GuideFlow: Constraint-Guided Flow Matching for Planning in End-to-End Autonomous Driving

Supplementary Material

A. Appendix

This supplementary material provides additional descriptions of the proposed GuideFlow framework, including the following supplementary material:

- **Sec. A.1:** Summary of Contributions.
- **Sec. A.2:** Proofs within the GuideFlow.
- **Sec. A.3:** The Details of Metrics.
- **Sec. A.4:** Implementation Details.
- **Sec. A.5:** Additional Metrics and Efficiency Analysis.
- **Sec. A.6:** More Ablation Studies.
- **Sec. A.7:** More Visualizations of Planning Results.

A.1. Summary of Contributions

Our contributions are summarized below.

1) **GuideFlow Framework.** We propose GuideFlow, an innovative framework based on constrained flow matching. This framework explicitly models the flow matching process and incorporates diverse conditional signals to guide trajectory generation, thereby effectively mitigating the "mode collapse" issue. By explicitly embedding safety constraints into the generation process, GuideFlow ensures strict compliance of output trajectories with safety requirements. This approach significantly enhances the stability and safety of motion planning in end-to-end autonomous driving systems.

2) **New* CVF, CF and RFE Modules.** We propose the Constraining the Velocity Field (CVF) module, which employs a predefined, constraint-adhering velocity field to actively correct the model's predicted velocity field, thereby steering the result to satisfy the constraints. The proposed Constraining the Flow States (CF) module enforces corrections on any deviating flow paths, thereby steering flow path toward the constraint-satisfying generation endpoint. Furthermore, we propose the Refining the Flow by EBM (RFE) module. By unifying flow matching architecture and EBM, we endow the model with the capacity for autonomous exploration within the data manifold, allowing it to "discover" constraint-satisfying results.

3) **New* Reward as Style Condition Module.** We propose the Reward as Style Condition Module. This module encodes the aggressiveness score, which evaluates trajectories in driving scenarios, into a conditional control signal. This signal enables GuideFlow to dynamically adjust the aggressiveness level of trajectories during the generation process, thereby producing driving trajectories with a broader range of behavioral styles. The details for calculating the aggressiveness score will be presented in Sec A.4.5.

A.2. Proofs within the GuideFlow

In this section, we prove the theoretical validity of the proposed RFE module.

A.2.1. Proofs for RFE Module

The starting point of RFE module is the JKO scheme [7]. The JKO scheme describes the discrete-time evolution of a probability distribution ρ_t along energy-minimizing trajectories in the Wasserstein space,

$$\rho_{t+\Delta t} = \arg\min_{\rho} \frac{W_2^2(\rho, \rho_t)}{2\Delta t} + \int V_\theta(x)\mathrm{d}\rho(x) + \varepsilon(t)\int \rho(x)\log\rho(x)\mathrm{d}x. \tag{1}$$

Here, θ denotes the learnable parameters of the scalar potential $V_\theta(x)$, and $\varepsilon(t)$ is a temperature-like parameter tuning the entropic term. The transport cost is given by the Wasserstein distance:

$$W_2^2(\rho, \rho_t) = \min_{\gamma\in\Gamma(\rho,\rho_t)} \int_{\mathbb{R}^d\times\mathbb{R}^d} \|x - x_t\|^2 \,\mathrm{d}\gamma(x, x_t), \tag{2}$$

where $\Gamma(\rho, \rho_t)$ is the set of couplings between ρ and ρ_t, i.e., the set of probability distributions on $\mathbb{R}^d \times \mathbb{R}^d$ with marginals ρ and ρ_t. Here, d is the dimensionality of the data. Following Energy Matching [1], the $\epsilon(t)$ is designed as a time-dependent linear schedule.

$$\varepsilon(t) = \begin{cases} 0, & 0 \le t < \tau^*, \\ \varepsilon_{\max}\frac{t-\tau^*}{1-\tau^*}, & \tau^* \le t \le 1, \\ \varepsilon_{\max}, & t \ge 1. \end{cases} \tag{3}$$

Then, we follow the approach in [1, 12] and study at each time t via its first-order optimality conditions:

$$\frac{(x_{t+\triangle t} - x_t)}{\triangle t} + \bigtriangledown_{x_t} v_\theta(x_t) + \varepsilon(t)\bigtriangledown_{x_t}\log(\phi_t(x_t)) = 0, \tag{4}$$

Near the target data manifold, the transport term disappears since $x_{t+\triangle t} = x_t$, so Eq. (4) reduces to:

$$\bigtriangledown_x v_\theta(x_t) + \varepsilon_{\max}\bigtriangledown_{x_t}\log(\phi_t(x_t)) = 0, \tag{5}$$

This implies that the terminal distribution follows a Boltzmann form:

$$\pi_1(x) \propto \exp(-\beta E_\theta(x)), \quad \beta = \epsilon_{\max}^{-1} > 0. \tag{6}$$

Thus, E_θ shapes the manifold into multiple low-energy basins, each corresponding to a distinct feasible mode (*e.g.,*

yield, merge). During sampling, the discretized update becomes:

$$x^{(k+1)} = x^{(k)} + v_\theta(x^{(k)}, t_k)\Delta t - \eta(t_k) \bigtriangledown_x E_\theta(x^{(k)}), \quad (7)$$

where $\eta(t)$ the discretized scheduler. In effect, the flow term efficiently transports samples towards the trajectory manifold for $0 < t < 1$, while for $t \geq \tau^*$, the energy term activates, guiding the samples into the distinct low-energy modes. This provides a principled foundation to ensure multi-modal diversity for our GuideFlow optimization.

