Normalizing Flows are Capable Trajectory Planners

Zhaohui Wang[1], **Bo Xu**[2], **Yingzhi Tang**[2], **He Liu**[2], **Chen Bai**[2,*]

[1]Peking University [2]Xpeng Motors

wzh2301112147@stu.pku.edu.cn

{xub2,tangyz3,he.liu.com,chenbai}@xiaopeng.com

Abstract

Generative models such as Diffusion Models and Flow Matching have improved multimodal trajectory planning in autonomous driving, but they still suffer from inference latency and lack of tractable probability densities, relying on heuristic trajectory selection. We propose BiDrive, a real-time end-to-end planning framework based on Normalizing Flows (NFs). By leveraging invertibility and exact likelihood estimation, BiDrive directly evaluates trajectory probabilities in a single forward pass, enabling principled maximum-likelihood-based decision making. To address the autoregressive bottleneck of expressive flows, we further develop a bidirectional distillation framework that compresses an autoregressive flow into a one-step generator, achieving real-time inference (50 FPS) while preserving modeling capacity, where the autoregressive model is used only during training as a probabilistic teacher. In addition, we introduce a latent-space optimization mechanism that incorporates differentiable safety and dynamic constraints for efficient test-time trajectory refinement. Experiments on NAVSIM closed-loop benchmarks demonstrate state-of-the-art performance in both safety and efficiency, highlighting the potential of NFs for real-time trajectory planning.

1 Introduction

Trajectory planning is a critical link between perception and control in end-to-end autonomous driving [1, 2, 3, 4]. Early systems, such as UniAD [1] and VAD [5], predominantly collapse the distribution into deterministic regression, which often suffers from mode collapse and cascading error accumulation in complex, long-tail scenarios. To capture the multimodal nature of human driving, recent research has shifted towards generative models, particularly Diffusion Models [6] and Flow Matching [7]. Despite their prowess in modeling diverse outcomes, these SDE/ODE-based paradigms shift the bottleneck from representation capacity to practical deployability: (1) **High Inference Latency:** The multi-step iterative sampling inherent in Diffusion and Flow Matching models limits their real-time deployment on embedded hardware; (2) **Intractable Probability:** Lacking exact probability densities, these models should rely on ad-hoc heuristic rules for trajectory selection rather than rigorous likelihood-based evaluation; (3) **Irreversibility:** Their non-bijective nature prevents analytical test-time refinement, making it difficult to instantly adjust trajectories that violate safety or dynamic constraints.

These challenges prompt a fundamental rethinking of the requirements for next-generation planning models. To bridge the aforementioned gaps, an ideal autonomous driving planner should simultaneously satisfy three core criteria: (1) Inference Efficiency to ensure low-latency response in dynamic environments; (2) Probabilistic Tractability to enable rigorous, likelihood-based trajectory selection; and (3) Analytical Refinability to support instant trajectory adjustment via invertible mappings. We observe that NFs [8, 9] offer an elegant theoretical framework that directly aligns

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

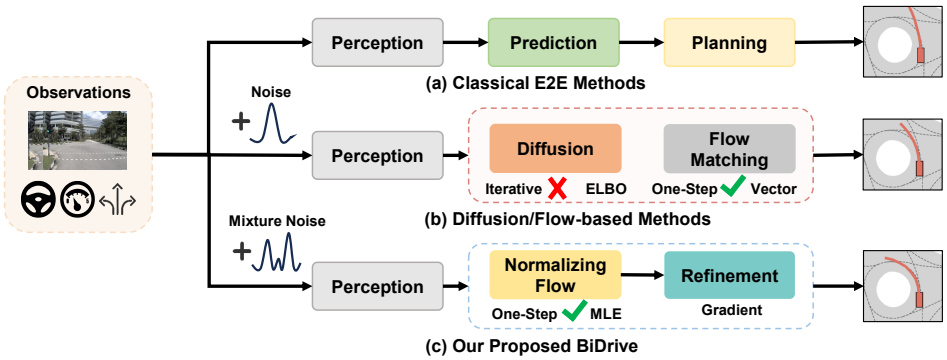

Figure 1: Comparison of different generative trajectory planning paradigms. (a) Perceptual decoupled learning-based model. (b) Previous methods rely on multi-step iterative generation and their generative mapping is fundamentally irreversible. (c) In contrast, BiDrive introduces NFs into end-to-end autonomous driving. It inherently achieves one-step generation and its strict bijection of NFs establishes invertible mapping.

with these objectives. Grounded in the change-of-variables formula, NFs enable the exact evaluation of conditional log-likelihoods $\log p_\theta(x|c)$, providing the required mathematical rigor for trajectory scoring. Crucially, the invertibility inherent in NFs establishes a strict bijection between data and latent spaces. This allows reward functions (e.g., safety or comfort) to be analytically backpropagated as exact Δz, enabling instant refinement of sampled trajectories. However, despite these compelling advantages, NFs remain critically underexplored in end-to-end planning. This is primarily because highly expressive Autoregressive Flows [10] suffer from the same sequential decoding bottleneck that limits real-time deployment.

Driven by these considerations, we present **BiDrive**, a novel generative trajectory planning framework that harmonizes probabilistic rigor with real-time performance. To the best of our knowledge, this is the first work to successfully integrate NFs and their exact-likelihood foundations into end-to-end autonomous driving. By reformulating trajectory generation as an invertible mapping, BiDrive concurrently addresses the trilemma of efficiency, tractability, and refinability. Specifically, we overcome the traditional NF latency bottleneck via a **bidirectional knowledge distillation** mechanism [11], which distills a high-capacity autoregressive prior into a NF-based one-step generator.

In summary, our primary contributions are as follows:

- We propose BiDrive, the first Normalizing Flow-based end-to-end planner. It replaces the brittle heuristic-based selection of previous generative models with a principled likelihood-based scoring mechanism grounded in MLE;
- We develop a bidirectional distillation framework that eliminates the sequential decoding bottleneck of expressive flows. This allows BiDrive to achieve a real-time inference speed of 50 FPS without compromising the complexity of the learned distribution;
- We introduce a gradient-based latent space refinement module. Leveraging the invertibility of NFs, this module enables analytical test-time adjustments to ensure the generated trajectories strictly adhere to safety and kinematic constraints;
- We demonstrate that BiDrive achieves state-of-the-art performance on the NAVSIM and NAVSIMv2 closed-loop benchmarks, significantly outperforming existing generative and deterministic baselines in both safety and efficiency.

2 Related Work

End-to-end autonomous driving. Early methods like TransFuser [12] pioneered multi-modal feature fusion, while UniAD [1] explicitly unified perception and planning within a dense Bird's-Eye-View (BEV) representation. To alleviate the prohibitive computational overhead of dense grids, subsequent paradigms shifted towards sparsity. VAD [5] and SparseDrive [13] utilized vectorized scene constraints and fully sparse queries to drastically improve structural awareness and inference efficiency, whereas ParaDrive [14] further tackled latency bottlenecks via a fully parallelized multi-task pipeline. VADv2 [15] formulated trajectory planning as a continuous probabilistic modeling task,

directly sampling driving actions from a predicted probability field to improve structural awareness and efficiency. Hydra-MDP [16] introduced a multi-target hydra-distillation framework, aggregating comprehensive rule-based and human-driven priors to enhance multimodal prediction. However, previous methods attempted to better fit human driving trajectories with a regression head [17, 18], lacking high-quality multimodal generation.

Generative models. Generative modeling experienced a paradigm shift with the advent of Score-Based Generative Models [19] and Denoising Diffusion Probabilistic Models [6], which modeled complex data distributions via stochastic differential equations (SDEs). To mitigate their extensive iterative sampling costs, DDIMs [20] and Consistency Models [21] were introduced to accelerate inference. Furthermore, Flow Matching [7] and Rectified Flow [22] emerged as elegant alternatives, constructing continuous-time ordinary differential equations (ODEs) through simulation-free vector field regression. Recent adaptations like MeanFlow [23] further optimized this process by learning mean velocity fields for single-step generation. However, despite their high sample quality, these SDE/ODE-based models fundamentally lacked tractable exact probability densities and strict analytical invertibility. NFs [8] elegantly addressed this through a sequence of differentiable bijections. Among them, autoregressive flows such as Masked Autoregressive Flow (MAF) [10, 24] provided exceptional expressivity compared to simple bipartite architectures (e.g., RealNVP [25]), as demonstrated by recent TarFlow [26]. Yet, their sequential point-by-point decoding remained prohibitively slow. To break this autoregressive bottleneck, building upon Inverse Autoregressive Flows (IAF) [27], BiFlow [11] recently introduced bidirectional probability density distillation, compressing the exact likelihood prior of a heavy MAF teacher into a fast, one-step generation network.

Multimodal trajectory planning. To effectively address the inherent multimodality and uncertainty of human driving, the field increasingly transitioned from deterministic regression to probabilistic generative modeling. Early pioneers like MotionDiffuser [28] and Diffusion Planner [29] demonstrated the efficacy of diffusion models in capturing diverse trajectory distributions through iterative denoising. However, to circumvent their prohibitive multi-step inference latency, subsequent frameworks introduced severe structural compromises. For instance, DiffusionDrive [30] relied on a truncated policy bounded by discrete heuristic anchors, while ConsistencyDrive [31] applied consistency distillation to force single-step outputs. Parallelly, Flow Matching gained immense traction due to its simpler ODE formulation. GoalFlow [32] heavily constrained the continuous generative space using explicit goal points, and MeanFuser [23] pushed inference limits via MeanFlow approximations seeded with heuristic Gaussian Mixture Noise.

