a specified target distribution. We use their skew-symmetric structure to learn trajectory corrections from driving rewards.

3 PRELIMINARIES

Trajectory Planning in Autonomous Driving. Let c denote camera history, navigation, and ego state. The planner predicts $\tau = \{(x_h, y_h, \psi_h)\}_{h=1}^{H}$, where each waypoint contains a position and heading. We use $H = 8$ and denote the normalized trajectory by $x \in \mathbb{R}^{3H}$. We learn sampling corrections while keeping the pretrained planner frozen.

Conditional Flow Matching. A flow planner transforms Gaussian noise x_0 into a trajectory x_1, conditioned on c. During training, rectified flow uses the interpolation $x_t = (1-t)x_0 + tx_1$ to learn a velocity field v_θ (Lipman et al., 2023; Liu et al., 2023) through

$$\mathcal{L}_{\mathrm{RF}} = \mathbb{E}_{x_0, x_1, t, c}\left[\left\|v_\theta(x_t, t, c) - (x_1 - x_0)\right\|_2^2\right]. \tag{1}$$

At inference, the planner follows an ordinary differential equation (ODE), approximated with K discrete updates:

$$\frac{dx_t}{dt} = v_\theta(x_t, t, c), \qquad x_{t_{k+1}} = x_{t_k} + \Delta t_k v_\theta(x_{t_k}, t_k, c). \tag{2}$$

Here k indexes the flow steps and $\Delta t_k = t_{k+1} - t_k$. The flow parameter t runs from 0 (Gaussian noise) to 1 (data). It is distinct from physical trajectory time.

ODE–SDE Correspondence. A continuous flow also admits a stochastic formulation. Let $p_t(x \mid c)$ denote the density transported by the flow and $s_t(x, c) = \nabla_x \log p_t(x \mid c)$ its score. With a time-dependent diffusion scale σ_t, the associated stochastic differential equation (SDE) is

$$\mathrm{d}x_t = \left[v_\theta(x_t, t, c) + \tfrac{1}{2}\sigma_t^2 s_t(x_t, c)\right]\mathrm{d}t + \sigma_t\,\mathrm{d}W_t. \tag{3}$$

Here W_t is standard Brownian motion. The score-dependent drift compensates for the added diffusion. With the exact score, matching initial distributions, and standard regularity conditions, this SDE and the flow ODE have the same time-marginal distributions (Song et al., 2021; Liu et al., 2025b). Section 4.2 uses this reference to motivate a correction at the final sampling step.

4 METHODOLOGY

A-Flow learns lightweight corrections from driving rewards while keeping the planner frozen (Figure 2). TBM provides temporal BEV features for trajectory generation and evaluation. GRPO optimizes correction matrices using source-scene rewards. FCT retrieves these matrices, applies them after a shared flow prefix, and selects the final trajectory using TBM features.

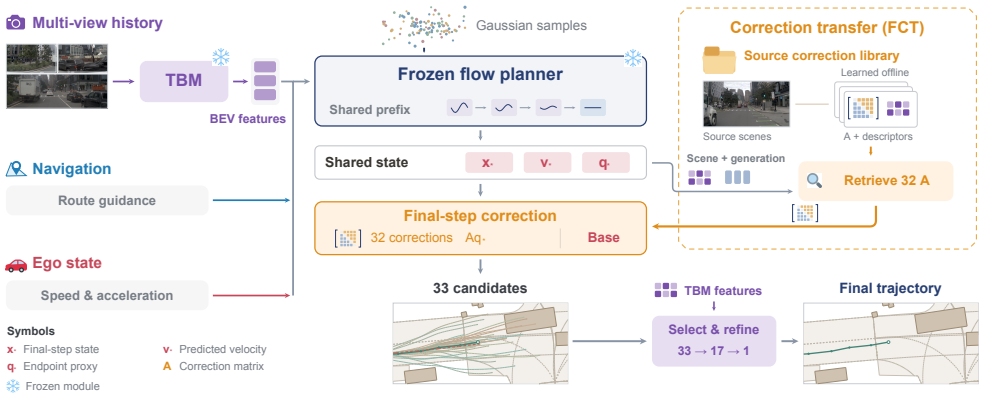

Figure 2: **Overview of the A-Flow framework.** TBM encodes multi-frame, multi-view images into BEV features for the frozen flow planner. FCT matches learned correction matrices to the target scene using scene and flow-trajectory descriptors. Candidate generation shares one flow prefix and applies matrix corrections only at the final sampling step. The resulting 32 corrected trajectories are evaluated alongside Base. TBM features guide selection through value and safety evaluation.

4.1 Temporal BEV Mapping

Temporal Bird's-Eye-View Mapping (TBM) aligns and fuses bird's-eye-view (BEV) features from multi-frame, multi-view images, providing a shared scene representation for both trajectory generation and later FCT evaluation. These features provide road-geometry and traffic-dynamics context. For trajectory generation, pooling them with observation-coverage cues gives F_{plan}, which is queried by the planner at evolving waypoints $p_{t,h}$:

$$f_{t,h} = \text{Bil}(F_{\text{plan}}, p_{t,h}), \qquad \widetilde{e}_{t,h} = e_{t,h} + t^2 W_{\text{loc}} f_{t,h}. \tag{4}$$

Here Bil denotes bilinear interpolation, $f_{t,h}$ is the local TBM feature sampled at waypoint $p_{t,h}$, $e_{t,h}$ is the original waypoint embedding, W_{loc} is a learned projection, and t^2 progressively strengthens local field conditioning toward later flow steps. The resulting $\widetilde{e}_{t,h}$ is used by the planner.

Crucially, the same TBM representation is retained as the scene field for FCT, which queries features along each candidate vehicle footprint to evaluate transferred corrections (Section 4.3). Thus, trajectory generation and correction evaluation are grounded in the same frozen scene representation rather than separately learned target-scene features. Appendix A details TBM construction. RF-DiT is pretrained using Eq. (1) with cached TBM features. After pretraining, both TBM and the planner remain frozen during correction learning and transfer.

4.2 Perturbative Correction with a Non-Reversible Prior

Rather than fine-tuning the pretrained planner, A-Flow learns a lightweight terminal perturbation while keeping the planner frozen. Such perturbative corrections could in principle admit different matrix parameterizations. We adopt the skew-symmetric construction from non-reversible Langevin dynamics (Duncan et al., 2016; Futami et al., 2020) as a structured prior because it has a precise conservative interpretation in the corresponding exact-score reference dynamics.

Let $p_t(\cdot \mid c)$ denote the marginal density transported by the frozen base flow ODE and let $s_t(x, c) = \nabla_x \log p_t(x \mid c)$ be its exact score. We consider the reference process

$$dx_t = \Big[v_\theta(x_t, t, c) + \frac{1}{2}\sigma_t^2 s_t(x_t, c) + \lambda_t A_i s_t(x_t, c)\Big]dt + \sigma_t dW_t, \tag{5}$$

where σ_t is independent of x and λ_t controls the skew perturbation. We parameterize

$$A_i = B_i - B_i^\top, \tag{6}$$

with unconstrained B_i, which guarantees $A_i^\top = -A_i$.

With this parameterization, the skew drift changes probability transport without altering the reference marginals. For state-independent A_i, skew symmetry gives $\nabla \cdot (p_t A_i s_t) = 0$. Together with

the score drift that compensates for diffusion, this preserves the flow ODE's time marginals (Appendix B.1). Moreover, $s_t^\top A_i s_t = 0$, so the perturbation is locally tangent to a level set of p_t in this exact-score reference. Appendix B.2 explains the associated geometry and non-reversible dynamics.

These properties motivate skew symmetry as a conservative structural prior for perturbative correction. We do not claim that skew-symmetric matrices are the only parameterization capable of improving driving reward, nor that the implemented A-Flow transition preserves the pretrained marginals exactly. The perturbation structure constrains how the trajectory may change. Reward optimization determines whether that change becomes a useful correction.

A-Flow applies the learned perturbation only at the final solver step $t_\star = 0.8$. From the frozen flow state $x_\star$, we construct

$$q_\star = \frac{x_\star + (1 - t_\star) v_\theta(x_\star, t_\star, c_i)}{t_\star}. \tag{7}$$

We define $q_\star$ as the scaled endpoint proxy used by the terminal correction. The implemented correction is proportional to $A_i q_\star$, rather than $A_i s_t$. Appendix B.3 treats this terminal perturbation separately from the exact-score reference theory and establishes properties that hold for the implemented update itself.