A.3. The Details of Metrics

For **NuScenes and ADV-NuScenes** datasets, we adopt the Collision Rate as the primary performance metric, while excluding the L_2 Error. This selection is justified by the observation that multiple feasible driving trajectories often exist in real-world driving scenarios. Therefore, simply evaluating the similarity between the predicted trajectory and the expert trajectory is insufficient for accurately assessing the model's multimodal trajectory generation capability. Specifically, the Collision Rate is computed over a 3s prediction horizon. The trajectory is sampled at a time interval of 0.5s and the average value is reported as the final result.

For **Bench2Drive** [6] dataset, we adopt the Success Rate (SR) and Driving Score (DS) as the primary performance metrics. The Success Rate measures the proportion of successfully completed routes within the allotted time and without traffic violations. A route is deemed successful if the ego vehicle reaches its destination without any rule infractions. The success rate is calculated as the ratio of successful routes to the total number of routes. This metric follows CARLA [4] official metric as reference. It considers both route completion and penalty for infractions. Specifically, it averages the route completion percentages and penalizes infractions based on their severity as shown in . The driving score is normalized by the total number of routes from same type or group as well.

$$\text{DS} = \frac{1}{n_{\text{total}}} \sum_{i=1}^{n_{\text{total}}} \text{RC}_i * p_i, \quad (8)$$

where n_{total} denotes the number of successful routes and total samples respectively; RC_i represents the percentage of route distance completed for the i-th route; p_i means the infraction penalty on the i-th route.

For **NavSim** [3] dataset, we employ the proposed extended PDM score (EPDMS) [3], which is a weighted combination of several sub-scores: No at-fault Collisions (NC), Drivable Area Compliance (DAC), Driving Direction Compliance (DDC), Traffic Light Compliance (TLC), Ego Progress (EP), Time to Collision (TTC) within bound, Lane Keeping (LK), History Comfort (HC) and Extended Comfort (EC).

A.4. Implementation Details

A.4.1. Experimental Setup

For **NuScenes and ADV-NuScenes**, we load the first-stage pre-trained model of SparseDrive. This model trains the sparse perception module from scratch, which encompasses 3D object detection, multi-object tracking, and online mapping, to learn sparse scene representations. Then, we train the 3D object detection, online mapping, motion, and planning modules without freezing the weights of the sparse perception module of GuideFlow. For GuideFlow, we use ResNet50 [5] as backbone network and the input image size is 256×704. We trained GuideFlow for 8 epochs using a learning rate of 2×10^{-4} and a total batch size of 48, using 8 NVIDIA 4090 GPUs. The loss function for the supervised process during the training is defined as follows,

$$\mathbf{L} = \mathbf{L_D} + \mathbf{L_M} + \mathbf{L_{MP}} + \mathbf{L_{RF}} + \mathbf{L_{RFE}}, \quad (9)$$

Where, $\mathbf{L_{RF}}$ denotes the loss function for the Flow Matching process, and $\mathbf{L_{RFE}}$ represents the loss generated during the RFE Module process. During the inference phase, GuideFlow generates 18 trajectory proposals by performing $K = 100$ sampling steps. These proposals are then selected using the scorer embedded within SparseDrive.

For **Bench2Drive**, GuideFlow is built upon the Hydra-Next baseline architecture and employs ResNet-50 as the image backbone to extract front-view and back-view image features. We train GuideFlow on the training data for 20 epochs with a total batch size of 256 and a learning rate of 2×10^{-4}, using 8 NVIDIA V100 GPUs. And the loss function for the supervised process during the training is defined as follows,

$$\mathbf{L} = \mathbf{L_{HY}} + \mathbf{L_{RF}} + \mathbf{L_{RFE}}. \quad (10)$$

GuideFlow replaces the original trajectory generation module of Hydra-Next. Consequently, the loss term $\mathbf{L_{HY}}$ is defined as the summation of the losses from all other modules of the original Hydra-Next architecture, excluding the trajectory generation component. During the inference phase, GuideFlow generates 10 trajectory proposals by performing $K = 100$ sampling steps.

For **NavSim**, the TransFuser serves as our baseline. We adopt ResNet-34 as the image backbone to extract visual features from the front-view, left-front view, and right-front view camera inputs. GuideFlow was trained for 100 epochs using a total batch size of 64 and a learning rate of 2×10^{-4}, using 8 NVIDIA V100 GPUs. And the loss function for the supervised process during the training is defined as follows,

$$\mathbf{L} = \mathbf{L_D} + \mathbf{L_M} + \mathbf{L_{RF}} + \mathbf{L_{RFE}}. \quad (11)$$

During the inference phase, GuideFlow generates 100 trajectory proposals by performing $K = 100$ sampling steps.

GTRS-Dense [8] (with v2-99 backbone) serves as a scorer for trajectory selection. Furthermore, we dynamically adjusted the model's scoring rule during inference. More details regarding this adjustment are provided in Sec. A.4.4.

A.4.2. Constraint Satisfaction Evaluation Function $\jmath(\cdot)$

In this paper, we focus on two primary constraints: first, collision avoidance with other agents in the environment, and second, guaranteeing the ego vehicle stays within drivable area. We will detail how to calculate the constraint satisfaction for trajectories.