3 Methodology

3.1 Preliminary

Problem formulation. The primary goal of trajectory planning is to infer the future trajectory of ego vehicle conditioned on sensor observations and high-level commands. Formally, at time step i, given the sensor observations comprising multi-view camera images $\mathcal{I}_i \in \mathbb{R}^{H\times W\times C}$, the ego-vehicle state $\mathcal{S}_i$, and navigation commands $\mathcal{C}_i$, the model aims to predict the future trajectory $\hat{\tau} = \{(x_i, y_i, \theta_i)\}_{i=1}^{T_f}$, where T denotes the prediction horizon. The definitions of each notation are shown in Appendix A.

Normalizing Flows. Normalizing Flows aim to transform a complex target data distribution $x \sim p_{data}$ into a simple base distribution e.g., a standard Gaussian distribution $z \sim p_0(z)$ through a sequence of invertible and differentiable functions denoted as f. According to the change of variables theorem in probability theory, the exact probability density of the target data x can be computed as:

$$p_\theta(x) = p_0(f(x))|\det(\frac{df(x)}{dx})| \tag{1}$$

where $\det$ denotes determinant of the Jacobian matrix. The network parameters θ are optimized via Maximum Likelihood Estimation (MLE). By taking the logarithm of both sides of Eq. 3, the training loss function is as follows:

$$\min_f -\log p_0(f(x)) - \log(|\det\left(\frac{df(x)}{dx}\right)|) \tag{2}$$

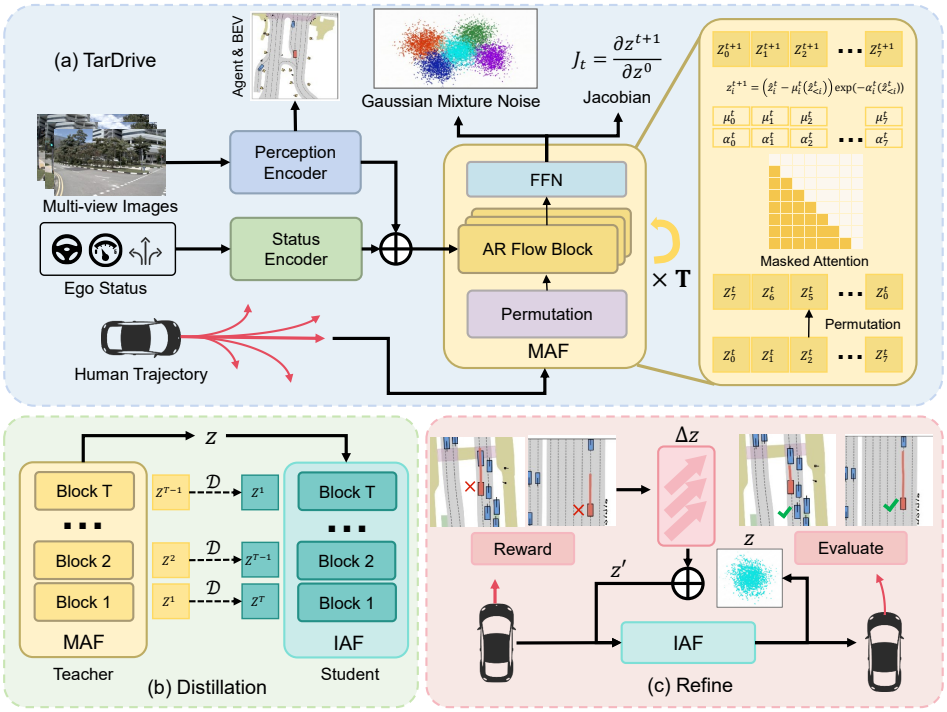

Figure 2: Overall architecture of the proposed BiDrive. (a) During training, a Masked Autoregressive Flow maps human trajectories into a Gaussian mixture noise, providing exact likelihoods. (b) We distill the autoregressive MAF teacher into a one-step IAF student for real-time inference. (c) The strict bijection enables test-time latent refinement via physical rewards for safe trajectory correction.

3.2 Framework

Overview. Given the sensor observations comprising multi-view camera images $\mathcal{I}_i \in \mathbb{R}^{H\times W\times C}$ at time step i, the ego-vehicle state $\mathcal{S}_i$, and navigation commands $\mathcal{C}_i$, BiDrive aims to predict the future trajectory $\hat{\tau} = \{(x_i, y_i, \theta_i)\}_{i=1}^{T_f}$, where T_f denotes the prediction horizon.

The overall architecture of our proposed framework BiDrive is illustrated in Fig. 2. To begin with, we present our MAF-based baseline model TarDrive, which utilizes a BEV encoder $\mathcal{E}_B$ and a Status Encoder $\mathcal{E}_S$ to process the multi-view input images $\mathcal{I}_i$ and ego-vehicle status $\mathcal{S}_i$ as illustrated in Fig. 2 (a). Simultaneously, the BEV features are routed through auxiliary branches to perform tasks including 3D object detection $\mathcal{E}_A$ and map segmentation $\mathcal{E}_M$. These auxiliary supervision signals implicitly enhance the structural awareness of the learned representations, yielding the aggregated conditional guidance feature $c = \{c_{bev}, c_S, c_{agent}, c_{map}\}$.

Instead of collapsing the future prediction into a deterministic regression, we model the exact conditional probability density $p_\theta(\tau|c)$ using NFs. As illustrated in the training pipeline, NFs aim to transform the complex ground-truth human trajectory distribution $\tau \sim p_{data}$ into a simple base distribution (e.g., a Gaussian mixture noise $z \sim p_0(z)$) through a sequence of invertible and differentiable functions denoted as $\mathcal{F}_\theta$. According to the change of variables theorem, the exact likelihood of the trajectory τ can be analytically computed as:

$$p_\theta(\tau|c) = p_0(\mathcal{F}_\theta(\tau, c)) \left|\det\left(\frac{\partial \mathcal{F}_\theta(\tau, c)}{\partial \tau}\right)\right| \tag{3}$$

where $\det$ denotes the determinant of the Jacobian matrix. By taking the logarithm of Eq. 3, our network parameters θ are directly optimized via Maximum Likelihood Estimation (MLE) using the following training objective:

$$\theta^* = \arg\min_\theta \left(-\log p_0(z) - \log\left|\det\left(\frac{\partial z}{\partial \tau}\right)\right|\right) \tag{4}$$

The extracted features $c = \{c_{bev}, c_S, c_{agent}, c_{map}\}$ serve as conditional guidance feature. During training, both the normalized ground-truth human trajectory and the guidance feature c are fed into the MAF which is constructed by stacking multiple invertible flow transformations f_t which are driven by Masked Attention blocks. Specifically, at the t-th flow layer, the Masked Attention network processes the current latent sequence z^t to predict a set of translation μ^t and scale α^t. These parameters are then used to transform z^t into the next latent state z^{t+1} via an element-wise affine transformation:

$$\begin{cases} \tilde{z}^t \leftarrow \pi^t(z^t), \\ z_i^{t+1} \leftarrow \begin{cases} \tilde{z}_i^t + \xi_t(c_a) & i = 0 \\ (\tilde{z}_i^t - \mu_i^t(\tilde{z}_{<i}^t; c_a)) \odot \exp(-\alpha_i^t(\tilde{z}_{<i}^t); c) + \xi_t(c_a) & i > 0 \end{cases} \end{cases} \tag{5}$$

where μ_i^t and α_i^t represent the parameters for the i-th dimension outputted by the masked attention module, and c_a represents anchor features. Crucially, to maintain the strict autoregressive mapping, we apply a causal mask within the attention mechanism. This ensures that the computation of μ_i^t and α_i^t depends exclusively on the strictly preceding sequence dimensions $z_{<i}^t$. This causal design not only allows all dimensions to be computed in parallel during training but also guarantees a lower-triangular Jacobian matrix J_t, enabling the elegant and exact computation of its determinant. Finally, a Permutation layer $\pi^t(z) = z_{N-1-i}$ is applied before passing the output to the next block, deterministically reversing the sequence ordering to ensure comprehensive feature interaction.

Since each flow transformation is strictly bijective, the corresponding inverse sampling or inference from z^{t+1} back to z^t (utilized during the sampling/generation phase) can be analytically derived as:

$$\begin{cases} \tilde{z}_i^t = \begin{cases} z_i^{t+1} - \xi_t(c_a), & i = 0 \\ (z_i^{t+1} - \xi_t(c_a)) \odot \exp(\alpha_i^t(\tilde{z}_{<i}^t)) + \mu_i^t(\tilde{z}_{<i}^t) & i > 0 \end{cases} \\ z^t = (\pi^t)^{-1}(\tilde{z}^t). \end{cases} \tag{6}$$

Note that during training, the exact likelihood computation in Eq. 3 can be fully parallelized by employing a causal mask within the attention mechanism. In contrast, during sampling, the generative process formulated in Eq. 6 is inherently sequential. Since the computation of $\tilde{z}_i$ depends on its preceding outputs $\tilde{z}_{<i}$, the trajectory must be decoded point-by-point in an autoregressive manner.