Group-Relative Correction Learning. Correction learning turns the structured perturbation into a reward-directed trajectory correction with the planner frozen. For each source scene, we sample G terminal trajectories and compare their driving rewards using the group-relative advantage

$$\hat{a}_i^{(g)} = \frac{R_i^{(g)} - \bar{R}_i}{s_i^R + \epsilon}. \tag{8}$$

Inspired by the group-relative reward normalization used in GRPO, these advantages weight a terminal likelihood surrogate that updates only B_i, while TBM and the pretrained planner remain frozen. We use $G = 50$ trajectories per update to obtain a more stable Monte Carlo estimate of relative reward under sampling variability. The matrix with the highest observed group mean reward is retained and stored with its scene and trajectory descriptors for later transfer. Appendix C.2 gives the complete optimization objective.

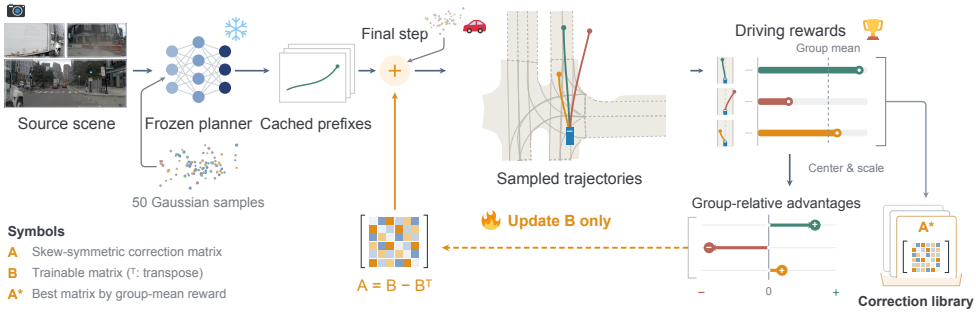

Figure 3: **From structured perturbations to learned corrections.** A skew-symmetric matrix $A = B - B^\top$ perturbs only the final step of cached frozen-planner prefixes. Group-relative driving rewards update only B, turning structured perturbations into reward-directed trajectory corrections. The best matrix $A^\star$ is stored for reuse.

4.3 Field-Guided Correction Transfer

The effect of a learned correction depends on the road geometry and surrounding traffic of the target scene. We therefore introduce Field-Guided Correction Transfer (FCT), which reuses the frozen TBM fields from Section 4.1 to evaluate transferred corrections under the target scene. Specifically, the fields are queried along candidate trajectories and used for scene-aware value and safety learning. The resulting frozen evaluators guide candidate selection and zero-shot scale adjustment, enabling corrections to be transferred to unseen scenes without target-scene reward optimization. Appendix D provides the complete implementation details of FCT.

Data: Correction Bank Construction. We start from the 85,109 source corrections learned by group-relative correction learning. For each training query q, scene and flow-trajectory similarities independently rank the corrections, and their rankings are fused by reciprocal rank fusion (RRF):

$$S_{\mathrm{RRF}}(A, q) = \sum_{m \in \{\mathrm{scene}, \mathrm{traj}\}} \frac{1}{60 + r_m(A, q) + 1}, \tag{9}$$

where A denotes a correction matrix and $r_m(A, q)$ is its zero-based rank under descriptor m. We then reverse these associations and evaluate each A across its matched training scenes using

$$Q(A) = \frac{\sum_{p \in P_A} \bar{d}_{A,p}}{|P_A| + 5}, \qquad d_{A,q} = \mathrm{PDMS}(A, q) - \mathrm{PDMS}(\mathrm{Base}, q) - b_{\mathrm{cohort}(q)}. \tag{10}$$

Here, $d_{A,q}$ is the bias-corrected PDMS gain of A on query q, $\bar{d}_{A,p}$ averages these gains within parent recording p, P_A is the set of parent recordings matched to A, and $b_{\mathrm{cohort}(q)}$ is the mean offset of the data cohort containing q. The additive constant 5 shrinks scores supported by few parent recordings toward zero. We rank matrices by $Q(A)$ and retain the top 15% (12,767) as a high-transferability correction bank for selector training and inference.

Training: Value and Safety Learning. We train value and safety evaluators from offline trajectory outcomes. The value evaluators estimate the driving quality of corrected trajectories, while the safety evaluators estimate correction-induced failure risk. Both use trajectory, ego, and TBM features so that transferred corrections can be evaluated under the current scene context. All evaluators are trained offline and remain frozen during inference. They are used to rank and select among retrieved candidate trajectories without target-scene optimization.

Inference: Scale Adjustment and Candidate Enhancement. At inference, FCT performs zero-shot scale adjustment by varying the magnitude of a selected correction while keeping its matrix and underlying flow realization fixed. It further expands retrieval coverage through an additional flow-trajectory retrieval branch, which matches the test-time Base flow trajectory against cached training flow trajectories and retrieves the nearest associated corrections. The frozen FCT evaluators re-evaluate the resulting candidates using predicted driving quality and safety, and select the final trajectory without target-scene reward optimization.

5 Experiments

5.1 Experimental Setup

Dataset and Metrics. We evaluate A-Flow on NAVSIM v1 and HUGSIM. NAVSIM (Dauner et al., 2024) is a planning benchmark built from nuPlan driving data. We train on `navtrain` and evaluate on 12,146 `navtest` scenarios using non-reactive simulation. We report PDM Score (PDMS) and its component metrics. HUGSIM (Zhou et al., 2024) evaluates closed-loop driving in reconstructed environments from KITTI-360, nuScenes, PandaSet, and Waymo. We evaluate our NAVSIM-trained system on all 436 scenarios without additional training or target-scene reward optimization. We report HD-Score, PDMS, route completion (RC), and the corresponding safety and comfort components. We use the same HUGSIM aggregation protocol as the published baselines.

Implementation Details. TBM encodes four frames from three cameras using ResNet-34-FPN. RF-DiT generates eight waypoints over four seconds using a five-step flow solver. We first train TBM, then train RF-DiT using its frozen representations. The pretrained RF-DiT planner conditioned on TBM features serves as our baseline. A-Flow builds on this frozen baseline through correction matrix learning and FCT.

With both modules frozen, we learn scene-specific correction matrices using group-relative correction learning. Correction learning covers 85,109 NAVSIM scenes. Each scene receives 250 matrix updates, with 50 sampled trajectories per update. Matrix optimization and merging take 14.33 hours of wall-clock time on 8 NVIDIA A100 GPUs. Peak GPU memory usage during optimization is approximately 3.95 GiB per device. The evaluators are trained offline.

At inference, each selection pass evaluates 32 corrected candidates together with the base trajectory. The corrected candidates share one flow prefix, with matrix corrections applied only at the

final sampling step. All models and correction matrices remain fixed during evaluation. Further implementation details and computational costs are provided in the appendix.

5.2 Benchmark Results and Planner Adaptation

Planning on NAVSIM. A-Flow achieves 90.7 PDMS in Table 1. Compared with MeanFuser, A-Flow improves EP by 3.4 points and also achieves higher DAC and TTC, while matching its NAC of 98.6. This is consistent with joint value and risk assessment: reward-guided corrections provide alternatives that improve progress, while safety evaluation uses target-scene road and traffic features to limit corrections with high predicted risk. The results suggest that A-Flow can exploit useful corrections while maintaining strong collision compliance.

Table 1: **Planning performance on NAVSIM v1.** (a) ResNet-34 image encoders. (b) Flow-based planners. C denotes camera, and L denotes LiDAR.

(a) ResNet-34 image encoders

Method	Planner	Input	NAC↑	DAC↑	TTC↑	Comf.↑	EP↑	PDMS↑
DiffusionDrive (2025)	Diffusion	C+L	98.2	96.2	94.7	100.0	82.2	88.1
WoTE (2025)	Anchor-based	C+L	98.5	96.8	94.9	99.9	81.9	88.3
SeerDrive (2025a)	Anchor-based	C+L	98.4	97.0	94.9	99.9	83.2	88.9
ResAD (2026b)	Diffusion	C+L	98.0	97.5	94.1	100.0	83.3	88.8
MeanFuser (2026b)	Flow	C	98.6	97.0	95.0	100.0	82.8	89.0
DriveSuprim (2026a)	Anchor-based	C	97.8	97.3	93.6	100.0	86.7	89.9
BeyondDrive (LTFv7) (2026a)	Regression	C	98.4	97.9	95.0	100.0	83.7	89.7
HAD (2026b)	Diffusion	C+L	98.2	97.3	97.5	100.0	87.4	90.2
HAD-L (2026b)	Diffusion	C	98.1	97.2	97.3	100.0	87.2	89.9
A-Flow (Ours)	Flow	C	98.6	97.6	95.5	99.4	86.2	**90.7**

(b) Flow-based planners

Method	Input	Backbone	NAC↑	DAC↑	TTC↑	Comf.↑	EP↑	PDMS↑
GoalFlow (2025)	C+L	V2-99	98.4	98.3	94.6	100.0	85.0	90.3
MeanFuser (2026b)	C	ResNet-34	98.6	97.0	95.0	100.0	82.8	89.0
WAM-Flow (2026)	C	Janus-1.5B	99.2	98.3	97.0	99.7	82.3	90.3
X-VLA (2026a)	C	Florence-Large	97.5	96.5	92.9	100.0	82.2	87.3
MeanFuser + BeyondDrive (2026a)	C	ResNet-34	98.5	98.4	95.1	100.0	84.2	90.3
A-Flow (Ours)	C	ResNet-34-FPN	98.6	97.6	95.5	99.4	86.2	**90.7**

Zero-shot Transfer to HUGSIM. We transfer the NAVSIM-trained system to all 436 HUGSIM scenes without additional training or target-scene reward optimization (Table 2). The Base is the pretrained planner without matrix corrections. Under the same official HUGSIM controller, A-Flow raises its HD-Score from 24.26 to 37.49, with gains in route completion and safety. Source-domain corrections thus remain useful in a new closed-loop environment. HD-Score improves across all four constituent datasets, with the largest gain of 25.67 points on Waymo. These consistent gains across datasets support the effectiveness of A-Flow under cross-domain transfer.