Collision Avoidance. Given a predicted trajectory $\tau \in \mathbb{R}^{T\times 2}$, we calculate the signed distance d_t between the ego-vehicle and surrounding agents at each timestep t. A positive value d_t^+ indicates no collision (i.e., safe separation), while a negative value d_t^- signifies that a collision has occurred. The satisfaction degree of the collision constraint for trajectory τ is calculated as follows:

$$\begin{aligned}\jmath_c(\tau) = &\frac{\sum_t^T 1_{d_t^+>0}\cdot f\left(\omega_{\mathrm{c}}\cdot\max\left(1-\frac{d_t^+}{r},0\right)\right)}{\omega_{\mathrm{c}}(\sum_t^T 1_{d_t^+>0}+\mathrm{eps})}\\ &\frac{\sum_t^T 1_{d_t^-<0}\cdot f\left(\omega_{\mathrm{c}}\cdot\max\left(1-\frac{d_t^-}{r},0\right)\right)}{\omega_{\mathrm{c}}(\sum_t^T 1_{d_t^-<0}+\mathrm{eps})},\end{aligned} \tag{12}$$

where $f(\cdot)=e^x-x$. r denotes the collision-sensitive distance and eps is added to ensure numerical stability.

Stay within Drivable Area. Given the road segmentation map, we convert the drivable area and other regions into a binary mask map. This mask is used to calculate the Signed Distance Field (SDF), where the drivable area is assigned negative values and regions outside the drivable area are assigned high positive values. We project the trajectory points onto the SDF and obtain the corresponding values d_t via nearest neighbor interpolation. Finally, the satisfaction degree of the drivable area constraint for the trajectory τ is calculated as follows:

$$\jmath_d(\tau)=\sum_t^T g\left(\omega_{\mathrm{d}}\cdot d_t\right), \tag{13}$$

where $g(\cdot)=e^x-x$.

A.4.3. Fusion Module F_θ in Detail

Plan anchor and target point encoding: For anchor $\in \mathbb{R}^{T\times 2}$, GuideFlow apply sinusoidal positional encoding to obtain $C_a \in \mathbb{R}^{T\times C}$, flatten it to $C_a \in \mathbb{R}^{TC}$, and then pass it through an MLP. The target point follows the same steps except flattening. Conditional fusion: During conditional fusion, the latent h_t first attends to the BEV feature B_t via cross-attention: $h_t \leftarrow \mathrm{CrossAttn}_\theta(h_t, B_t)$. Then, the concatenated driving guidance embeddings $(C_p \oplus C_g \oplus C_d)$ and aggressiveness score C_r are randomly masked ($p=0.2$) and added to h_t: $h_t \leftarrow h_t + \mathcal{M}(C_p \oplus C_g \oplus C_d) + \mathcal{M}(C_r)$.

A.4.4. Optimization of the Scoring Strategy for NavSim

During inference on the NavSim dataset, we observed that the GTRS-Dense trajectory scorer exhibited a strong bias towards selecting trajectories from the pre-clustered anchor trajectory table, rather than the real-time generated trajectories. This behavior is attributed to the fact that GTRS-Dense was trained solely on anchor scores (due to the difficulty of obtaining accurate ground-truth scores for online generated trajectories during training). Consequently, the model was only exposed to sub-optimal anchor trajectories rather than the optimal real-time solutions. To mitigate this bias, we enforce a higher selection rate for the generated trajectories during inference by manually boosting their scores by an additive factor of 0.1.

A.4.5. The Computation Method for EP Reward

To quantify the aggressiveness score of trajectories, GuideFlow introduces the EP Reward. This reward is designed to evaluate the ego vehicle's progress in advancing along the lane centerline within a specified time frame. Specifically, for each trajectory, we first determine the projection positions of its start point P_{s} and end point P_{e} onto the reference centerline. The raw progress value is defined as the difference between the arc length of the end point projection and that of the start point projection:

$$EP=\mathbf{MAX}(0,\mathrm{proj}(P_{\mathrm{e}})-\mathrm{proj}(P_{\mathrm{s}})), \tag{14}$$

where the projection function $\mathrm{proj}(\cdot)$ maps points to their closest points on the centerline. $\mathrm{proj}(\cdot)$ is implemented using the *shapely* Python library.

A.5. Additional Metrics and Efficiency Analysis

A.5.1. Additional Clarification

In the ablation with fewer inference steps, the hyperparameters of the CF and RFE modules (k_c and $\triangle t$) were not adjusted accordingly. We have now fixed this by clarifying the hyperparameter correspondence, and report the corrected results in Tab. 3.

A.5.2. Additional Trajectory Diversity and Temporal Consistency Evaluation

Multi-modal driving behavior is fundamentally reflected in trajectory diversity. As shown in Tab. 1, we quantify the diversity of GuideFlow using the **Div** metric, while assessing trajectory quality with the mean EPMDS of the top-K trajectories (EPMDS@K), following DIVER [11] and DiffusionDriveV2 [13]. GuideFlow significantly outperforms TransFuser and DiffusionDrive, demonstrating its ability to mitigate mode collapse while maintaining high-quality trajectory generation.

We further validate this advantage on NuScenes, where Tab. 2 confirms consistent improvements in Div. Notably,

the official NuScenes evaluation protocol reports only collision rate and L_2 error. To provide a more comprehensive assessment, we additionally evaluate Div (diversity) and TPC (temporal consistency), metrics adopted from DIVER [11] and MomAD [10]. GuideFlow achieves state-of-the-art performance on both Div and L_2, while also improving temporal consistency.

Table 1. Additional tests of Div and EPDMS@K on the NavHard split. "mmt" refers multi-modal trajectory variant of TransFuser.