The determinant of each flow transformation is illustrated in Appendix B. Due to $\mathcal{F} = f_1 \circ f_2 \ldots f_T$, the determinant of the final Jacobian is given by:

$$\log(|\det(\frac{df(z)}{dz})|) = -\sum_{t=0}^{T-1}\sum_{i=1}^{N-1}\sum_{j=0}^{D-1}\alpha_i^t(\tilde{z}_{<i}^t)_j. \tag{7}$$

Likelihood. Previous approaches to trajectory proposal selection predominantly rely on the network's heuristic confidence scores for predefined anchors, lacking a rigorous mathematical foundation in probability density estimation. To address this limitation, we introduce a hierarchical filtering mechanism based on the MLE properties of NFs. Specifically, we first employ a lightweight Softmax head to isolate the Top-K proposals.

$$\mathcal{X}_{top} = \{x_k \mid \text{Rank}(P_{cls}(z|c)) \leq K\} \tag{8}$$

Subsequently, we exclusively search within reliable proposals to identify the trajectory that maximizes the exact NFs likelihood. The magnitude of this NFs likelihood mathematically quantifies the trajectory's density within the data distribution, inherently guaranteeing the selection of the trajectory that exhibits the highest fidelity to human driving behavior as:

$$x^* = \arg\max_{x_i \in \mathcal{X}_{top}} [\log p_{NF}(x_i|c)] \tag{9}$$

3.3 Bidirectional Knowledge Distillation

Although MAF inherits the powerful expressive capacity of NFs, its structural constraints mandate a strictly sequential generation process during sampling. This autoregressive bottleneck severely compromises the real-time inference speed of TarDrive. To alleviate this limitation, inspired by BiFlow [11], we introduce a novel knowledge distillation mechanism into our architecture.

Specifically, we define the pre-trained MAF as *forward model* TarDrive, denoted as $\mathcal{F}$, which maps the trajectory to Gaussian noise. Concurrently, we define an independent, trainable *reverse model*,

denoted as $\mathcal{R}$, dedicated to mapping the noise back to the trajectory. The overall distillation training scheme is illustrated in Fig. 2 (b).

The reverse model $\mathcal{R}$ comprises an identical number of blocks r as the forward model $\mathcal{F}$. However, instead of strictly adhering to the exact mathematical inverse of MAF $r = f^{-1}$, the reverse model utilizes an inverse autoregressive network [27] structure to achieve single-pass parallel generation. Formally, given the noise z and guidance feature c, the k-th layer of the reverse model computes the hidden state H^k_{stu} as:

$$z^t_{\text{r}} = \text{AttnBlock}^k \left(z^{t-1}_{\text{r}}; c, \theta^k_{\text{r}} \right) \tag{10}$$

where $z^0_r = \mathcal{F}(c)$, and θ^k_{r} represents the trainable parameters of the causal masked attention blocks.

To ensure the reverse model faithfully reconstructs the trajectory distribution without mode collapse, we employ the pre-trained forward model to supervise the reverse model. We force the student's hidden states to align with the teacher's corresponding latent variables in reverse order.

$$\mathcal{L}_{\text{align}}(z) = \sum_i \mathcal{D} \left(z^i, \varphi_i(z^i_r) \right) \tag{11}$$

where φ_i is a set of learnable projection heads. Furthermore, to enhance the robustness and generalization capabilities of the Forward Model, we introduce a noise augmentation strategy to the input trajectories. Specifically, given a ground-truth trajectory $\hat{\tau}$, we inject a controlled noise to yield the augmented input z. The relationship between the original and corrupted trajectory can be formulated as $y = x + \epsilon$ for $\epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$ where σ controls the noise scale. Consequently, the mapping process within the forward model inherently encompasses a denoising mechanism, robustly guiding the corrupted data distribution towards the standard latent prior.

Correspondingly, to ensure the reverse model comprehensively captures the probability density dynamics learned by the teacher, we extend the distillation objective to include the determinant of the Jacobian matrix as:

$$z \sim p_{\mathcal{F}}, z^r = \mathcal{R}(z), \tau = z^r + \sigma^2 \nabla_{z^r} \log p_{\mathcal{R}}(z^r) \tag{12}$$

The calculation of Eq. 12 is further illustrated in Appendix C. By explicitly predicting the determinant alongside the latent states, the reverse network mimics the topological and volumetric transformations of the masked flow.

3.4 Invertible Trajectory Refinement via Latent Optimization

In this section, we propose a novel Invertible Trajectory Refinement mechanism [33] that maximally exploits the strict mathematical bijection of NFs. Unlike ODE-based models, TarDrive acts as a highly efficient, explicit function $\tau = \mathcal{F}_\theta(z, c)$, mapping latent noise $z \sim \mathcal{N}(0, I)$ to the trajectory τ conditioned on features c. Because this mapping is strictly invertible and strictly differentiable, it allows us to add complex, differentiable physical constraints to Gaussian latent space.

Specifically, we define a differentiable scoring function $\mathcal{E}(\tau)$ that incorporates critical driving priors, such as dynamic collision avoidance and kinematic comfort. Given a generated trajectory $\tau_0 = \mathcal{F}_\theta(z_0, c)$, we compute the gradient of the energy function with respect to the sampling noise z using the chain rule:

$$\nabla_z \mathcal{E}_i = \frac{\partial \mathcal{E}_i(\tau)}{\partial \tau} \cdot \frac{\partial \mathcal{R}_\theta(z, c)}{\partial z} \tag{13}$$

where $\mathcal{E} = \mathcal{E}_i(\tau)$. Subsequently, we perform a gradient ascent step in the latent space to update the noise vector:

$$z' = z_0 + \eta_i \nabla_z \mathcal{E}_i \tag{14}$$

where η_i is the weight for $\mathcal{E}_i$. The refined trajectory is instantly obtained via a single forward pass $\tau' = \mathcal{F}_\theta(z', c)$. To effectively guide the latent refinement, we design a set of differentiable reward functions representing distinct driving priors. The total reward is formulated as a weighted sum $\mathcal{E}(\tau) = \sum_i \eta_i \mathcal{E}_i(\tau)$, ensuring a holistic refinement across safety, comfort and task progress. To guarantee safety, we establish a continuous, differentiable repulsion field around all predicted dynamic agents. Instead of using non-differentiable step functions, we penalize proximity using a Gaussian-like soft margin:

$$\mathcal{E}_{col}(\tau) = -\sum_{t=1}^{T} \sum_{j=1}^{N_a} \exp \left(-\frac{\|\tau_t - p^j_t\|^2}{2\sigma^2_{safe}} \right) \tag{15}$$

Table 1: Performance on the NAVSIMv1 navtest benchmark. "C" denotes camera, and "L" denotes LiDAR. * indicates results reported from the official papers. † indicates results trained with reinforcement learning.

Method	Input	NC↑	DAC↑	TTC↑	Comf.↑	EP↑	PDMS↑	Params	FPS	GFlops
TransFuser [12]	C & L	97.7	92.8	92.8	**100**	79.2	84.0	56	63	27.4
GoalFlow* [32]	C & L	98.3	93.8	94.3	**100**	79.8	85.7	62	10	45.8
ARTEMIS [34]	C & L	98.3	95.1	94.3	**100**	81.4	87.0	–	–	–
DiffusionDrive [30]	C & L	98.2	96.2	94.7	**100**	82.2	88.1	61	38	37.6
Epona [35]	C & L	97.9	95.1	93.8	99.9	80.4	86.2	2591	<1	5014
WoTE [36]	C & L	98.5	96.8	94.9	99.9	81.9	88.3	64	29	75
SeerDrive [37]	C & L	98.4	97.0	94.9	99.9	83.2	88.9	67	28	76.33
VADv2 [15]	C & L	97.2	89.1	91.6	**100**	76.0	80.9	–	–	–
DiffusionDriveV2† [38]	C & L	98.3	**97.9**	94.8	**100**	**87.5**	91.2	61	38	37.6
UniAD [1]	C	97.8	91.9	92.9	**100**	78.8	83.4	99	3	1060
PARADrive [14]	C	97.9	92.4	93.0	99.8	79.3	84.0	–	–	–
Hydra-MDP [16]	C	98.3	96.0	94.6	**100**	78.7	86.5	65	20	121.5
VGGDrive [39]	C	98.5	96.3	**95.5**	**100**	82.9	88.7	–	–	–
MeanFuser [23]	C	98.6	97.0	95.0	**100**	82.8	89.0	54.6	59	27.5
TarDrive (Ours)	C	95.6	94.6	89.1	97.2	80.5	86.4	59	16	54.4
BiDrive (Ours)	C	**98.8**	97.4	95.1	**100**	85.1	90.0	52	50	34.2
BiDrive-RL† (Ours)	C	**98.8**	97.8	95.3	**100**	87.2	**91.4**	52	50	34.2

where τ_t is the ego waypoint at time step t, p_t^j denotes the position of the j-th agent at step t, and σ_{safe} controls the safety margin.

To enforce physical smoothness, we compute the velocity v_t, acceleration a_t, and jerk j_t using finite differences over τ. The comfort energy penalizes dynamic states that exceed the physical limits of a passenger vehicle:

$$\mathcal{E}_{comf}(\tau) = -\sum_t \left(\max(0, \|a_t\| - a_{max}) + \max(0, \|j_t\| - j_{max})\right) \tag{16}$$

where a_{max} and j_{max} are the predefined thresholds for maximum comfortable acceleration and jerk.