Adaptation to a ReCogDrive-Based Planner. We use a flow-matching planner built on the ReCogDrive Stage 2 backbone and learn a new correction library while keeping the planner frozen. We then keep this library fixed and retrain FCT on its candidates. PDMS increases from 86.12 to 90.3 on 12,146 scenes (Table 3). Official diffusion results provide separate references (Li et al., 2026b). Appendix E details the adaptation procedure.

5.3 Correction and Transfer Ablations

Correction ablation. Table 4 separates correction potential from the performance achieved by selection. Base obtains 85.4 PDMS. Random skew-symmetric matrices match the learned matrices in variance and Frobenius norm. They occasionally produce useful corrections, but FCT frequently retains Base, yielding 86.83 final PDMS. Learned skew-symmetric A achieves 95.13 Oracle@32 and 90.69 final PDMS, compared with 93.18 and 88.81 for the learned unconstrained matrix. Reward

Table 2: **Closed-loop driving on 436 HUGSIM scenes.**

(a) Overall performance

Method	NC	DAC	TTC	Comfort	PDMS	RC	HD-Score
Published baselines							
LTF (Dauner et al., 2024)	44.28	92.75	37.51	94.78	36.53	38.04	23.10
UniAD (Hu et al., 2023)	65.55	93.20	51.56	66.33	49.40	43.83	31.24
VAD (Jiang et al., 2023)	41.17	90.28	27.98	95.34	28.31	30.06	13.93
DrivoR (Kirby et al., 2026)	52.17	95.59	46.20	93.90	44.75	47.21	32.52
Our evaluation							
Base	50.20	96.42	41.24	86.68	40.90	39.36	24.26
A-Flow	**66.98**	**97.03**	**60.37**	**97.98**	**60.55**	46.51	**37.49**

(b) HD-Score by dataset (our evaluation)

Method	nuScenes	KITTI-360	Waymo	PandaSet	Overall
Base	27.50	12.28	23.33	33.93	24.26
A-Flow	**37.67**	**17.68**	**49.00**	**45.62**	**37.49**
Improvement	+10.17	+5.40	+25.67	+11.69	+13.23

Table 3: **Adaptation to ReCogDrive.** (a) Official diffusion results. (b) Our flow-planner evaluation on 12,146 NAVSIM scenes.

(a) Diffusion planner: official results (Li et al., 2026b)

Model	IL PDMS↑	RL PDMS↑	Gain
ReCogDrive-2B	86.5	90.8	+4.3
ReCogDrive-8B	86.5	90.4	+3.9

(b) Flow planner: our evaluation

Setting	PDMS↑	NAC↑	DAC↑	EP↑	TTC↑	Comf.↑	DDC↑
ReCogDrive-2B IL	86.12	98.09	94.44	80.60	94.08	99.97	97.83
+ A-Flow	**90.30**	97.98	**97.49**	**86.88**	93.93	99.89	97.07

learning turns the skew-structured perturbation into a substantially stronger correction. FCT aims to realize this potential through candidate evaluation and selection. The remaining oracle gap indicates room for better selection. Figure 5 in Appendix G illustrates how a learned skew correction improves clearance and passes the TTC check.

Table 4: **Correction ablation on NAVSIM.** Oracle@32 averages, across scenes, the best PDMS among the 32 corrected candidates in each scene. Gain is relative to Base.

Setting	Final PDMS↑	Oracle@32↑	Gain
Base ($A = 0$)	85.4	–	–
Random skew-symmetric A	86.83	92.05	+1.43
Learned unconstrained M	88.81	93.18	+3.41
Learned skew-symmetric A	**90.69**	**95.13**	**+5.29**

Ablation of FCT Components. In Table 5, value evaluation reaches 89.6 PDMS with 33.62 ms of additional latency over Base. Safety evaluation further improves PDMS, NAC, DAC, and TTC, reaching 90.3 PDMS at 125.02 ms. These gains support assessing both correction value and risk. The full configuration reaches 90.7 PDMS at 241.33 ms. Sharing the flow prefix avoids repeating full sampling for each correction. FCT offers a controllable accuracy–latency trade-off: core value and safety evaluation captures most of the gain, while optional scale adjustment and candidate enhancement trade additional computation for higher planning performance.

Table 5: **Cumulative FCT ablation on NAVSIM.** Runtime is measured on a single NVIDIA A100 80GB GPU with a network batch size of 1.

Setting	PDMS↑	NAC↑	DAC↑	EP↑	TTC↑	Comfort↑	DDC↑	Time (ms)↓
Base	85.4	97.9	94.4	80.0	93.0	100.0	97.9	**66.61**
+ Value learning	89.6	98.5	96.7	85.4	94.8	99.5	96.7	100.23
+ Safety learning	90.3	98.6	97.3	85.8	95.3	99.4	96.8	125.02
+ Scale adjustment	90.4	98.6	97.4	86.1	95.3	99.2	96.6	157.46
+ Candidate enhancement	**90.7**	98.6	97.6	86.2	95.5	99.4	96.5	241.33

5.4 Correction and Transfer Analysis

Figure 4 illustrates correction reuse in unseen scenes: (a) repairing a severe failure, (b) improving multiple targets with one matrix, and (c) refining already strong trajectories. Thus, a fixed learned perturbation can induce scene-dependent corrections through the target-dependent flow representation. These examples are qualitative cases selected to illustrate correction effects rather than final selector outputs.



Figure 4: **Reusing the same matrix across scenes.** Red dashed and teal solid lines show base and corrected trajectories.

6 Conclusion

A-Flow learns structured perturbations as reusable trajectory corrections for a frozen flow planner. Lightweight skew-symmetric matrices parameterize these perturbations. FCT uses TBM features to evaluate candidate driving quality and failure risk before selecting the final trajectory. Experiments demonstrate improved planning on NAVSIM, zero-shot transfer to HUGSIM, and adaptation to a ReCogDrive-based flow planner. These results show how a structured perturbation can become a useful correction across scenes without updating the pretrained planner.

AI USE STATEMENT

We used generative AI tools, including ChatGPT and Codex, to assist with writing and language editing, identifying and summarizing related work, and research ideation and execution. Research assistance included experimental design discussions, method implementation, code editing, and interpretation of results. We also used these tools for figure and table preparation and LaTeX formatting. All AI-assisted outputs were reviewed and verified by the authors. The authors take responsibility for the final content, including methods, results, and conclusions.

REPRODUCIBILITY STATEMENT

Our source code is available anonymously at `https://anonymous.4open.science/r/A-Flow-4D0B/`. Appendices A–D provide model details, theoretical derivations, training objectives, and inference procedures.

REFERENCES

Daniel Dauner, Marcel Hallgarten, Tianyu Li, Xinshuo Weng, Zhiyu Huang, Zetong Yang, Hongyang Li, Igor Gilitschenski, Boris Ivanovic, Marco Pavone, Andreas Geiger, and Kashyap Chitta. NAVSIM: Data-driven non-reactive autonomous vehicle simulation and benchmarking. In *Advances in Neural Information Processing Systems*, volume 37, pp. 28706–28719. Curran Associates, Inc., 2024. doi: 10.52202/079017-0902.

Andrew B. Duncan, Tony Lelièvre, and Grigorios A. Pavliotis. Variance reduction using nonreversible Langevin samplers. *Journal of Statistical Physics*, 163(3):457–491, 2016. doi: 10.1007/s10955-016-1491-2.

Futoshi Futami, Issei Sato, and Masashi Sugiyama. Accelerating the diffusion-based ensemble sampling by non-reversible dynamics. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 3337–3347. PMLR, 2020.

Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, Lewei Lu, Xiaosong Jia, Qiang Liu, Jifeng Dai, Yu Qiao, and Hongyang Li. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17853–17862, June 2023.