Method	Div ↑	EPDMS@1↑	EPDMS@3↑	EPDMS@5↑
$\text{TransFuser}_{\text{mmt}}$	0.12	23.3	22.6	21.5
DiffusionDrive	0.20	24.2	23.8	22.2
GuideFlow	**0.24**	**27.1**	**26.7**	**26.0**

Table 2. Additional Diversity and Quality Results on NuScenes.

Method	*Div (avg)*↑	TPC (avg)↓	Col. Rate (avg)↓	L_2 (avg)↓
MomAD	0.11	0.54	0.08	0.60
SparseDrive	0.13	0.57	0.08	0.61
GuideFlow	**0.18**	0.55	0.07	0.57

A.5.3. Efficiency Analysis

In the main paper, $K = 100$ is used as the default setting. For completeness, we further provide ablation results with fewer inference steps in this appendix. The corresponding results are reported in Tab. 3. We observe that smaller K still maintains strong performance, demonstrating the robustness of GuideFlow under different inference budgets. And our method remains lightweight and comparable to TransFuser [2] and DiffusionDrive [9] in params and memory, making it suitable for deployment.

Table 3. Ablation study of inference steps and comparison of inference efficiency with other methods. bs denotes batch size.

SubTable 1				SubTable 2		
k_c	$\triangle t$	K	EPDMS	Method	Params	Mem (GB with bs=1)
20	0.02	50	26.9	Tranfuser	56.1M	0.209
10	0.04	25	26.9	DiffusionDrive	60.7M	0.226
4	0.10	10	26.6	GuideFlow	66.7M	0.248

A.6. More Ablation Studies

A.6.1. Combinations of CVF, CF, and RFE Modules

In this section, we present additional ablation studies about CVF, CF and RFE modules, as shown in Tab. 4. To systematically validate the effectiveness of the proposed approach, we conducted a comprehensive evaluation of different combinations of the CVF, CF, and RFE modules within the GuideFlow framework. Experimental results show that while the combination of CVF and CF improves performance compared to using CF alone, it still falls short of the performance achieved by CVF individually. This indicates that both CF and CVF perform corrections during the flow matching process, and simply combining them leads to "over-correction" issues, resulting in irregular probability flows and consequently suboptimal performance. The combination of RFE and CVF modules further confirms this finding. Since the RFE module performs optimization after the flow matching process, its integration with CVF effectively avoids multiple correction conflicts during the matching phase, thus achieving better performance than using either module alone.

Table 4. Ablation studies of different modules in GuideFlow over NavSim [3] HavHard Split. "EP" stands for Ego Progress subscore. "CVF" denotes "Constraining the Velocity Field" module, "CF" denotes "Constraining the Flow State", "RFE" denotes "Refining the Flow by EBM" and "RAS" denotes "Reward as Style Condition".

Modules				NavSim Hard (No Scorer)			
CVF	CF	RFE	RAS	EP	Stage1 EPDMS	Stage2 EPDMS	EPDMS
				84.1	56.7	40.0	23.1
✓				80.9	56.9	41.3	24.5
	✓			81.7	57.1	44.7	25.1
✓	✓			82.8	56.2	44.8	24.9
		✓		81.1	53.3	45.3	25.5
	✓	✓		79.6	54.9	**47.9**	**27.1**
✓		✓		80.0	56.5	45.4	26.2
	✓	✓	✓	82.3	**60.1**	43.7	26.3
✓	✓	✓	✓	82.5	60.0	43.7	26.1

A.6.2. Effect of the RAS module guidance strength

To further investigate the effectiveness of the RAS module, we systematically examined its impact on the model by configuring different "EP Reward" values during inference. As shown in Tab. 5, as the "EP Reward" increases from 0.1 to 1, the model's EP score progressively rises and eventually peaks, demonstrating the RAS module's capability to effectively guide trajectory style variations through reward signaling.

Furthermore, we observe a notable phenomenon: EPDMS reaches its optimum at "EP Reward" of 0.8, then decreases with further reward increase. This indicates that indiscriminately encouraging aggressive behavior is not suitable for all driving scenarios (e.g., congested road conditions). The optimal reward setting should be adaptively determined by the model rather than manually specified. Automating the configuration of reward will be a direction for our future work. Despite this observation, the ablation study robustly validate the core innovation of the RAS module: effectively guiding trajectory evolution toward desired

Table 5. Ablation studies on the effects of "EP Reward" parameter configuration within the RAS Module on model performance over NavSim HavHard Split.

Reward	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
EP	79.4	79.1	79.8	80.3	80.3	80.6	81.3	81.7	81.6	82.3
EPDMS	25.6	25.4	25.5	26.0	26.2	26.3	26.1	26.8	26.5	26.3