A common failure mode in safety-critical refinement is the model learning to simply "freeze" the vehicle to avoid collisions, resulting in zero task progress. To counteract this, we incorporate a progress energy term that encourages the ego vehicle to maintain a context-appropriate target speed v_{target}:

$$\mathcal{E}_{prog}(\tau) = -\sum_{t=1}^{T} \left(\|v_t\| - v_{target}\right)^2 \tag{17}$$

Crucially, to mathematically guarantee that the gradient-updated latent z' remains within the Gaussian prior z, we introduce an explicit prior regularization term $\mathcal{E}_{prior}(z) = -||z' - z||^2$. Without this constraint, optimization for physical rewards could push the latent out-of-distribution, leading to structurally collapsed or physically unrealistic generative artifacts.

By projecting these analytical, physics-based constraints back into the Gaussian latent space, our framework performs test-time refinement. This plug-and-play mechanism significantly boosts the safety and compliance of generated trajectories without introducing any auxiliary training burden.

4 Experiments

The details of the datasets and implementation are provided in Appendix D and E.

4.1 Comparison with state of the art

As demonstrated in Table 1, our baseline TarDrive delivers a highly competitive performance with a PDMS of 86.4. Building upon this robust probabilistic prior, our distilled BiDrive establishes a new state-of-the-art on the NAVSIMv1 benchmark with a remarkable PDMS of 90.0. Notably, BiDrive effectively outperforms heavy sensor-fused architectures such as GoalFlow and DiffusionDrive, as

Table 2: Performance of the NAVSIMv2 navtest benchmark. $*$ denotes testing with the official checkpoint. † indicates results trained with reinforcement learning. All metrics are evaluated under the same setting.

Method	NC↑	DAC↑	DDC↑	TLC↑	EP↑	TTC↑	LK↑	HC↑	EC↑	EPDMS↑
Ego Status MLP [40]	93.1	77.9	92.7	99.6	86.0	91.5	89.4	**98.3**	85.4	64.0
TransFuser [12]	96.9	89.9	97.8	99.7	87.1	95.4	92.7	**98.3**	87.2	76.7
Hydra-MDP++ [16]	97.2	97.5	99.4	99.6	83.1	96.5	94.4	98.2	70.9	81.4
DriveSuprim [41]	97.5	96.5	99.4	99.6	88.4	96.6	95.5	**98.3**	77.0	83.1
ARTEMIS [34]	98.3	95.1	98.6	**99.8**	81.5	97.4	96.5	**98.3**	89.1	83.1
DiffusionDrive* [30]	98.2	96.1	99.5	**99.8**	87.4	97.3	96.9	**98.3**	87.5	84.8
MeanFuser* [23]	98.3	97.2	**99.6**	**99.8**	87.6	97.4	97.3	**98.3**	88.2	86.0
DiffusionDriveV2† [38]	97.7	96.6	99.2	**99.8**	**88.9**	97.2	96.0	97.8	**91.0**	85.5
TarDrive (Ours)	97.9	96.4	99.1	99.6	87.6	96.6	96.4	98.1	87.8	84.2
BiDrive (Ours)	**98.6**	97.5	99.5	**99.8**	88.1	97.5	97.3	**98.3**	88.6	86.4
BiDrive-RL† (Ours)	**98.6**	**97.6**	**99.6**	**99.8**	**88.9**	**97.7**	**97.5**	**98.3**	89.2	**86.8**

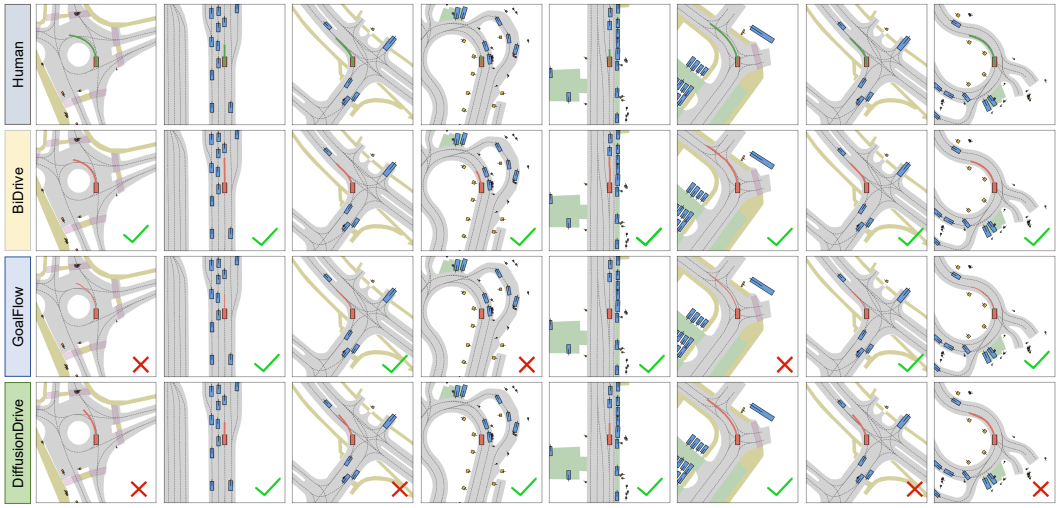

Figure 3: Visualization of trajectories on bird's eye view. × indicates that the trajectory is away from the center line or not smooth, while ✓ represents an accurate trajectory. The results highlight that BiDrive generates higher-quality trajectories compared to other methods. 🔍 Zoom in for details.

well as the recent vision-only MeanFuser. Furthermore, BiDrive achieves peak NC and DAC, proving that our Invertible Refinement performs precise trajectory corrections that surpass the inefficient "regenerate-and-test" fallbacks of prior generative methods, while elegantly resolving the sequential error accumulation observed in the TarDrive baseline. Finally, incorporating reinforcement learning elevates BiDrive-RL to a peak PDMS of 91.4, outperforming the multimodal DiffusionDriveV2. For more qualitative analysis and RL training details, please refer to Appendix F and Appendix H.

As shown in Table 2, we further evaluate our framework on the reactive NAVSIMv2 benchmark to assess error-recovery capabilities in highly dynamic scenarios. Consistent with previous results, while TarDrive yields a solid and reliable baseline with 84.2 EPDMS, BiDrive achieves performance gains to set the highest overall EPDMS of 86.4. By substantially outperforming recent flow-based and diffusion-based planners such as MeanFuser and DiffusionDrive, these results firmly demonstrate the superiority and robustness of NFs in complex closed-loop trajectory planning.

4.2 Qualitative Analysis

As illustrated in Fig. 3, BiDrive consistently generates superior trajectories compared to baselines. In complex topologies like the roundabout, while DiffusionDrive and GoalFlow struggle with spatial compliance and produce jagged trajectories, BiDrive strictly adheres to the lane center-line, outputting a highly smooth and kinematically natural curve. Furthermore, in dense interactive traffic,

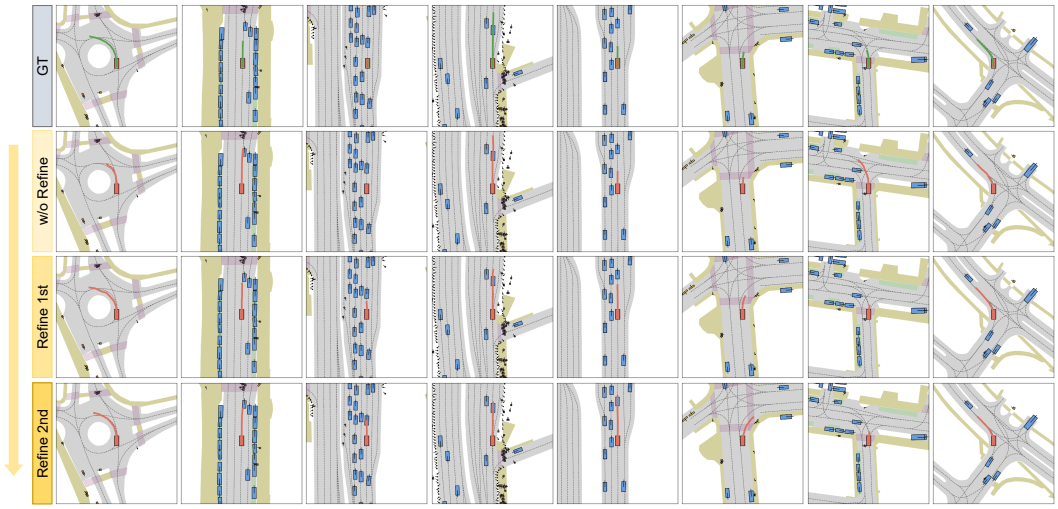

Figure 4: Visualization of refined trajectories. The results highlight that BiDrive generates higher quality trajectories after trajectory refinement. 🔍 Zoom in for details.

the generative baselines exhibit overly conservative planning with prematurely truncated trajectories. As illustrated in Fig. 4, empowered by our latent refinement, BiDrive projects a significantly longer trajectory, maximizing EP while rigorously maintaining safety and avoiding collisions. For more detailed analysis please refer to Appendix F.

4.3 Ablation study

Table 3: Ablation study on distillation methods on NAVSIMv1 dataset.

Method	NC ↑	DAC ↑	TTC ↑	Comf. ↑	EP ↑	PDMS ↑
Density Distillation	**98.8**	96.2	**95.2**	99.9	82.2	89.4
Hidden Distillation	98.1	96.0	94.9	**100**	80.1	88.9
Hidden Alignment	**98.8**	**97.4**	95.1	**100**	**85.1**	**90.0**

Effect of distillation methods. As shown in Table 3, while Density Distillation ensures basic safety, it struggles with spatial compliance and task progress. Hidden Alignment resolves this by explicitly synchronizing the intermediate latent representations between the forward and reverse models. This strategy perfectly preserves the teacher's strict safety and comfort priors while drastically boosting Drivable Area Compliance and Ego Progress, ultimately achieving the highest PDMS of 90.0. For more ablation studies, please refer to Appendix G.