Zhiyu Huang, Xinshuo Weng, Maximilian Igl, Yuxiao Chen, Yulong Cao, Boris Ivanovic, Marco Pavone, and Chen Lv. Gen-Drive: Enhancing diffusion generative driving policies with reward modeling and reinforcement learning fine-tuning. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3445–3451. IEEE, 2025. doi: 10.1109/ICRA55743.2025.11127286.

Bo Jiang, Shaoyu Chen, Qing Xu, Bencheng Liao, Jiajie Chen, Helong Zhou, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. VAD: Vectorized scene representation for efficient autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8340–8350, October 2023.

Ellington Kirby, Alexandre Boulch, Yihong Xu, Yuan Yin, Gilles Puy, Éloi Zablocki, Andrei Bursuc, Spyros Gidaris, Renaud Marlet, Florent Bartoccioni, Anh-Quan Cao, Nermin Samet, Tuan-Hung Vu, and Matthieu Cord. Driving on registers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 32058–32069, June 2026.

Daehee Lee, Minjong Yoo, Woo Kyung Kim, Wonje Choi, and Honguk Woo. Incremental learning of retrievable skills for efficient continual task adaptation. In *Advances in Neural Information Processing Systems*, volume 37, pp. 17286–17312. Curran Associates, Inc., 2024. doi: 10.52202/079017-0549.

Hongchen Li, Tianyu Li, Jiazhi Yang, Mingyang Shang, Gaoqiang Wu, Caojun Wang, Haochen Tian, Zengrong Lin, Zhihui Hao, XianPeng Lang, Jia Hu, and Hongyang Li. PlannerRFT: Reinforcing diffusion planners through closed-loop and sample-efficient fine-tuning. In *Proceedings*

of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 24929–24938, June 2026a.

Yingyan Li, Yuqi Wang, Yang Liu, Jiawei He, Lue Fan, and Zhaoxiang Zhang. End-to-end driving with online trajectory evaluation via BEV world model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 27137–27146, October 2025.

Yongkang Li, Kaixin Xiong, Xiangyu Guo, Fang Li, Sixu Yan, Gangwei Xu, Lijun Zhou, Long Chen, Haiyang Sun, Bing Wang, Kun Ma, Guang Chen, Hangjun Ye, Wenyu Liu, and Xinggang Wang. ReCogDrive: A reinforced cognitive framework for end-to-end autonomous driving. In *International Conference on Learning Representations*, 2026b. URL https://iclr.cc/virtual/2026/poster/10010169.

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)*, pp. 12037–12047, June 2025.

Haohong Lin, Yunzhi Zhang, Wenhao Ding, Jiajun Wu, and Ding Zhao. Model-based policy adaptation for closed-loop end-to-end autonomous driving. In *Advances in Neural Information Processing Systems*, volume 38, pp. 41532–41562. Curran Associates, Inc., 2025. doi: 10.52202/085713-1386.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

Haochen Liu, Tianyu Li, Haohan Yang, Li Chen, Caojun Wang, Ke Guo, Haochen Tian, Hongchen Li, Hongyang Li, and Chen Lv. Reinforced refinement with self-aware expansion for end-to-end autonomous driving. *arXiv preprint arXiv:2506.09800*, 2025a. URL https://arxiv.org/abs/2506.09800.

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-GRPO: Training flow matching models via online RL. In *Advances in Neural Information Processing Systems*, volume 38, pp. 40783–40818. Curran Associates, Inc., 2025b. doi: 10.52202/085713-1362.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*, 2023.

Thomas Schmied, Markus Hofmarcher, Fabian Paischer, Razvan Pascanu, and Sepp Hochreiter. Learning to modulate pre-trained models in RL. In *Advances in Neural Information Processing Systems*, volume 36, pp. 38231–38265. Curran Associates, Inc., 2023. doi: 10.52202/075280-1660.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2011.13456.

Junli Wang, Zhihua Hua, Xueyi Liu, Zebin Xing, Haochen Tian, Kun Ma, Hangjun Ye, Guang Chen, Long Chen, and Qichao Zhang. Beyond imitation: Learning safe end-to-end autonomous driving from hard negatives. In *European Conference on Computer Vision*, 2026a. URL https://arxiv.org/abs/2605.19771.

Junli Wang, Yinan Zheng, Xueyi Liu, Zebin Xing, Pengfei Li, Kun Ma, Hangjun Ye, Guang Chen, Guang Li, Long Chen, Zhongpu Xia, and Qichao Zhang. MeanFuser: Fast one-step multi-modal trajectory generation and adaptive reconstruction via MeanFlow for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17884–17893, June 2026b.

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 (CVPR)*, pp. 1602–1611, June 2025.

Yifang Xu, Jiahao Cui, Zhihao Zhu, Hanlin Shang, Shan Luan, Mingwang Xu, Feipeng Cai, Neng Zhang, Yaoyi Li, Jia Cai, and Siyu Zhu. WAM-Flow: Parallel coarse-to-fine motion planning via discrete flow matching for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24918–24928, June 2026.

Wenhao Yao, Zhenxin Li, Shiyi Lan, Zi Wang, Xinglong Sun, Jose M. Alvarez, and Zuxuan Wu. DriveSuprim: Towards precise trajectory selection for end-to-end planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 11910–11918, 2026a. doi: 10.1609/aaai.v40i14.38178.

Wenhao Yao, Xinglong Sun, Zhenxin Li, Shiyi Lan, Zi Wang, Jose M. Alvarez, and Zuxuan Wu. HAD: Combining hierarchical diffusion with metric-decoupled RL for end-to-end driving. In *European Conference on Computer Vision*, 2026b. URL https://arxiv.org/abs/2604.03581.

Xiu Yuan, Tongzhou Mu, Stone Tao, Yunhao Fang, Mengke Zhang, and Hao Su. Policy decorator: Model-agnostic online refinement for large policy model. In *International Conference on Learning Representations*, pp. 28129–28164, 2025.

Bozhou Zhang, Nan Song, Jingyu Li, Xiatian Zhu, Jiankang Deng, and Li Zhang. Future-aware end-to-end driving: Bidirectional modeling of trajectory planning and scene evolution. In *Advances in Neural Information Processing Systems*, volume 38, 2025a.

Tonghe Zhang, Chao Yu, Sichang Su, and Yu Wang. ReinFlow: Fine-tuning flow matching policy with online reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 38, pp. 106282–106319. Curran Associates, Inc., 2025b. doi: 10.52202/085713-3547.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, Ya-Qin Zhang, Jiangmiao Pang, Jingjing Liu, Tai Wang, and Xianyuan Zhan. X-VLA: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. In *International Conference on Learning Representations*, 2026a. URL https://arxiv.org/abs/2510.10274.

Zhiyu Zheng, Shaoyu Chen, Haoran Yin, Xinbang Zhang, Jialv Zou, Xinggang Wang, Qian Zhang, and Lefei Zhang. ResAD: Normalized residual trajectory modeling for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3729–3739, June 2026b.

Hongyu Zhou, Longzhong Lin, Jiabao Wang, Yichong Lu, Dongfeng Bai, Bingbing Liu, Yue Wang, Andreas Geiger, and Yiyi Liao. HUGSIM: A real-time, photo-realistic and closed-loop simulator for autonomous driving. *arXiv preprint arXiv:2412.01718*, 2024.

A TEMPORAL BEV MAPPING

TBM takes the current frame and three past frames sampled at 0.5-second intervals. For each frame, a ResNet–FPN extracts image features, which are lifted into BEV using predicted depth and camera calibration. We align each BEV feature map B_r to the current ego frame before temporal fusion:

$$\bar{B}_r = \mathcal{W}(B_r, T_{r\to 0}), \qquad H_r = \mathrm{ConvGRU}(\bar{B}_r, H_{r-1}), \quad r = -3, \dots, 0. \tag{11}$$

Here B_r is the BEV feature of frame r, $T_{r\to 0}$ is the relative ego transform, $\mathcal{W}$ is spatial warping, and H_r is the temporally fused recurrent state. We initialize $H_{-4} = 0$. Alignment compensates for ego motion, allowing recurrent fusion to capture changes in surrounding traffic.

The fused BEV produces latent scene features Z and predicted planning fields F_{TBM}. Separate heads predict drivable area, road signed distance, boundaries, lane direction, a navigation-conditioned route corridor, and future occupancy and motion. The latent features Z retain scene context not explicitly represented by these supervised field outputs. The fields provide geometry and traffic predictions at candidate locations. Spatial pooling packs the predicted TBM fields together with observation-coverage cues into F_{plan}. Both Z and F_{plan} use a coarse BEV grid; their tokens are fused for global conditioning.