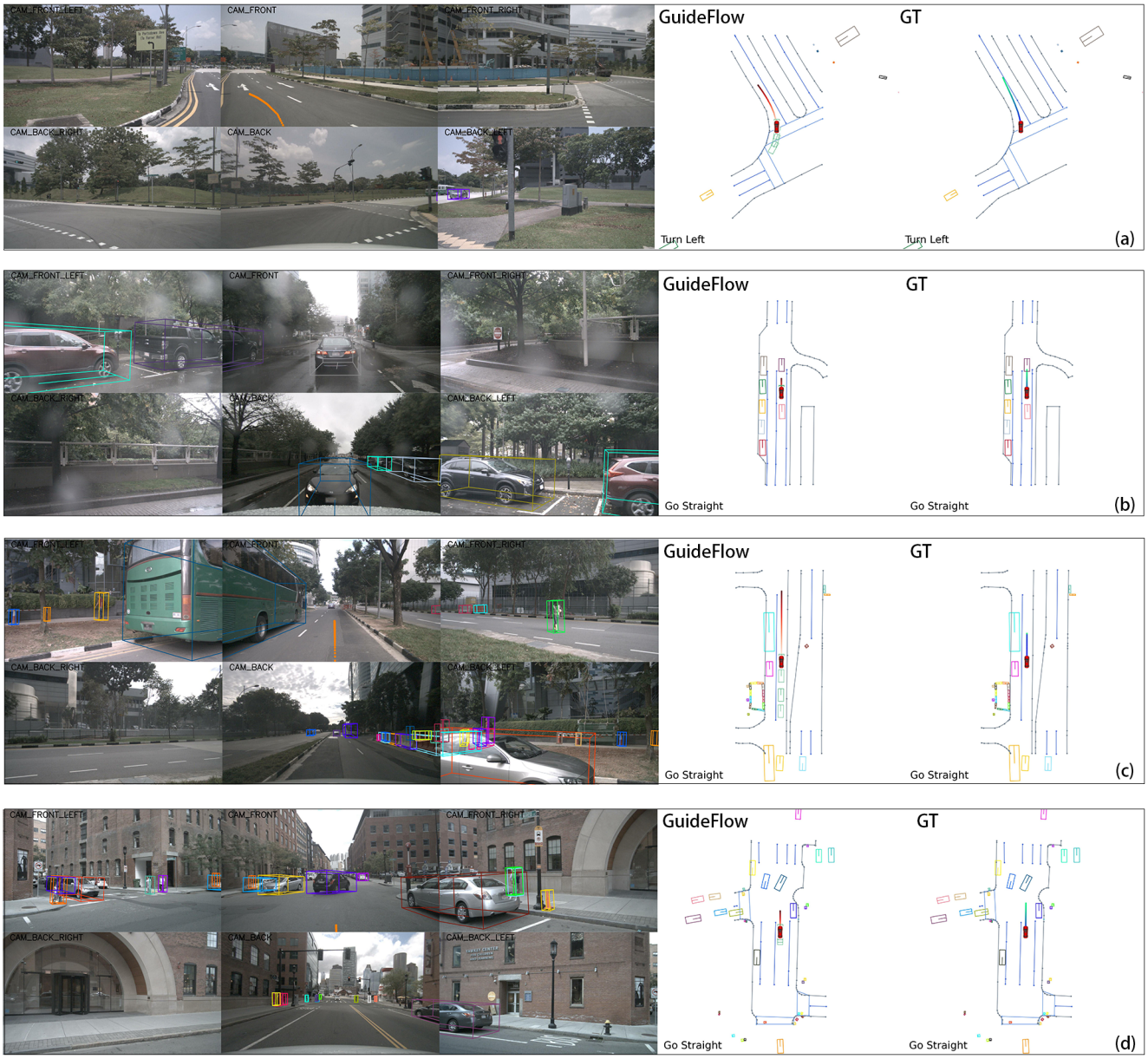

Figure 1. **Better than expert trajectory.** Visualization results in NuScenes dataset. (a) Turning scenario on a clear road. (b) Car following scenario, need maintain a safe distance from the leading vehicle to avoid collisions. (c) Straight road driving scenario, free of vehicles ahead. (d) Straight road driving scenario, a vehicle ahead is executing a turn, need the ego vehicle to give way.

directions through dynamic reward adjustment.

A.6.3. Sensitivity of Hyper-parameters in GuideFlow

Impact of λ. As the λ increases from 0.1 to 0.5, the collision rates of GuideFlow on both NuScenes and ADV-NuScenes datasets demonstrate a upward trend. The performance degradation stems not from the constraint strategy itself, but from excessive interference with the predicted velocity field, which compromises the smoothness of the flow and reduces trajectory quality as shown in Tab. 6.

Impact of k_c. As shown in Tab. 6, the observed continuous decline in collision rate as k increases from 10 to 50 confirms the effectiveness of the CF module. This trend indicates that applying constraints nearer to the endpoint of flow matching provides stronger and more direct guidance for trajectory generation, thereby resulting in a higher probability of satisfying the desired constraints.

Impact of K. As the number of sampling steps decreases, GuideFlow exhibits a consistent decline in performance. This phenomenon can be attributed to the fact that reduc-