Table 4: Ablation study on refinement methods on NAVSIMv1 dataset.

Collision	Comfort	Progress	NC ↑	DAC ↑	TTC ↑	Comf. ↑	EP ↑	PDMS ↑
			98.1	96.1	94.1	99.7	82.6	89.2
✓			98.6	96.8	94.5	99.7	82.2	89.5
✓	✓		98.1	96.2	94.5	**100**	82.7	89.4
✓		✓	97.9	96.0	94.0	99.8	**85.2**	89.8
✓	✓	✓	**98.8**	**97.4**	**95.1**	100	85.1	**90.0**

Effect of Refinement Energy Components. As shown in Table 4, we conduct ablation studies on latent refinement reward function. The comfort reward smooths jagged curves, the collision avoidance reward shortens paths to maintain safe buffers, and the progress reward ensures continuous forward

movement. This synergistic optimization reshapes initial predictions into safe, smooth, and highly efficient trajectories, directly enabling BiDrive's SOTA metrics PDMS = 90.0.

Table 5: Ablation study on iterations of refinement on NAVSIMv1 dataset.

Number	NC ↑	DAC ↑	TTC ↑	Comf. ↑	EP ↑	PDMS ↑
0	98.1	96.1	94.1	99.7	82.6	89.2
1	98.3	96.8	94.8	99.9	84.2	89.7
2	98.8	97.4	**95.1**	**100**	**85.1**	**90.0**
3	**99.0**	**97.5**	95.0	100	84.9	89.8

Effect of Refinement Iterations. Table 5 investigates the impact of refinement iterations on trajectory quality. We adopt 2 iterations as our main configuration, as it achieves an optimal balance between safety and driving efficiency, reaching peak performance with the highest PDMS and Ego Progress. Notably, increasing the refinement to 3 iterations leads to a slight decline in both EP and PDMS, despite yielding marginal gains in NC and DAC. This performance drop indicates that excessive refinement causes the reward function to over-optimize for safety constraints, resulting in overly conservative driving behavior that ultimately hinders forward task progress.

5 Conclusion

In this paper, we introduced BiDrive, pioneering the introduction of NFs in end-to-end autonomous driving. By leveraging the exact likelihood estimation and strict bijection properties of NFs, our framework achieves mathematically rigorous trajectory scoring and invertible latent space refinement. Furthermore, our distillation mechanism successfully enables real-time and one-step generation, establishing a new state-of-the-art on the NAVSIM benchmarks.

Limitations and future work. Despite these advancements, our framework presents several limitations. First, the baseline TarDrive model suffers from slow autoregressive inference speed and sub-optimal performance due to sequential error accumulation. Second, while BiDrive resolves the speed bottleneck, its training relies on the pre-trained TarDrive as a teacher model. Future work will focus on exploring teacher-free, single-pass normalizing flow architectures and enhancing the multi-modal generative diversity for complex driving scenarios.

References

[1] Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, et al. Planning-oriented autonomous driving. In *CVPR*, pages 17853–17862, 2023.

[2] Wenchao Sun, Xuewu Lin, Yining Shi, Chuang Zhang, Haoran Wu, and Sifa Zheng. Sparsedrive: End-to-end autonomous driving via sparse scene representation. In *ICRA*, 2025.

[3] Xiaosong Jia, Junqi You, Zhiyuan Zhang, and Junchi Yan. Drivetransformer: Unified transformer for scalable end-to-end autonomous driving. In *ICLR*, 2025.

[4] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020.

[5] 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 *ICCV*, pages 8340–8350, 2023.

[6] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, volume 33, pages 6840–6851, 2020.

[7] Yaron Lipman, Ricky T. Q. Chen, Heli Ben Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *ICLR*, 2023.

[8] George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021.

[9] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *NeurIPS*, 31, 2018.

[10] George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. In *NeurIPS*, volume 30, 2017.

[11] Yiyang Lu, Qiao Sun, Xianbang Wang, Zhicheng Jiang, Hanhong Zhao, and Kaiming He. Bidirectional normalizing flow: From data to noise and back. *arXiv preprint arXiv:2512.10953*, 2025.

[12] Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(11):12878–12895, 2022.

[13] Wenchao Sun, Xuewu Lin, Yining Shi, Chuang Zhang, Haoran Wu, and Sifa Zheng. Sparsedrive: End-to-end autonomous driving via sparse scene representation. In *ICRA*, pages 8795–8801, 2025.

[14] Xinshuo Weng, Boris Ivanovic, Yan Wang, Yue Wang, and Marco Pavone. Para-drive: Parallelized architecture for real-time autonomous driving. In *CVPR*, pages 15449–15458, 2024.

[15] Bo Jiang, Shaoyu Chen, Hao Gao, Bencheng Liao, Qian Zhang, Wenyu Liu, and Xinggang Wang. Vadv2: End-to-end autonomous driving via probabilistic planning. In *ICLR*, 2026.

[16] Zhenxin Li, Kailin Li, Shihao Wang, Shiyi Lan, Zhiding Yu, Yishen Ji, Zhiqi Li, Ziyue Zhu, Jan Kautz, Zuxuan Wu, et al. Hydra-mdp: End-to-end multimodal planning with multi-target hydra-distillation. *arXiv preprint arXiv:2406.06978*, 2024.

[17] Felipe Codevilla, Eder Santana, Antonio M López, and Adrien Gaidon. Exploring the limitations of behavior cloning for autonomous driving. In *ICCV*, pages 9329–9338, 2019.

[18] Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning. In *ICRA*, pages 4693–4700, 2018.

[19] Sungbin Lim, Eun-Bi Yoon, Taehyun Byun, Taewon Kang, Seungwoo Kim, Kyungjae Lee 0001, and Sungjoon Choi. Score-based generative modeling through stochastic evolution equations in hilbert spaces. In *NeurIPS*, 2023.

[20] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021.

[21] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *ICML*, 2023.

[22] Xingchao Liu, ChengYue Gong, and Qiang Liu 0001. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*, 2023.

[23] Junli Wang, Xueyi Liu, Yinan Zheng, Zebing Xing, Pengfei Li, Guang Li, Kun Ma, Guang Chen, Hangjun Ye, Zhongpu Xia, et al. Meanfuser: Fast one-step multi-modal trajectory generation and adaptive reconstruction via meanflow for end-to-end autonomous driving. *arXiv preprint arXiv:2602.20060*, 2026.

[24] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *NeurIPS*, 31, 2018.

[25] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

[26] Shuangfei Zhai, Ruixiang Zhang, Preetum Nakkiran, David Berthelot, Jiatao Gu, Huangjie Zheng, Tianrong Chen, Miguel Angel Bautista, Navdeep Jaitly, and Josh Susskind. Normalizing flows are capable generative models. *arXiv preprint arXiv:2412.06329*, 2024.

[27] Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *NeurIPS*, volume 29, 2016.

[28] Chiyu Jiang, Andre Cornman, Cheolho Park, Benjamin Sapp, Yin Zhou, Dragomir Anguelov, et al. Motiondiffuser: Controllable multi-agent motion prediction using diffusion. In *CVPR*, pages 9644–9653, 2023.

[29] Yinan Zheng, Ruiming Liang, Kexin Zheng, Jinliang Zheng, Liyuan Mao, Jianxiong Li, Weihao Gu, Rui Ai 0001, Shengbo Eben Li, Xianyuan Zhan, and Jingjing Liu. Diffusion-based planning for autonomous driving with flexible guidance. In *ICLR*, 2025.

[30] Bencheng Liao, Shaoyu Chen, Haoran Yin, Bo Jiang, Cheng Wang, Sixu Yan, Xinbang Zhang, Xiangyu Li, Ying Zhang, Qian Zhang, et al. Diffusiondrive: Truncated diffusion model for end-to-end autonomous driving. In *CVPR*, pages 12037–12047, 2025.

[31] Junli Wang and Qichao Zhang. Consistencydrive: Efficient end-to-end autonomous driving with consistency models. In *2025 10th International Conference on Control, Robotics and Cybernetics (CRC)*, pages 57–62, 2025.

[32] 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 *CVPR*, pages 1602–1611, 2025.

[33] Brandon Amos, Ivan Jimenez, Jacob Sacks, Byron Boots, and J Zico Kolter. Differentiable mpc for end-to-end planning and control. *NeurIPS*, 31, 2018.

[34] Renju Feng, Ning Xi, Duanfeng Chu, Rukang Wang, Zejian Deng, Anzheng Wang, Liping Lu, Jinxiang Wang, and Yanjun Huang. Artemis: Autoregressive end-to-end trajectory planning with mixture of experts for autonomous driving. *IEEE Robotics and Automation Letters*, 11(1):226–233, 2025.

[35] Kaiwen Zhang, Zhenyu Tang, Xiaotao Hu, Xingang Pan, Xiaoyang Guo, Yuan Liu, Jingwei Huang, Li Yuan, Qian Zhang, Xiao-Xiao Long, et al. Epona: Autoregressive diffusion world model for autonomous driving. In *ICCV*, pages 27220–27230, 2025.

[36] 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 *ICCV*, pages 27137–27146, 2025.

[37] 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. *arXiv preprint arXiv:2510.11092*, 2025.