The RF-DiT planner uses Eq. (4) to query F_{plan} at each evolving waypoint. The t^2 gate progressively strengthens local field conditioning toward later flow steps as the trajectory becomes more resolved. The map stays fixed during sampling, while updated waypoints determine subsequent queries. We train TBM with geometric and dynamic supervision, then train the flow planner on its cached outputs. Appendix C.1 gives the flow pretraining objective.

B SKEW CORRECTIONS AND NON-REVERSIBLE DYNAMICS

B.1 EXACT-SCORE MARGINAL PRESERVATION

ODE–SDE Correspondence without the Skew Term. Let $p_t(x \mid c)$ be the density transported by the pretrained flow

$$\frac{dx_t}{dt} = v_\theta(x_t, t, c), \tag{12}$$

whose density satisfies the continuity equation

$$\partial_t p_t = -\nabla \cdot (p_t v_\theta). \tag{13}$$

Define the score $s_t = \nabla \log p_t$ and consider

$$dx_t = \left[v_\theta + \frac{1}{2}\sigma_t^2 s_t \right] dt + \sigma_t dW_t, \tag{14}$$

where σ_t is independent of x. Its Fokker–Planck equation is

$$\partial_t p_t = -\nabla \cdot (p_t v_\theta) - \frac{1}{2}\sigma_t^2 \nabla \cdot (p_t s_t) + \frac{1}{2}\sigma_t^2 \Delta p_t. \tag{15}$$

Since $s_t = \nabla \log p_t$, we have $p_t s_t = \nabla p_t$, and therefore

$$-\frac{1}{2}\sigma_t^2 \nabla \cdot (p_t s_t) + \frac{1}{2}\sigma_t^2 \Delta p_t = 0. \tag{16}$$

Hence the SDE reduces to the same continuity equation in Eq. (13), and therefore has the same time marginals as the original flow ODE under the standard regularity conditions.

Adding the Skew-Symmetric Drift. We add a skew-symmetric score drift to Eq. (14):

$$dx_t = \left[v_\theta + \frac{1}{2}\sigma_t^2 s_t + \lambda_t A s_t \right] dt + \sigma_t dW_t, \qquad A^\top = -A, \tag{17}$$

where A and λ_t are independent of x. The additional Fokker–Planck term is $-\lambda_t \nabla \cdot (p_t A s_t)$. Since $p_t s_t = \nabla p_t$,

$$\begin{aligned} \nabla \cdot (p_t A s_t) &= \nabla \cdot (A \nabla p_t) \\ &= \sum_{i,j} A_{ij}\, \partial_i \partial_j p_t = 0, \end{aligned} \tag{18}$$

because A is skew-symmetric whereas the Hessian of p_t is symmetric. Together with Eq. (16), this again gives

$$\partial_t p_t = -\nabla \cdot (p_t v_\theta). \tag{19}$$

Thus, with the exact score, the skew-augmented SDE has the same time-marginal distributions as the base flow ODE.

B.2 Non-Reversible Structure of the Skew Term

The skew drift in Eq. (17) induces the additional probability current

$$J_{A,t} = \lambda_t p_t A s_t = \lambda_t A \nabla p_t. \tag{20}$$

As shown in Eq. (18),

$$\nabla \cdot J_{A,t} = 0. \tag{21}$$

Thus, although $J_{A,t}$ can be nonzero, it makes no contribution to the time evolution of p_t. This divergence-free circulation is the same skew-current structure used in non-reversible Langevin dynamics. In the stationary Langevin setting, a nonzero current of this form violates detailed balance; here we use the construction as a structural prior for the time-dependent flow dynamics.

The skew structure also has a direct geometric interpretation. Since $s_t = \nabla \log p_t$ and $A^\top = -A$,

$$\frac{d}{d\varepsilon} \log p_t(x + \varepsilon A s_t(x))\Big|_{\varepsilon=0} = s_t(x)^\top A s_t(x) = 0. \tag{22}$$

Thus, to first order, the skew drift moves locally along a level set of p_t rather than directly toward higher- or lower-density regions. The skew term therefore changes the direction of probability transport while preserving the density evolution established in Appendix B.1.

B.3 From the Reference Prior to the Implemented Perturbation

Appendix B.1 establishes marginal preservation for the exact-score reference process, while Appendix B.2 characterizes the geometry of the associated divergence-free probability current and its connection to the standard non-reversible Langevin construction. These results motivate the skew-symmetric parameterization used by A-Flow, but they apply to the reference drift $A_i s_t$, not directly to the implemented correction.

Implemented Endpoint Perturbation. A-Flow does not evaluate the exact score during terminal correction. Instead, it uses the scaled endpoint proxy $q_\star$ defined in Eq. (7), which is constructed from the frozen flow state and velocity and is not identified with the exact score s_t.

Define the zero-correction terminal proposal used by the implementation as

$$b_\star = x_\star + \Delta t\, v_\theta(x_\star, t_\star, c_i) + \frac{1}{2}\eta^2 q_\star. \tag{23}$$

The corrected terminal transition is

$$\mu_i = \mathcal{C}(b_\star + \rho A_i q_\star), \qquad X_{A_i} = \mathcal{C}\big(\mu_i + \eta \bar{\xi}_i\big), \tag{24}$$

where $\mathcal{C}(z) = \mathrm{clip}(z, -1, 1)$, $\bar{\xi}_i = \mathrm{clip}(\xi_i, -5, 5)$, $\xi_i \sim \mathcal{N}(0, I)$, η is the terminal noise scale, and ρ is the correction scale. Here X_{A_i} denotes the random terminal output of the corrected sampler.

Equation (24) defines the actual proxy-based terminal sampler. It is not a direct Euler–Maruyama discretization of the exact-score reference process in Appendix B.1, and the marginal-preservation result proved there is not asserted for this transition.

Geometry of the Implemented Correction. Although the proxy-based sampler does not inherit the exact marginal-preservation guarantee, skew symmetry still imposes the pointwise constraint:

$$q_\star^\top A_i q_\star = 0. \tag{25}$$

Hence $A_i q_\star$ has no component parallel to $q_\star$ in the normalized trajectory representation before clipping. Equivalently,

$$\frac{d}{d\varepsilon} \frac{1}{2} \left\| q_\star + \varepsilon A_i q_\star \right\|_2^2 \Big|_{\varepsilon=0} = q_\star^\top A_i q_\star = 0. \tag{26}$$

Thus, in the endpoint representation, the skew perturbation produces no first-order change in Euclidean magnitude and instead acts as a reorientation. This geometric property is distinct from the density-weighted divergence-free condition $\nabla \cdot (p_t A_i s_t) = 0$ established for the exact-score reference in Appendix B.1.

Bounded Terminal Deviation. The terminal-only implementation also permits a direct bound on the effect of the learned perturbation. Couple the corrected transition with its zero-correction stochastic counterpart using the same frozen prefix and the same terminal noise:

$$X_{A_i} = \mathcal{C}\big(\mathcal{C}(b_\star + \rho A_i q_\star) + \eta\bar{\xi}\big)\,, \tag{27}$$

$$X_0 = \mathcal{C}\big(\mathcal{C}(b_\star) + \eta\bar{\xi}\big)\,. \tag{28}$$

Since coordinate clipping is non-expansive in the Euclidean norm,

$$\|X_{A_i} - X_0\|_2 \le \|\mathcal{C}(b_\star + \rho A_i q_\star) - \mathcal{C}(b_\star)\|_2 \tag{29}$$

$$\le \rho\|A_i q_\star\|_2 \tag{30}$$

$$\le \rho\|A_i\|_2\|q_\star\|_2. \tag{31}$$

Thus, the perturbation magnitude bounds the terminal deviation.

Let P_{A_i} and P_0 denote the laws of X_{A_i} and X_0, respectively, conditioned on scene c_i, with expectation taken over the shared prefix randomness and terminal noise. This coupling yields the following Wasserstein-2 bound on the terminal distribution shift:

$$W_2^2(P_{A_i}, P_0) \le \mathbb{E}\big[\|X_{A_i} - X_0\|_2^2\big] \tag{32}$$

$$\le \rho^2\mathbb{E}\big[\|A_i q_\star\|_2^2\big]\,, \tag{33}$$

and therefore

$$W_2(P_{A_i}, P_0) \le \rho\left(\mathbb{E}\big[\|A_i q_\star\|_2^2\big]\right)^{1/2}. \tag{34}$$

Equation (34) directly relates the distributional deviation to the realized correction magnitude: reducing $\rho A_i q_\star$ necessarily tightens the upper bound on the Wasserstein distance from the corresponding zero-correction terminal sampler.

This Wasserstein bound holds for a general correction matrix and therefore quantifies the magnitude of the terminal perturbation independently of its parameterization. Skew symmetry provides the complementary geometric constraint in Eq. (25), removing the component of the correction parallel to $q_\star$. Reward optimization then determines which such structured terminal perturbations improve driving performance.