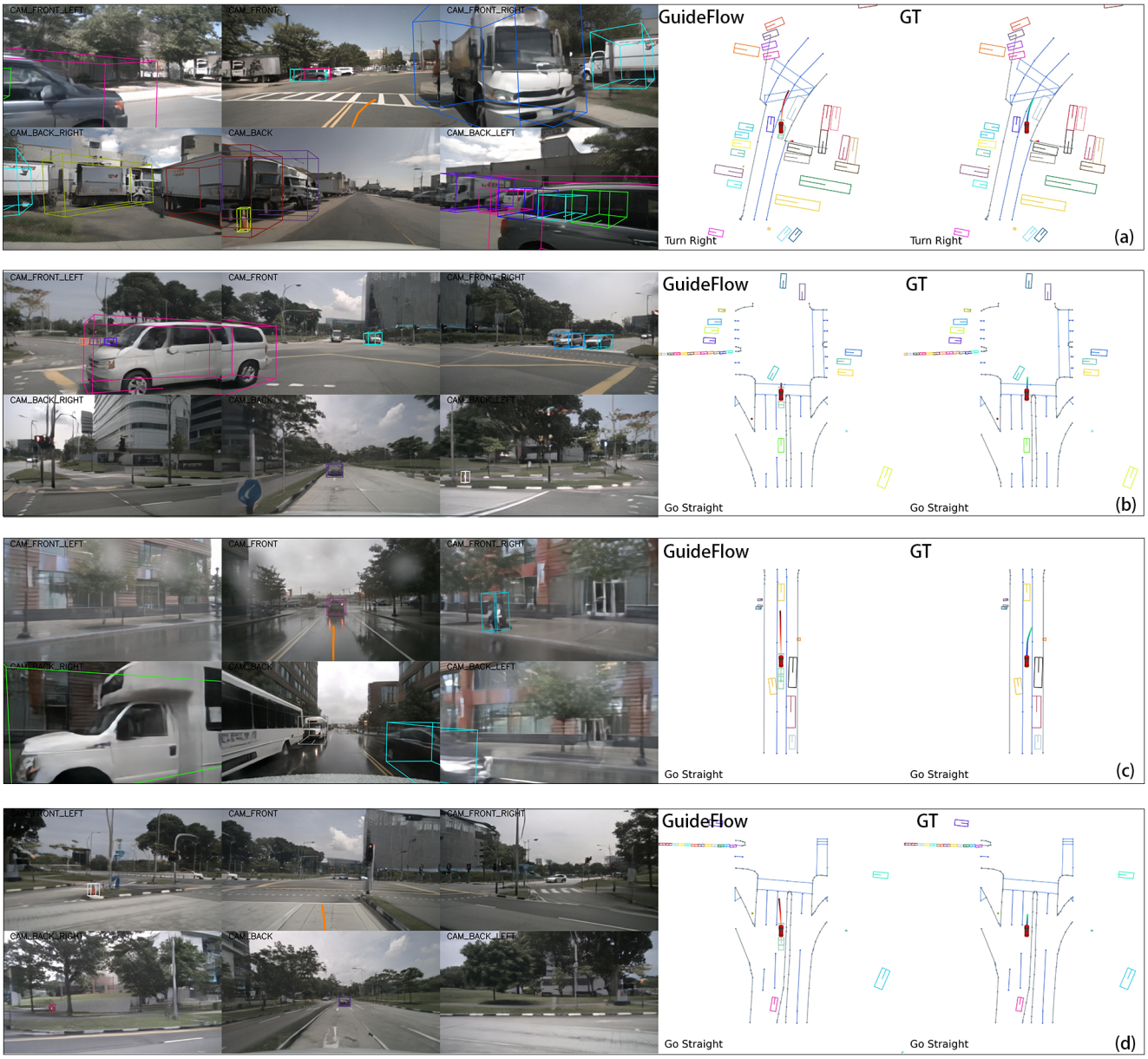

Figure 2. **Better than expert trajectory.** Visualization results in ADV-NuScenes dataset. (a) Emergency evasion scenario, a vehicle is encroaching into the lane, necessitating an immediate evasive maneuver by the ego vehicle. (b) Straight road driving scenario, a vehicle ahead is executing a turn, need the ego vehicle to give way. (c) Car following scenario, need maintain a safe distance from the leading vehicle to avoid collisions. (d) Straight road scenario, need ego vehicle avoid encroaching upon the non drivable area on its right.

ing the number of sampling steps corresponds to increasing the step size. This strategy only preserves sampling fidelity when the probability flow is a "straight" flow. However, since the learned probability flow in the GuideFlow is not "straight", enlarging the step size directly disrupts the sampling process, leading to performance degradation, as shown in Tab. 6.

A.7. More Visualizations of Planning Results

To better illustrate the exceptional planning capabilities of GuideFlow, we present visualizations of its planning outcomes across multiple complex traffic scenarios, including turning maneuvers, congested traffic, and vehicle avoidance situations. We provide four qualitative results: (1) Planning under scenarios in NuScenes, (2) Planning under adversarial scenarios in ADV-NuScenes, (3) Closed-loop planning