[38] 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 preprint arXiv:2512.07745*, 2025.

[39] Jie Wang, Guang Li, Zhijian Huang, Chenxu Dang, Hangjun Ye, Yahong Han, and Long Chen. Vggdrive: Empowering vision-language models with cross-view geometric grounding for autonomous driving. *arXiv preprint arXiv:2602.20794*, 2026.

[40] Zhiqi Li, Zhiding Yu, Shiyi Lan, Jiahan Li, Jan Kautz, Tong Lu, and Jose M Alvarez. Is ego status all you need for open-loop end-to-end autonomous driving? In *CVPR*, pages 14864–14873, 2024.

[41] 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. *arXiv preprint arXiv:2506.06659*, 2025.

[42] Daniel Dauner, Marcel Hallgarten, Tianyu Li, Xinshuo Weng, Zhiyu Huang, Zetong Yang, Hongyang Li, Igor Gilitschenski, Boris Ivanovic, Marco Pavone, et al. Navsim: Data-driven non-reactive autonomous vehicle simulation and benchmarking. In *NeurIPS*, volume 37, pages 28706–28719, 2024.

[43] Napat Karnchanachari, Dimitris Geromichalos, Kok Seang Tan, Nanxiang Li, Christopher Eriksen, Shakiba Yaghoubi, Noushin Mehdipour, Gianmarco Bernasconi, Whye Kit Fong, Yiluan Guo, et al. Towards learning-based planning: The nuplan benchmark for real-world autonomous driving. In *ICRA*, pages 629–636, 2024.

[44] Wei Cao, Marcel Hallgarten, Tianyu Li, Daniel Dauner, Xunjiang Gu, Caojun Wang, Yakov Miron, Marco Aiello, Hongyang Li, Igor Gilitschenski, et al. Pseudo-simulation for autonomous driving. *arXiv preprint arXiv:2506.04218*, 2025.

[45] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[46] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Appendix of Normalizing Flows are Capable Trajectory Planners

A Notations

As shown in Table 6, we provide a lookup table for notations used in the paper.

Table 6: Notation and Description

Notation	Description
$\mathcal{I}$	multi-view images
τ	multi-modal trajectories
$\mathcal{S}$	ego status
$\mathcal{C}$	navigation commands
$H\&W$	spatial dimensions of the images
M	the number of trajectory modes
z	the Gaussian Mixture Noise
J_t	the t-th jacobian matrix of flow transformation
z^t	the t-th latent variable of forward model
$\mathcal{F}$	the forward model
z_r	the sampling trajectory
z_t^r	the t-th latent variable of reverse model
$\mathcal{R}$	the reverse model

B Calculation of Determinant

In our proposed framework, the ability to perform exact likelihood scoring relies fundamentally on the tractable Jacobian determinant of the Masked Autoregressive Flow (MAF) and Inverse Autoregressive Flow (IAF). This section provides the detailed derivation of the Jacobian matrix and its corresponding log-determinant.

Let $x = (x_1, x_2, \ldots, x_D) \in \mathbb{R}^{D\times 3}$ represent a continuous trajectory in the physical space, and $z = (z_1, z_2, \ldots, z_D) \in \mathbb{R}^{D\times 3}$ represent the corresponding representation in the latent Gaussian space. The forward pass of MAF (mapping data x to latent noise z) applies an autoregressive affine transformation. For each dimension $i \in \{1, 2, \ldots, D\}$, the transformation is defined as Eq. 5.

To compute the exact likelihood using the change-of-variables formula, we must evaluate the determinant of the Jacobian matrix $\mathbf{J} \in \mathbb{R}^{D\times D}$, where each element is defined as $\mathbf{J}_{i,j} = \frac{\partial z_{t+1}}{\partial z_t}$.

Based on the autoregressive dependency, we can analyze the partial derivatives in three cases: 1) Upper-diagonal elements ($j > i$): Since z_{t+1} does not depend on any future variables z_t, the derivative is strictly zero: $\frac{\partial z_{t+1}}{\partial z_t} = 0$. 2) Diagonal elements ($j = i$): The derivative of z_{t+1} with respect to z_t only involves the scaling factor, yielding: $\frac{\partial z_{t+1}}{\partial z_t} = \exp\left(-\alpha_i(z_{1:i-1})\right)$. 3) Lower-diagonal elements ($j < i$): These derivatives involve the complex gradients of the neural networks computing μ_i and α_i. We denote these arbitrarily complex terms as $\mathbf{J}_{i,j} = *$. Consequently, the Jacobian matrix $\mathbf{J}$ forms a lower triangular matrix:

$$J_t = \frac{\partial z^{t+1}}{\partial z^t} = \begin{bmatrix} \frac{\partial z_0^{t+1}}{\partial z_0^t} & 0 & 0 & \cdots & 0 \\ \frac{\partial z_1^{t+1}}{\partial z_0^t} & \frac{\partial z_1^{t+1}}{\partial z_1^t} & 0 & \cdots & 0 \\ \frac{\partial z_2^{t+1}}{\partial z_0^t} & \frac{\partial z_2^{t+1}}{\partial z_1^t} & \frac{\partial z_2^{t+1}}{\partial z_2^t} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial z_n^{t+1}}{\partial z_0^t} & \frac{\partial z_n^{t+1}}{\partial z_1^t} & \frac{\partial z_n^{t+1}}{\partial z_2^t} & \cdots & \frac{\partial z_n^{t+1}}{\partial z_n^t} \end{bmatrix} \tag{18}$$

A fundamental property of linear algebra states that the determinant of any triangular matrix is simply the product of its diagonal entries. Therefore, we can completely bypass the computationally expensive $O(D^3)$ operation typically required for dense matrix determinants. The absolute determinant of our transformation simplifies to an $O(D)$ operation:

$$|\det(J_t)| = \prod_{i=1}^{D} \exp\left(-\alpha_i(z_{1:i-1})\right) \tag{19}$$

While Eq. 19 establishes the tractable determinant for a single flow transformation f_t, a complete forward model $\mathcal{F}$ is constructed by sequentially composing T transformations: $\mathcal{F} = f_T \circ f_{T-1} \circ \cdots \circ f_1$. According to the chain rule of calculus, the Jacobian matrix is the product of the individual Jacobian matrices:

$$\mathbf{J}_{\mathcal{F}} = \frac{\partial z^T}{\partial x} = \prod_{t=1}^{T} \frac{\partial z^t}{\partial z^{t-1}} = \prod_{t=1}^{T} J_t \tag{20}$$

where $z^0 = x$ represents the input data and $z^T = z$ denotes the final latent representation. Consequently, the determinant of the full model is the product of the individual layer determinants.

In practice, to prevent numerical underflow and to directly formulate the exact log-likelihood optimized in our MLE objective, we compute the log-determinant. This seamlessly transforms the product of determinants into a computationally stable summation over all T layers and D dimensions:

$$\log|\det(\mathbf{J}_{\mathcal{F}})| = \sum_{t=1}^{T} \log|\det(J_t)| = \sum_{t=1}^{T}\sum_{i=1}^{D} -\alpha_i^t(\tilde{z}_{<i}^{t-1}) \tag{21}$$

C Score Based Denoising

In this section, we provide the detailed derivation of Tweedie's formula, which bridges the gap between the score function and the optimal minimum mean square error denoiser. We consider a joint distribution (x, y) where the clean data $x \sim p_{\text{data}}$ and the noisy observation $y = x + \epsilon$ with Gaussian corruption $\epsilon \sim \mathcal{N}(0, \sigma^2\mathbf{I})$. Thus, the conditional distribution $p(y|x)$ is an isotropic Gaussian, yielding the gradient:

$$\nabla_y p(y|x) = -\frac{y-x}{\sigma^2} p(y|x) \implies xp(y|x) = yp(y|x) + \sigma^2 \nabla_y p(y|x) \tag{22}$$

To obtain the marginal relationship, we multiply both sides by the data prior $p_{\text{data}}(x)$ and integrate over all possible values of x:

$$\int xp(y|x)p_{\text{data}}(x)dx = \int yp(y|x)p_{\text{data}}(x)dx + \sigma^2 \int \nabla_y p(y|x)p_{\text{data}}(x)dx \tag{23}$$

Recognizing the joint distribution $p(x, y) = p(y|x)p_{\text{data}}(x)$ and applying the Leibniz integral rule to interchange the gradient and integral, we have:

$$\int xp(x, y)dx = y \int p(x, y)dx + \sigma^2 \nabla_y \int p(x, y)dx \tag{24}$$

Since the marginal distribution of the noisy data is defined as $q(y) = \int p(x, y)dx$, the equation simplifies to:

$$\int xp(x, y)dx = yq(y) + \sigma^2 \nabla_y q(y) \tag{25}$$

Finally, by dividing both sides by $q(y)$ and applying Bayes' theorem $p(x|y) = p(x, y)/q(y)$ and the logarithmic derivative identity $\nabla_y \log q(y) = \nabla_y q(y)/q(y)$, we arrive at Tweedie's formula:

$$\mathbb{E}[x|y] = y + \sigma^2 \nabla_y \log q(y) \tag{26}$$

D Dataset and Metrics

NAVSIMv1 [42] is a large-scale, data-driven pseudo-simulation dataset designed for end-to-end autonomous driving. Derived from the real-world driving logs of nuPlan [43], it provides comprehensive multi-modal sensor inputs, including multi-view images and LiDAR. By explicitly excluding stationary and constant-velocity scenes, NAVSIMv1 offers a highly challenging benchmark.