Extension to Different Initial States. The preceding bound uses a shared frozen prefix. More generally, consider a corrected trajectory initialized from Z_1 and a zero-correction trajectory initialized from Z_0 under the same condition c, producing prefix states $x_\star^1 = \Phi(Z_1)$ and $x_\star^0 = \Phi(Z_0)$, respectively. Coupling their terminal noise and using the non-expansiveness of $\mathcal{C}$ gives

$$\|X_A^1 - X_0^0\|_2 \le \|b_\star^1 - b_\star^0\|_2 + \rho\|Aq_\star^1\|_2. \tag{35}$$

Let Φ denote the implemented frozen prefix map from the initial noise to $x_\star$. If each frozen solver update is Lipschitz in its state, their finite composition is also Lipschitz; let L_Φ denote its Lipschitz constant. At the terminal step, suppose $v_\theta(\cdot, t_\star, c)$ is L_v-Lipschitz. From Eq. (7),

$$\|q_\star(x) - q_\star(y)\|_2 \le L_q\|x - y\|_2, \qquad L_q = \frac{1 + (1 - t_\star)L_v}{t_\star}. \tag{36}$$

Consequently, for

$$b_\star(x) = x + \Delta t\, v_\theta(x, t_\star, c) + \frac{1}{2}\eta^2 q_\star(x), \tag{37}$$

we have

$$\|b_\star(x) - b_\star(y)\|_2 \le L_b\|x - y\|_2, \qquad L_b = 1 + \Delta t L_v + \frac{\eta^2}{2}L_q. \tag{38}$$

Using $x_\star^1 = \Phi(Z_1)$ and $x_\star^0 = \Phi(Z_0)$ therefore gives

$$\|X_A^1 - X_0^0\|_2 \le L_b L_\Phi\|Z_1 - Z_0\|_2 + \rho\|Aq_\star^1\|_2. \tag{39}$$

Now let $Z_1 \sim \nu_1$ and $Z_0 \sim \nu_0$, and let P_A^1 and P_0^0 denote the corresponding terminal distributions. For any coupling $\pi \in \Pi(\nu_1, \nu_0)$, using the same terminal noise for the two transitions and applying Minkowski's inequality to Eq. (39) gives

$$\left(\mathbb{E}_\pi \|X_A^1 - X_0^0\|_2^2\right)^{1/2} \le L_b L_\Phi \left(\mathbb{E}_\pi \|Z_1 - Z_0\|_2^2\right)^{1/2} \tag{40}$$

$$+ \rho \left(\mathbb{E}\|Aq_\star^1\|_2^2\right)^{1/2}. \tag{41}$$

Taking the infimum over π yields

$$W_2(P_A^1, P_0^0) \le L_b L_\Phi W_2(\nu_1, \nu_0) + \rho \left(\mathbb{E}\|Aq_\star^1\|_2^2\right)^{1/2}. \tag{42}$$

The first term propagates the initial distribution shift through the frozen prefix, while the second isolates the additional deviation introduced by the terminal correction. For the same Gaussian prior, $\nu_1 = \nu_0 = \mathcal{N}(0, I)$, the first Wasserstein term vanishes, recovering Eq. (34); this concerns equality of the initial distributions and does not require independently drawn realizations to satisfy $Z_1 = Z_0$.

C Training Objectives and Optimization

C.1 Flow Planner Pretraining

We pretrain RF-DiT using the frozen TBM representations described in Section 4.1. Given an expert trajectory x_1, we sample $x_0 \sim \mathcal{N}(0, I)$ and construct the rectified-flow path

$$x_t = (1-t)x_0 + tx_1, \qquad t \sim \mathcal{U}(0,1). \tag{43}$$

The corresponding target velocity is $x_1 - x_0$. Conditioned on the cached TBM features c, the planner predicts $v_\theta(x_t, t, c)$ and minimizes

$$\mathcal{L}_{\mathrm{RF}} = \mathbb{E}_{x_0, x_1, t, c}\left[\|v_\theta(x_t, t, c) - (x_1 - x_0)\|_2^2\right]. \tag{44}$$

The current flow state also provides the endpoint estimate

$$\hat{x}_1 = x_t + (1-t)v_\theta(x_t, t, c). \tag{45}$$

We supervise this estimate with the expert trajectory and additionally match expert planar acceleration and jerk through a smoothness loss. The complete planner objective is

$$\mathcal{L}_{\mathrm{planner}} = \mathcal{L}_{\mathrm{RF}} + 0.2\mathcal{L}_{\mathrm{endpoint}} + 0.02\mathcal{L}_{\mathrm{smooth}}. \tag{46}$$

After pretraining, both TBM and RF-DiT remain frozen during correction learning and transfer.

C.2 Group-Relative Correction Learning

After flow pretraining, TBM and RF-DiT remain frozen and only the scene-specific matrix B_i is optimized through

$$A_i = B_i - B_i^\top. \tag{47}$$

Using the proxy-based terminal update defined in Appendix B.3, we cache the flow prefix up to the final state $x_\star^{(g)}$ and use the corresponding endpoint proxy $q_\star^{(g)}$. For rollout g, the corrected terminal mean is

$$\mu_i^{(g)} = \mathcal{C}\left(x_\star^{(g)} + \Delta t\, v_\theta(x_\star^{(g)}, t_\star, c_i) + \frac{1}{2}\eta^2 q_\star^{(g)} + \rho_{\mathrm{train}} A_i q_\star^{(g)}\right). \tag{48}$$

For source learning, we use $\rho_{\mathrm{train}} = 1$ and $\eta = 0.01$. The terminal noise and sample are

$$\xi_i^{(g)} \sim \mathcal{N}(0, I), \qquad \bar{\xi}_i^{(g)} = \mathrm{clip}\left(\xi_i^{(g)}, -5, 5\right), \qquad x_{1,i}^{(g)} = \mathcal{C}\left(\mu_i^{(g)} + \eta\bar{\xi}_i^{(g)}\right). \tag{49}$$

Conditional on the cached prefix and before the outer output clipping, B_i changes only the center of the terminal noise perturbation, while the noise scale η remains fixed.

For the G sampled trajectories, we compute the group-relative advantage defined in Eq. (8). Let

$$y_i^{(g)} = \mathrm{sg}\left[x_{1,i}^{(g)}\right] \tag{50}$$

be the detached terminal sample, where $\mathrm{sg}[\cdot]$ denotes the stop-gradient operator. Let $D = 3H$ denote the trajectory dimension and $\tau_{\log} > 0$ the log-likelihood clipping threshold. We use the Gaussian likelihood surrogate

$$\ell_i^{(g)} = \frac{1}{D} \sum_{d=1}^{D} \mathrm{clip}\left(\log \mathcal{N}\left(y_{i,d}^{(g)}; \mu_{i,d}^{(g)}, \eta^2\right), -\tau_{\log}, \tau_{\log}\right). \tag{51}$$

The matrix objective is

$$\mathcal{L}_{A_i} = -\frac{1}{KG} \sum_{g=1}^{G} \mathrm{sg}\left[\hat{a}_i^{(g)}\right] \ell_i^{(g)}, \qquad \mathcal{L}_{\mathrm{corr}} = \sum_i \mathcal{L}_{A_i}. \tag{52}$$

Here, $K = 5$ is the number of flow solver steps and G is the number of terminal samples per condition. The factor $1/K$ is the fixed normalization used in the implementation; correction learning itself acts only on the terminal step. $\mathcal{L}_{\mathrm{corr}}$ sums the matrix losses over source conditions optimized in the current batch. Gradients pass through $\mu_i^{(g)}$ and $A_i = B_i - B_i^\top$, while sampled trajectories, rewards, and group-relative advantages are detached.

Because the terminal noise and output are clipped, the Gaussian density above is not the exact likelihood of the implemented sampler. We use it as a differentiable likelihood surrogate and do not claim an unbiased likelihood-ratio estimator for the clipped transition. The optimization uses the group-relative reward normalization of GRPO without a PPO-style importance ratio, policy clipping, or KL regularization.

D FCT Evaluator and Inference Details

This appendix details the four components evaluated in Table 5: value learning, safety learning, scale adjustment, and candidate enhancement. All learned evaluators are trained offline and remain frozen during inference. We refer to the trajectory generated by the original frozen flow planner as Base, which is always retained as a separate reference and fallback. Corrected trajectories use a default correction scale of $\rho = 0.7$ before zero-shot scale adjustment. For HUGSIM (Table 2), we use the same aggregation protocol as the published baselines.

D.1 Dataset Repartition

We repartition the 85,109 Navtrain scenes into 57,756 training, 9,154 calibration, 9,125 monitoring, and 9,074 audit scenes. These partitions serve different roles in evaluator development rather than assuming recording-level statistical independence. Only the training split is used to update evaluator weights. The calibration split is used solely for post-hoc calibration of predicted risk probabilities, the monitoring split for training diagnostics, and the audit split for final source-domain sanity checks.