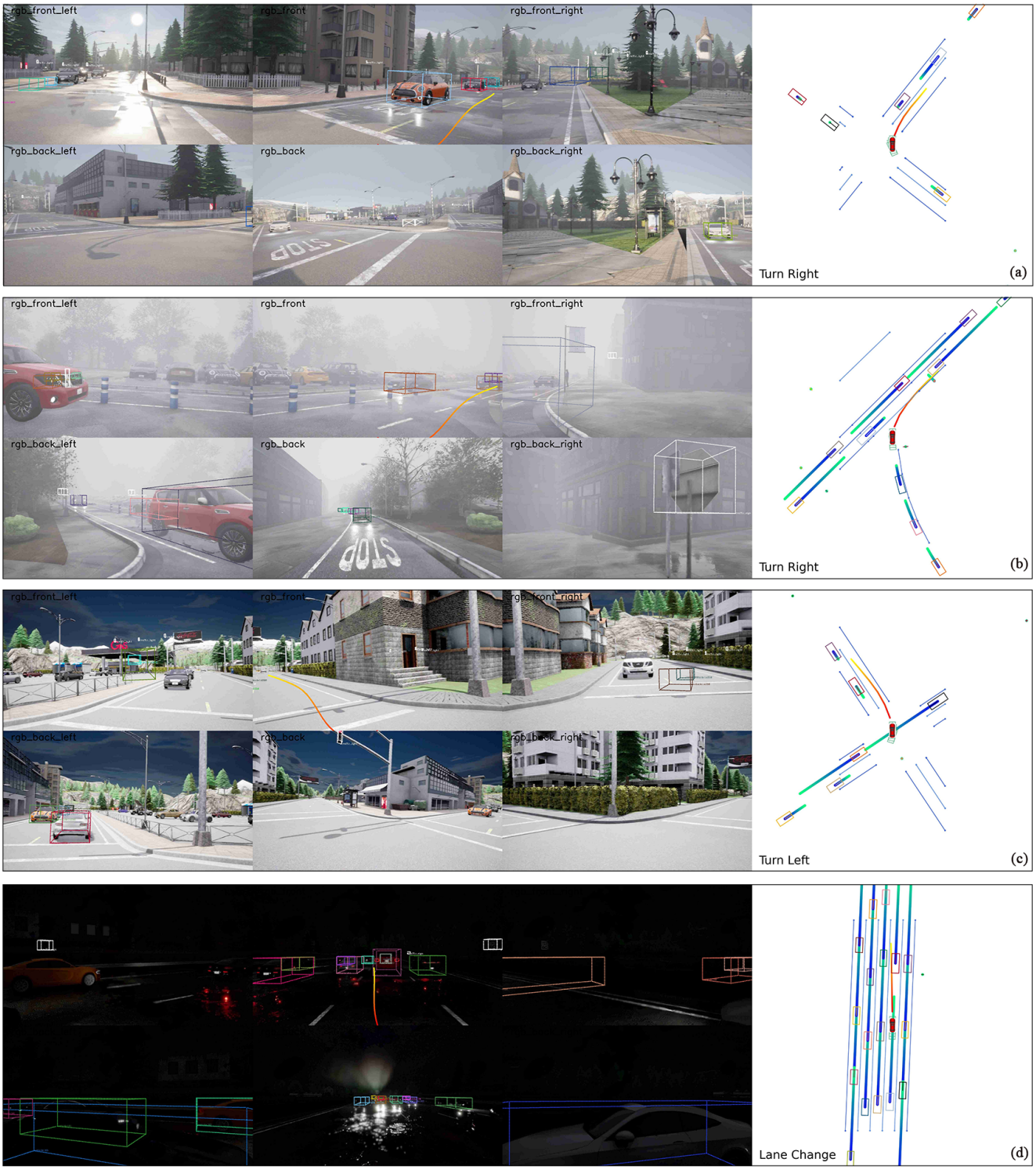

Figure 3. **Visualization results (Closed loop) in Bench2Drive dataset.** (a) Turning scenario on a clear road. (b) Turning scenario in congested traffic, need the ego vehicle avoid a collision. (c) Turning scenario in congested traffic, need the ego vehicle avoid a collision. (d) Lane change scenario in congested traffic, need the ego vehicle avoid a collision.

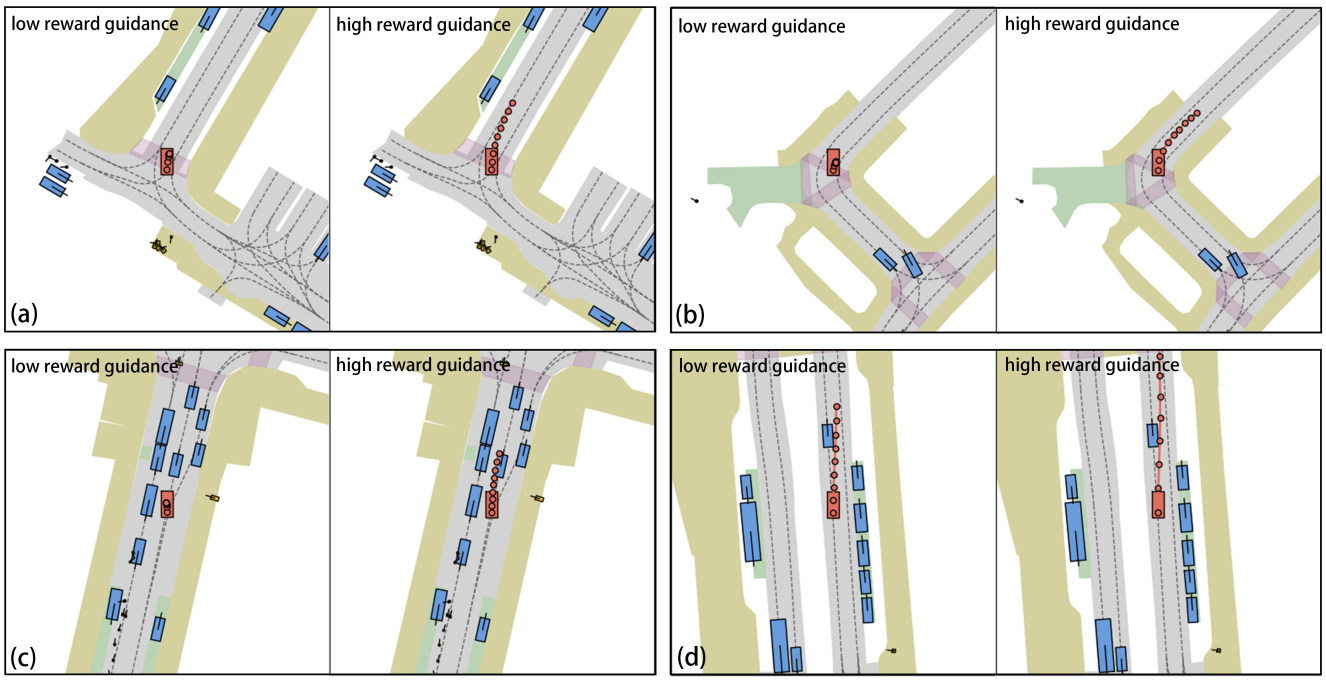

Figure 4. **Visualization of Trajectory Style Transitions on the NavSim Dataset.** The left side of each subfigure corresponds to trajectory generation guided by a low EP reward, while the right side corresponds to generation under a high EP reward.