NAVSIMv2 [44] extends this framework by addressing the inherent limitations of open-loop evaluation against compounding errors. To target next-generation planning challenges, NAVSIMv2 introduces a reactive simulation environment with responsive background traffic and synthetic novel view generation.

Metrics. NAVSIMv1 evaluates models using a non-reactive pseudo-simulation paradigm, primarily quantified by the PDM Score (PDMS), a closed-loop metric assessing collision avoidance, drivable area compliance and driving progress. No collisions (NC), drivable area compliance (DAC), time-to-collision (TTC), comfort (Comf.) and Ego Progress (EP) are also reported. For NAVSIMv2, model performance is evaluated using the rigorously Extended PDM Score (EPDMS), which comprehensively reflects a model's robustness and error-recovery capabilities in highly dynamic and interactive scenarios.

E Implementation Details

We adopt the same perception modules in DiffusionDrive [30] for fair comparison. In the NF layer, we employ spatial causal attention to interact with guidance features. We stack 4 NF layers or 4 attention layers at a hidden dimension of 256 as a trajectory head. We use camera image resolution 1024×256 without BEV LiDAR as input. Our model is trained on navtrain split from scratch for 100 epochs for both stage 1 and stage 2 with AdamW optimizer on 8 NVIDIA H800 GPUs with total batch size of 512. We freeze the perception models and NF layers during stage 2. No test-time augmentation is applied. The model samples from a 4-component Gaussian mixture and outputs an 8-waypoint trajectory over 4 seconds at 2 Hz for evaluation.

F Additional Qualitative Results

As shown in Figure 3, BiDrive consistently generates higher-quality trajectories compared to generative baselines like DiffusionDrive and GoalFlow. In complex topologies such as roundabouts and intersections (Col 1, 3, 4 and 7), BiDrive strictly adheres to the lane center-lines and outputs remarkably smooth kinematic arcs, effectively eliminating the dangerous lateral deviations and rigid curves often produced by anchor-based or heuristic models. Furthermore, in dense traffic conditions (Col 2, 5, 6 and 8), BiDrive confidently projects significantly longer collision-free trajectories than its conservative counterparts.

As shown in Figure 5, BiDrive refines its initial trajectory planning through iterative gradient refinement in the latent space. This refinement effectively balances three distinct driving priors: a comfort reward that smooths jagged paths into natural kinematic arcs, a collision avoidance reward that detects unsafe proximity, and a progress reward that prevents overly conservative stopping to ensure continuous forward movement. Together, these constraints iteratively shape the initial predictions into smooth, safe, and highly efficient paths, directly contributing to confident driving behavior. For more visualization results, please refer to supplementary material.

As shown in Figure 6, BiDrive demonstrates superior closed-loop planning from the front-camera perspective compared to generative baselines. In safety-critical scenarios involving dense intersections and crossing pedestrians, BiDrive reliably executes safe yielding behaviors, whereas DiffusionDrive frequently generates dangerous collisions. Furthermore, empowered by our latent refinement, BiDrive maintains perfectly stable kinematic control during car-following and strictly adheres to navigational commands in complex topologies, effectively eliminating the structural violations commonly observed in GoalFlow.

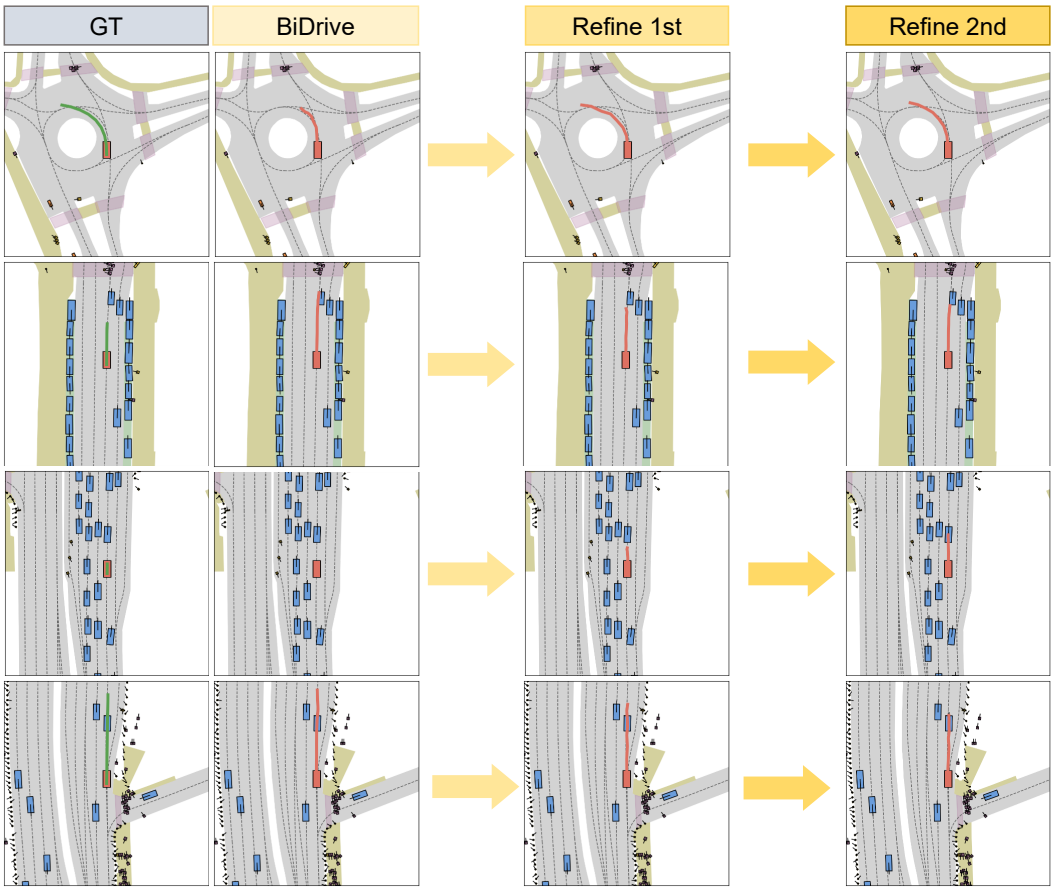

Figure 5: Visualization of refined trajectories. The results highlight that BiDrive generates higher-quality trajectories after trajectory refinement.

G Additional Ablation Studies

As shown in Table 7, BiDrive achieves an exceptional performance-efficiency trade-off, establishing a SOTA PDMS of 90.0 at a real-time 50 FPS with only 52M parameters and 34.2 GFlops. Our bidirectional distillation effectively eliminates the autoregressive bottleneck of the TarDrive baseline, boosting overall speed from 16 to 50 FPS. Furthermore, BiDrive significantly outperforms DiffusionDrive in both speed and driving metrics.

Effect of blocks. To systematically evaluate our framework, we compare TarDrive and BiDrive across varying flow blocks $T \in \{2, 4, 6, 8\}$ as shown in Fig. 7. Although BiDrive introduces a larger parameter footprint, it fundamentally breaks the autoregressive bottleneck, achieving a real-time 26 FPS even at $T = 8$. Furthermore, BiDrive overtakes TarDrive in NAVSIM PDM Score, demonstrating that its bidirectional parallel architecture better captures global trajectory than strict causal masking. This indicates that our probability density distillation not only accelerates inference but also acts as an implicit regularization mechanism, yielding smoother and safer trajectory proposals.

Effect of Gaussian Mixture Components. Table 8 investigates the impact of the number of components in the Gaussian Mixture prior. Using a single Gaussian prior yields suboptimal overall performance 89.1 PDMS. This indicates that a standard Gaussian prior is insufficient to capture the inherently diverse and multimodal nature of human driving behavior. Increasing the number of mixture components to 4 drastically enhances the model's expressivity, leading to peak performance across overall PDMS 90.0. Therefore, we adopt 4 components as the optimal configuration to balance generative diversity and training stability.

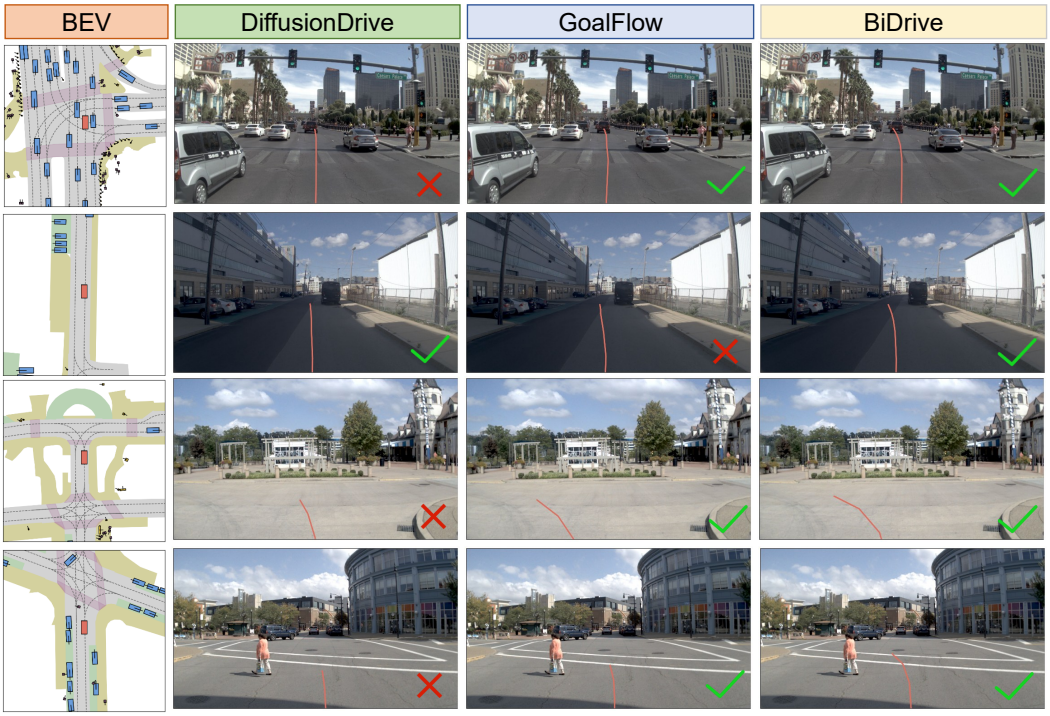

Figure 6: Visualization of trajectories on front camera. × indicates that the trajectory results in collision or away from command, while ✓ represents an accurate trajectory. The results highlight that BiDrive generates higher-quality trajectories compared to other methods.