We further include 12,878 additional training scenes from the NAVSIM training data and assign them exclusively to the training role, yielding 70,634 training conditions in total ($57,756 + 12,878$). None of the calibration, monitoring, or audit scenes is used to update the learned evaluator weights.

D.2 Value Learning

Coarse Workset Construction. FCT starts from the Base trajectory and 32 corrected candidates retrieved by RRF using the condition and flow-trajectory descriptors. The Coarse evaluator independently encodes each corrected trajectory together with the ego state and TBM observations, predicting candidate utility, failure tendency, uncertainty, and a learned embedding. These outputs are combined with trajectory and risk diversity to construct a workset of 16 corrected candidates without using ground-truth candidate outcomes at selection time. The Base trajectory is excluded from these 16 slots and retained separately, forming a 17-candidate set consisting of Base and 16 corrected trajectories.

Joint Candidate Refinement. The Refiner jointly processes the retained candidates and scene context rather than scoring each trajectory independently. It produces scene-conditioned candidate representations and planning predictions while modeling interactions within the candidate set.

Training uses relative progress, safety, event-ranking, and failure-cliff objectives so that the representation reflects both trajectory quality and relative preference within the current scene. Based on these predictions, the Refiner ranks the retained candidates and establishes a provisional top-ranked candidate, which serves only as the reference for subsequent relative value and rank evaluation.

Relative Value and Rank Evaluation. The final value stage further evaluates the Refiner's provisional reference through relative candidate comparison. For each candidate m, we compare its representation z_m with that of the current reference z_b using

$$\phi_{m|b} = [z_m,\, z_b,\, z_m - z_b]. \tag{53}$$

Here z_m and z_b are the Refiner representations of candidate m and the current reference b, respectively. The feature $\phi_{m|b}$ combines their individual representations with the relative difference $z_m - z_b$. Two gradient-boosted models are trained on these pairwise features. The value model predicts the residual quality change relative to the reference, while the rank model predicts relative candidate ordering. Their outputs determine the value-based current choice, allowing the provisional reference to be replaced before the subsequent safety stages.

D.3 Safety Learning and Trajectory Refinement

Zero-Score Risk Augmentation. To improve supervision for catastrophic failures, we search the top 256 RRF neighbors for zero-PDMS trajectories. If the original 17-candidate set contains no zero-score trajectory, the highest-ranked valid zero-PDMS neighbor replaces its lowest-ranked corrected candidate, while Base and the candidate count remain unchanged. This uses only existing scored training trajectories and no validation or test labels. The augmentation introduces a zero-PDMS candidate into 13,690 additional training conditions, resulting in 25,975 of 70,634 conditions containing at least one zero-PDMS candidate.

TTC and DAC Repair. Starting from the value-selected trajectory, a shared dual-output risk network predicts TTC-penalty and DAC-noncompliance probabilities for each of the 17 candidates using planning fields sampled along the future trajectory, together with trajectory, ego-motion, and scene features. TTC and DAC repair are applied sequentially, replacing the current choice only when the corresponding predicted risk is reduced and the alternative satisfies fixed eligibility constraints on value degradation and trajectory consistency.

Map-Based Joint Risk Prediction. A separate scene-aware safety model predicts four candidate-level risk probabilities from map features sampled along each candidate trajectory and its vehicle footprint, together with global scene and kinematic features. These risks cover zero PDMS and zero-score failures of safety-critical PDMS components, and are learned from training-set scores and event records. The predicted risks are calibrated on the designated source-domain calibration split and combined with the predicted value to rerank the 17 candidates before the subsequent Gate.

Learned Safety Gate. The Gate learns how these risk estimates affect the final replacement decision. It is trained with a differentiable approximation to the deployed eligibility rules, which compare each alternative with the current choice using predicted value and calibrated failure probabilities. At deployment, a replacement must improve the risk-aware utility while satisfying fixed bounds on zero-PDMS, collision, DAC, TTC, and value degradation; otherwise the current trajectory is retained.

Post-Gate Kinematic Refinement. After the Gate, we apply a conservative deterministic refinement only when the selected trajectory violates kinematic constraints. We evaluate 15 bounded scale lateral shift variants, followed when necessary by a smoothness projection, a conservative blend toward Base, and a braking fallback. Each operation is designed to remain close to either the Gate-selected trajectory or the frozen Base trajectory, and the predicted yaw is left unchanged. A trajectory that already satisfies the constraints is kept unchanged. This module supports robustness under distribution shift and is applied identically in NAVSIM and HUGSIM.

D.4 Zero Shot Scale Adjustment

Correction-Strength Variants. A retrieved correction may have the appropriate direction but an unsuitable magnitude for a new scene. If the frozen selector chooses a corrected trajectory with matrix A, we therefore keep the same matrix, flow prefix, endpoint proxy, and terminal noise, and generate 32 variants by changing only the correction scale,

$$x^{(j)} = \mathcal{C}(x_{\text{base}} + \rho_j A q_\star)\,, \qquad \rho_j \in [0.5, 0.9], \tag{54}$$

where the 32 values are uniformly spaced. If the branch selects Base, no scale search is applied.

Frozen Re-evaluation. The 32 scale variants are converted back to physical trajectories, their TBM and trajectory features are recomputed, and the entire pipeline above is reapplied under a strictly zero shot setting. No target-scene reward, gradient update, or evaluator retraining is used. Scale adjustment therefore performs zero shot adaptation of correction magnitude using only the learned FCT evaluators.

D.5 Dual-Branch Candidate Enhancement

Complementary Retrieval Branches. The standard FCT branch retrieves 32 matrices by reciprocal-rank fusion of condition similarity and flow-trajectory similarity. Candidate enhancement adds a parallel flow-trajectory retrieval branch that directly matches the test-time Base flow trajectory against cached training flow trajectories by trajectory distance and retrieves the 32 nearest associated corrections. Both branches use the same correction bank, exclude sources from the query's parent recording, and retain duplicated matrices across branches when they arise naturally.

Branch-Wise Selection. Each branch independently forms Base plus 32 corrected trajectories, applies the complete frozen selector, and then performs the zero-shot scale adjustment described above. This produces one final trajectory from the RRF branch and one from the Flow-KNN branch.

Final Branch Selection. To reuse the same fixed 32-candidate selector interface, the two branch outputs are each repeated 16 times and combined with Base. These repetitions are identical copies rather than newly sampled trajectories. The same frozen selector is applied once more to choose among Base and the two branch outputs. Thus, candidate enhancement expands retrieval coverage through the additional branch that matches the test-time Base flow trajectory against cached training flow trajectories, without introducing another learned selector or target-domain optimization.

E ADAPTATION TO RECOGDRIVE

Correction Matrix Learning. We use a flow-matching planner built on the ReCogDrive Stage 2 backbone and learn correction matrices following Section 4.2. The planner remains frozen, and each source scene has an independent matrix:

$$A_s = B_s - B_s^\top, \qquad B_s \in \mathbb{R}^{24\times 24}. \tag{55}$$

The 24 dimensions represent the positions and headings of eight future poses. GRPO training uses groups of 12 source scenes, with 50 sampled trajectories and 250 updates per source. Driving rewards optimize only the correction matrices, which are applied at the final flow sampling step. This produces a correction library covering 85,109 `navtrain` scenes.

Candidate Generation. Subsequent FCT adaptation keeps the planner and correction library fixed, without further updates to A. We retrieve 32 correction matrices through RRF fusion of scene and generation-process descriptors. Each scene contains 32 corrected trajectories and an additional Base trajectory. Corrected candidates share the flow prefix, with matrix corrections applied only at the final sampling step.

Evaluator Adaptation. We recompute motion and TBM features for the ReCogDrive candidates using the trajectory and vehicle-footprint queries described in Appendix D.3. TBM remains frozen. Normalization statistics for candidate metadata are re-estimated from the ReCogDrive training data. We then retrain the coarse and joint evaluators, together with the value and risk predictors, using offline trajectory scores and risk labels from 85,109 `navtrain` scenes.

Inference and Results. During inference, coarse filtering retains 16 corrected trajectories for joint evaluation with Base. The final trajectory is selected using the value and risk decisions and the relative acceptance rules in Appendix D.3. All models and correction matrices remain frozen during inference. On 12,146 `navtest` scenes, PDMS increases from 86.12 to 90.3, while the number of zero-PDMS scenes decreases from 881 to 528. Table 3 reports the full metrics. The gains mainly appear in drivable-area compliance and ego progress, supporting the applicability of the correction and selection procedure to another planner.