Table 6. The hyper-parameter λ, k_c and K effects on GuideFlow's performance for the NuScenes and ADV-NuScenes Dataset. "C.R." denotes "Collison Rate".

λ	NuS C.R.	ADV-NuS C.R.	k_c	NuS C.R.	ADV-NuS C.R.	K	NuS C.R.	ADV-NuS C.R.
0.1	0.08	0.81	10	0.08	0.86	100	0.07	0.73
0.2	0.07	0.82	20	0.07	0.83	50	0.07	0.87
0.3	0.09	0.90	30	0.08	0.82	25	0.09	1.01
0.4	0.09	0.95	40	0.07	0.73	10	0.09	1.00
0.5	0.10	1.12	50	0.07	0.73	-	-	-

under scenarios in Bench2Drive and (4) Style transitions of planning guided by varying EP rewards. It is noteworthy that GuideFlow even surpasses the expert(GT) trajectories in certain scenarios of both the NuScenes and ADV-NuScenes datasets.

Planning under scenarios in NuScenes. As shown in Fig. 1, we demonstrate the planning capability of GuideFlow across multiple scenarios from the NuScenes dataset. In subfigure (b), compared to the GT trajectory, GuideFlow adopts a more conservative car following strategy to mitigate the risk of colliding with the preceding vehicle. This behavioral discrepancy highlights the advantage of GuideFlow's non imitation learning training framework and constrained generation capability. A similar tendency is observed in subfigure (d). It is worth noting that in subfigure (c), where no vehicle is present ahead, GuideFlow exhibits more aggressive maneuvering. It further illustrates GuideFlow's capacity to dynamically adapt to varying scene.

Planning under adversarial scenarios in ADV-NuScenes. Fig. 2 illustrates the performance of GuideFlow under various adversarial driving scenarios from the ADV-NuScenes dataset. In subfigure (a), when a vehicle illegally occupies the adjacent lane, GuideFlow demonstrates a timely avoidance strategy. In subfigure (b), compared to GT trajectory, GuideFlow exhibits a more proactive deceleration behavior to prevent a potential collision. Furthermore, in subfigure (d), to avoid encroaching into the non drivable area on the right, GuideFlow executes a pronounced evasion maneuver.

Closed-loop planning under scenarios in Bench2Drive. To evaluate the closed loop planning capability of GuideFlow, we visualized its planning performance in the CARLA simulator as shown in Fig. 3. The results demonstrate that GuideFlow consistently produces smooth and safe trajectories during various maneuvers, including turns and lane changes. Notably, it maintains reliable planning performance even under adverse visual conditions, such as foggy weather and nighttime.

Style transitions of planning guided by varying EP rewards. Fig. 4 illustrates the trajectory style transitions capability of GuideFlow. When guided by a low EP reward value, the system generates trajectories with conservative driving behavior. Conversely, under high EP reward guidance, it exhibits more aggressive maneuvering, such as overtaking strategies, as exemplified in subfigures (c) and (d) of Fig. 4.

References

[1] Michal Balcerak, Tamaz Amiranashvili, Antonio Terpin, Suprosanna Shit, Lea Bogensperger, Sebastian Kaltenbach, Petros Koumoutsakos, and Bjoern Menze. Energy matching: unifying flow matching and energy-based models for generative modeling. *arXiv preprint arXiv:2504.10612*, 2025. 1

[2] 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. 4

[3] 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. *Advances in Neural Information Processing Systems*, 37:28706–28719, 2024. 2, 4

[4] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017. 2

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

[6] Xiaosong Jia, Zhenjie Yang, Qifeng Li, Zhiyuan Zhang, and Junchi Yan. Bench2drive: Towards multi-ability benchmarking of closed-loop end-to-end autonomous driving. *Advances in Neural Information Processing Systems*, 37:819–844, 2024. 2

[7] Richard Jordan, David Kinderlehrer, and Felix Otto. The variational formulation of the fokker–planck equation. *SIAM journal on mathematical analysis*, 29(1):1–17, 1998. 1

[8] Zhenxin Li, Wenhao Yao, Zi Wang, Xinglong Sun, Joshua Chen, Nadine Chang, Maying Shen, Zuxuan Wu, Shiyi Lan, and Jose M Alvarez. Generalized trajectory scoring for end-to-end multimodal planning. *arXiv preprint arXiv:2506.06664*, 2025. 3

[9] 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 *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 12037–12047, 2025. 4

[10] Ziying Song, Caiyan Jia, Lin Liu, Hongyu Pan, Yongchang Zhang, Junming Wang, Xingyu Zhang, Shaoqing Xu, Lei Yang, and Yadan Luo. Don't shake the wheel: Momentum-aware planning in end-to-end autonomous driving. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 22432–22441, 2025. 4

[11] Ziying Song, Lin Liu, Hongyu Pan, Bencheng Liao, Mingzhe Guo, Lei Yang, Yongchang Zhang, Shaoqing Xu, Caiyan Jia, and Yadan Luo. Diver: Reinforced diffusion breaks imitation bottlenecks in end-to-end autonomous driving. *arXiv preprint arXiv:2507.04049*, 2025. 3, 4

[12] Antonio Terpin, Nicolas Lanzetti, Martín Gadea, and Florian Dörfler. Learning diffusion at lightspeed. *Advances in Neural Information Processing Systems*, 37:6797–6832, 2024. 1

[13] 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. 3