Table 7: **Model parameter size, inference speed, and performance.** "*Dim*" indicates the number of hidden neurons in the model, "*FPS*" represents the median inference speed over multiple runs, and "*FPS*" refers to the inference speed of trajectory planning excluding the perception encoder.

Method	Dim	Params (M)	PDMS ↑	Plan FPS ↑	FPS ↑	GFlops
Epona	–	2591	86.2	–	0.15	5014
TransFuser	256	56	84.0	**3934**	**63**	27.4
GoalFlow	256	62	85.7	11	10	45.8
Hydra-MDP	256	65	86.5	25	20	121.5
SeerDrive	256	67	88.9	–	28	76.33
WoTE	256	64	88.3	–	29	75
UniAD	256	99	83.4	115	3	1060
DiffusionDrive	256	61	88.1	75	38	37.6
MeanFuser	128	54.6	<u>89.0</u>	<u>434</u>	<u>59</u>	27.5
TarDrive	256	59	86.4	32	16	54.4
BiDrive	256	52	**90.0**	93	50	34.2

H Additional Experiments with Reinforcement Learning

To further align the generative distribution of BiDrive with safety-critical driving requirements, we employ a reinforcement learning fine-tuning stage. Unlike conventional RL frameworks that rely on Markov Decision Process (MDP) formulations [38], our BiDrive operates as a one-step generator, allowing for a more efficient single-step optimization.

Algorithmic Formulation. We adopt the Group Relative Policy Optimization (GRPO) [45, 46] to optimize the distilled BiDrive policy π_θ as shown in Fig. 8. Following the design of DiffusionDriveV2 [38], we leverage the intrinsic multimodality of our framework by utilizing M driving

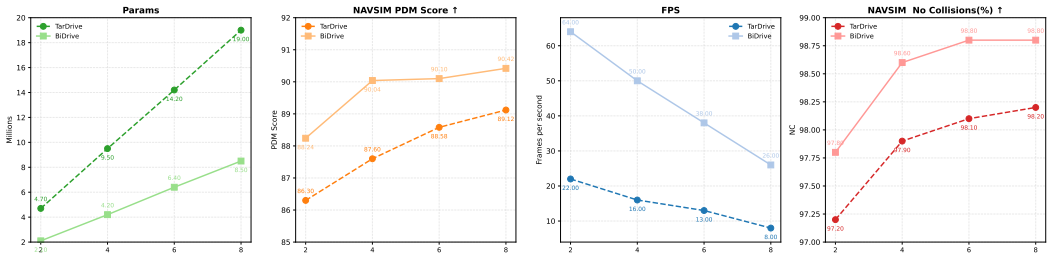

Figure 7: Ablation study on number of blocks on NAVSIMv1 dataset. Increasing the number of blocks enhances performance while reducing FPS.

Table 8: Ablation study on Gaussian components on NAVSIMv1 dataset.

Number	NC ↑	DAC ↑	TTC ↑	Comf. ↑	EP ↑	PDMS ↑
1	98.3	95.6	94.8	100	84.8	89.1
2	98.6	94.1	95.0	100	84.9	89.7
4	**98.8**	**97.4**	95.1	**100**	**85.1**	**90.0**
8	98.0	96.9	**95.2**	99.9	83.2	88.9

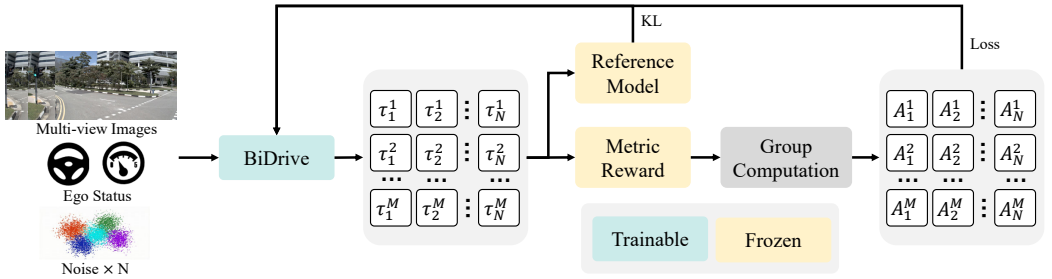

Figure 8: Illustration of the reinforcement learning pipeline for BiDrive.

anchors. For each anchor m, we sample a group of N noise vectors $\{z_{m,n}\}_{n=1}^{N}$ from the Gaussian prior to generate a collection of trajectory proposals $\{\tau_{m,n}\}_{n=1}^{N}$.

The advantage function $A_{m,n}$ is computed relative to the mean reward within each anchor group, eliminating the need for a value model:

$$A_{m,n} = \frac{R(\tau_{m,n}) - \frac{1}{N}\sum_{k=1}^{N} R(\tau_{m,k})}{\text{std}(\{R(\tau_{m,k})\}_{k=1}^{N})} \tag{27}$$

where the reward $R(\tau)$ is defined as the sum of the energy functions $\mathcal{E}_{col}, \mathcal{E}_{comf}$, and $\mathcal{E}_{prog}$ described in Section 3.4.

Objective and Regularization. The policy is optimized using the clipped surrogate objective of PPO, regularized by a KL divergence penalty to maintain stability and prevent mode collapse:

$$\mathcal{J}(\theta) = \mathbb{E}\left[\frac{1}{MN}\sum_{m,n}\text{clip}(\pi_\theta, A_{m,n}) - \beta\mathbb{D}_{KL}(\pi_\theta||\pi_{ref})\right] \tag{28}$$

where π_{ref} is the frozen supervised-trained BiDrive model. Since our Normalizing Flow provides an analytical expression for the log-probability $\log \pi_\theta(\tau|c)$ via the Jacobian determinant:

$$\log \pi_\theta(\tau|c) = \log p_0(z) - \log\left|\det \frac{\partial \mathcal{F}_\theta(z,c)}{\partial z}\right|$$

The KL term can be calculated directly during the one-step generation. This alignment stage ensures that while BiDrive remains a fast, one-step planner, its trajectory outputs are specifically optimized

for the non-differentiable PDMS metrics on the NAVSIM benchmark. As demonstrated in our main results (Table 1), this GRPO-based fine-tuning effectively explores superior trajectories while constraining low-quality modes, ultimately pushing BiDrive-RL to achieve a SOTA PDMS of 91.4.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims illustrate in Figure 1 in Section 1 (introduction). Our main contributions in this paper are summarized as follows:

   - We propose TarDrive, marking the first successful introduction of NFs into end-to-end autonomous driving. By leveraging explicit MLE, our method provides a mathematically hierarchical filtering mechanism that fundamentally replaces brittle heuristic rules for optimal trajectory selection.
   - We introduce an invertible trajectory refinement mechanism that exploits the strict bijection property of NFs. This allows collision costs to be analytically backpropagated as exact gradient shifts in the latent space.
   - We design a knowledge distillation mechanism to overcome the inherent inference latency of highly expressive NFs.
   - We conduct extensive experiments on NAVSIM and NAVSIM v2 closed-loop benchmarks, demonstrating state-of-the-art performance in both planning safety and inference efficiency.

   Guidelines:

   - The answer [NA] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [NA] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitation in Conclusion.

   Guidelines:

   - The answer [NA] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.

- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: We provide the mathematics properties for NFs in Section 3, for full proof is in Appendix A and B.

   Guidelines:

   - The answer [NA] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Please refer to the released codes in supplemental material.

   Guidelines:

   - The answer [NA] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.

(b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We provide the code in supplemental material.

   Guidelines:

   - The answer [NA] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We detail them in Appendix D.

   Guidelines:

   - The answer [NA] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

Justification: This paper does not report error bars.

Guidelines:

- The answer [NA] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Please refer to Appendix D.

   Guidelines:

   - The answer [NA] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This paper conforms with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [NA] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No] , they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: There is no societal impact of the work performed.

Guidelines:

- The answer [NA] means that there is no societal impact of the work performed.
- If the authors answer [NA] or [No] , they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: This paper poses no such risks.

Guidelines:

- The answer [NA] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We cite the datasets we use in experiments and introduce them.

Guidelines:

- The answer [NA] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.

- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provide it in the supplemental material.

    Guidelines:

    - The answer [NA] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [NA] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [NA] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original or non-standard components.

    Guidelines:

    - The answer [NA] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.