F Computational Cost of Correction Learning

F.1 Optimization Scope and Measured Cost

During correction learning, TBM and the flow planner remain frozen. We update only the scene-specific matrix B_i, using the skew-symmetric parameterization in Section 4.2. With eight waypoints, each $B_i \in \mathbb{R}^{24\times24}$ has 576 trainable parameters. The resulting A_i has 276 independent degrees of freedom. A training group with 60 scenes therefore has 34,560 matrix parameters. Only the matrices in active groups require optimizer states.

For each group, we compute the frozen flow prefixes and terminal-transition quantities before matrix optimization. We reuse them across updates. Repeated computation consists mainly of terminal corrections, candidate reward evaluation, and matrix updates. It does not require repeated full flow sampling or backpropagation through the planner.

Table S1 reports a completed correction-learning run over 85,109 NAVSIM scenes in 1,422 groups. Each scene receives 250 updates with 50 candidates per update. The run uses eight A100 GPUs, 32 training processes, and 192 CPU scoring processes. Matrix optimization and merging take 14 h 19 min 42 s. Including cache reconstruction and startup preparation gives 15 h 22 min 20 s. The first interval corresponds to 114.63 allocated GPU-hours. These measurements cover correction learning, not backbone pretraining, subsequent library filtering, or evaluator training.

Table S1: **Measured cost of correction learning on NAVSIM.** Allocated GPU-hours use the matrix-optimization and merging interval. Storage counts only the dense matrices before filtering.

Item	Value
Training scenes / groups	85,109 / 1,422
Updates per scene / candidates per update	250 / 50
GPUs	8×A100
Training / CPU scoring processes	32 / 192
Trainable parameters per scene	576
Independent degrees of freedom per A_i	276
Matrix optimization and merging	14 h 19 min 42 s
Including cache reconstruction and startup	15 h 22 min 20 s
Allocated GPU-hours	114.63 GPU-h
Dense matrices before filtering (FP32)	∼187 MiB

The 85,109 learned matrices contain 49,022,784 elements in total, requiring approximately 187 MiB in FP32. This excludes descriptors and file-format overhead. It is the storage of the full source collection, not the filtered library used at inference. The parameter count per active group determines the matrix optimizer state, while the full collection determines storage.

F.2 Five-Step Flow-GRPO Profiling

To examine repeated network computation, we profile a five-step Flow-GRPO implementation that updates the Stage 2 backbone while keeping Stage 1 frozen. It uses the same NAVSIM reward-scoring protocol and 50 candidates per scene. The profiling run uses eight A100 GPUs, eight synchronized training processes, and 48 CPU scoring processes. One synchronized update processes up to 480 scenes and 24,000 trajectories. Table S2 gives the timing breakdown.

Table S2: **Five-step Flow-GRPO timing from a short profiling run.** Each synchronized update processes up to 480 scenes with 50 candidates each.

Component	Time per update
Scene-feature loading	0.49 s
Five-step trajectory generation	2.76 s
PDM simulation and reward scoring	5.08 s
Network forward/backward	7.93 s
Synchronization, optimizer, and waiting	0.22 s
Total	16.48 s

Trajectory generation and network forward/backward take 10.69 s together, or 64.9% of the measured update time. Freezing the planner and reusing its prefixes avoid these repeated network operations during matrix optimization. Both implementations still require candidate reward evaluation.

For a nominal budget of 85,109 scenes, 50 candidates per scene, and 250 exposures, the measured update time gives

$$N_{\text{updates}} = 250 \left\lceil \frac{85{,}109}{480} \right\rceil = 44{,}500, \qquad T_{\text{proj}} = 44{,}500 \times 16.48\ \text{s} \approx 203.7\ \text{h}. \tag{56}$$

This is a workload projection from short profiling, not a completed training run or the time needed to reach a given PDMS. It does not assume that the shared backbone needs 250 dataset exposures to reach its best performance. Correction learning and Flow-GRPO also use different CPU scoring concurrency (192 versus 48 processes). We therefore do not interpret their wall-clock times as a controlled training-speed comparison.

F.3 GPU Memory Usage

Each GPU runs four matrix-training processes. Peak GPU memory usage over the full matrix-optimization run is approximately 3.95 GiB per device.

The Flow-GRPO profiling run records a PyTorch tensor-allocation peak of 18.62 GiB with computation chunk size 512. A separate run with chunk size 256 records 11.42 GiB. Because the measurement procedures differ, these values do not establish a memory-reduction ratio. Flow-GRPO requires backbone activations, parameter gradients, and optimizer states. Matrix optimization freezes the backbone and stores gradients and optimizer states only for active matrices.

G Qualitative Analysis of Skew-Symmetric Corrections

Figure 5 compares four matrix settings under shared sampling in an unseen NAVSIM scene.

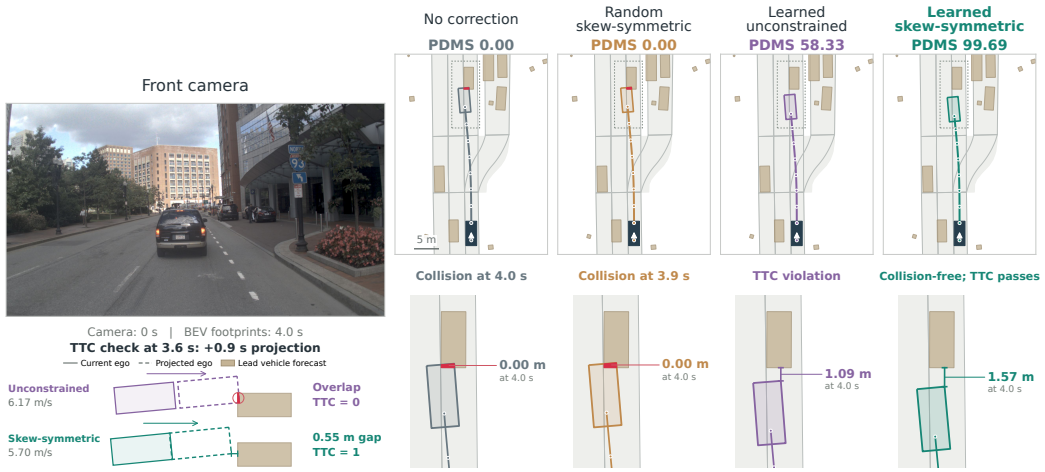

Figure 5: **A qualitative comparison of matrix corrections.** Under shared sampling, a norm-matched random skew-symmetric matrix fails to avoid collision. The learned unconstrained matrix avoids collision but fails the TTC check, scoring 58.33 PDMS. The learned skew-symmetric matrix adjusts speed and clearance to pass both checks, reaching 99.69 PDMS. Close-ups show the executed vehicle footprints and TTC projections.

What the skew constraint contributes. For the shared endpoint proxy $q_\star$, the correction before clipping is $\delta_A = \rho A q_\star$ (Appendix B.3). Skew symmetry imposes

$$q_\star^\top A q_\star = 0, \qquad q_\star^\top M q_\star = q_\star^\top \frac{M + M^\top}{2} q_\star, \tag{57}$$

where M is an unconstrained matrix. The symmetric part of M can introduce a component along $q_\star$. The skew constraint removes this component in the trajectory representation space. Driving rewards determine which directions within this constraint produce useful corrections.

Comparison under shared sampling. All four settings share the frozen planner, flow prefix, and initial and terminal noise. The three matrices use the same source scene and correction scale. The source and target come from different recordings. Both learned trajectories are final FCT selections. The random matrix is a paired control that matches the learned A in Frobenius norm. Matching matrix norms controls parameter scale, but does not match the magnitude or direction of $Aq_\star$. In this scene, the random correction remains close to Base and collides at 3.9 s; Base collides at 4.0 s. This comparison shows that skew symmetry and matrix norm alone are insufficient: reward learning must identify a useful correction.

Learned skew-symmetric versus unconstrained corrections. The learned M avoids collision but fails the TTC check, scoring 58.33 PDMS. The learned A passes both checks and reaches 99.69. NAVSIM evaluates TTC using constant-speed, constant-heading projections of the ego footprint (Dauner et al., 2024). At 3.6 s, the trajectories from M and A have speeds of 6.17 and 5.70 m/s, respectively. After a 0.9 s projection, the former overlaps the forecast front vehicle, while the latter retains 0.55 m clearance. The actual clearances at 4.0 s are 1.09 and 1.57 m, respectively. Thus, M produces a useful change, but its speed and spacing still trigger a projected collision. The learned A maintains enough clearance to pass the TTC check. TTC contributes $5/12$ of the weighted PDMS component, so failing this check removes 41.67 points when the multiplicative checks pass. The slightly lower progress of A accounts for its score being below 100.

This case illustrates the benefit of the speed and clearance adjustment learned under the skew constraint. It complements the aggregate comparison with unconstrained matrices in Table 4, supporting our choice of the skew-symmetric parameterization.