Supplementary Material: End-to-End Multi-Mode Autonomous Driving Distillation by Isomorphic Hetero-Source Planning Model

Rui Yu[1] Xianghang Zhang[2] Runkai Zhao[3] Huaicheng Yan[1†] Meng Wang[1]
[1] East China University of Science and Technology [2] SenseAuto Research [3] The University of Sydney
y80220166@mail.ecust.edu.cn zhangxianghang@senseauto.com rzha9419@uni.sydney.edu.au
hcyan@ecust.edu.cn mengwang@ecust.edu.cn [†] Corresponding author

A. Supplementary Methodology

In this section, we present additional content that is not covered in detail in the paper to further support the theoretical framework of the argumentative article.

A.1. IRL-based Teacher Model

Position Embedding. For the temporal feature $\boldsymbol{I_A}$, we apply temporal positional encoding $\boldsymbol{TE}$, indexed by its sequence length, as described by the following equation.

$$pos(i) = \begin{cases} \sin\left(\frac{pos}{10000^{\frac{2i}{D}}}\right), & \text{if } i \text{ is even,} \\ \cos\left(\frac{pos}{10000^{\frac{2i}{D}}}\right). & \text{if } i \text{ is odd.} \end{cases} \quad (1)$$

For the global position encoding $\boldsymbol{PE_{A/M}}$, the initial position $pos_i^{t_0}$ is mapped to a higher-dimensional space in the same manner as Eq. (1), by passing through a linear layer and activation function to provide a global position prior.

Map Representation. For the map's feature encoding, $\mathbf{c_m} \in \mathbb{R}^{N_L \times 2}$ denotes the location point at the indexed $N_P/2$ position in the subset $\boldsymbol{M} \in \mathbb{R}^{N_L \times N_P \times 2}$ of lane lines. The vectorized representation $\mathbf{v}_\mathrm{m}$ and deviation $\mathbf{d}_\mathrm{m}$ are calculated using the following formula, while the orientation θ_m is obtained by calculating the inverse tangent of v_m.

$$v_j^i, d_j^i = (\boldsymbol{M}_j^i - \boldsymbol{M}_j^{i-1}), (\boldsymbol{M}_j^i - \boldsymbol{c_m}), \quad (2)$$

where j denotes the j-th lane line, i denotes the i-th point, and $\boldsymbol{M}_j^i$ is a 2D coordinate in the lane line object.

Ego Motion Encoder. As described in the main paper, we obtain $\boldsymbol{I_E}$ and $\boldsymbol{PE_E}$ by vectorizing and extracting endpoints from cluster center trajectories. Vectorization, achieved by solving trajectory differences (similar to the operation of p_i^t in the Agent Encoder), enables us to abstract the motion behavior of clustered trajectories in motion space. It is then mapped to the implicit space via Eq. (1) and further encoded by a learnable network layer to obtain the implicit embedding $\boldsymbol{I_E}$. For $\boldsymbol{PE_E}$, we use its endpoints to represent the final target points across different modes and map them to a unified high-dimensional representation, similar to positional encoding.

Temporal Decoder. All decoders take as input the concatenated Ego and Agent features from the current frame, denoted as $\boldsymbol{I} \in \mathbb{R}^{(N_E+N_A)\times 1\times D}$, serving as the query. The corresponding position embedding is represented as $\boldsymbol{PE} \in \mathbb{R}^{(N_E+N_A)\times 1\times D}$, where N_E, N_A, and D denote the number of ego instances, agent instances, and feature dimensions, respectively. In this decoder, the sequence length is represented by the time course, and the spatial-temporal correlation of each agent over time is captured through the cross attention mechanism. For temporal features, its multi-frame information $\boldsymbol{I_T}, \boldsymbol{PE_T} \in \mathbb{R}^{(N_E+N_A)\times T\times D}$ for each agent has been obtained through memory bank.

$$\boldsymbol{I} = Attn(Q(\boldsymbol{I}+\boldsymbol{PE}), K(\boldsymbol{I_T}+\boldsymbol{PE_T}), V(\boldsymbol{I_T})). \quad (3)$$

Agent Decoder. In the Agent Decoder, we focus on the interaction between Ego and Agent to achieve cross-target feature querying. Therefore, we use the number of targets $(N_E + N_A)$ as the sequence length and enable feature interaction through the following self-attention mechanism.

$$\boldsymbol{I} = Attn(Q(\boldsymbol{I}+\boldsymbol{PE}), K(\boldsymbol{I}+\boldsymbol{PE}), V(\boldsymbol{I})). \quad (4)$$

Map Decoder. To better understand the guiding role of static scenes in the planning process, the map decoder is designed to interact with the Ego and Agent instances $\boldsymbol{I}$. It uses the Map instance $\boldsymbol{I_M}$ and the lane center position embedding $\boldsymbol{PE_M}$ as the key and value for the decoder.

$$\boldsymbol{I} = Attn(Q(\boldsymbol{I}+\boldsymbol{PE}), K(\boldsymbol{I_M}+\boldsymbol{PE_M}), V(\boldsymbol{I_M})). \quad (5)$$

Imitation Learning Loss. The planning model predicts the trajectory $\hat{p}_{traj}$ directly using the expert trajectory $\bar{p}_{traj}$ constraints to obtain L_{reg}. In our setup, the objective is to learn the displacement between frames, effectively avoiding the issue of excessive variance in the regression values:

$$L_{reg} = \sum_{i=1}^{T} \|\hat{p}_{traj} - \bar{p}_{traj}\|. \quad (6)$$

Similarly, the predicted self-vehicle status $\hat{s}$ are constrained using the expert driver's status $\bar{s}$ (N variables as acceleration, angular velocity, speed, etc.).

$$L_{status} = \sum_{i=1}^{N} \|\hat{s} - \bar{s}\|. \tag{7}$$

The classification loss L_{cls} leverages expert probability distribution supervision, enabling the model to capture richer information for multi-mode planning. In our setup, the clustered trajectory closest to the expert trajectory is assigned the highest confidence (0.8), while the remaining nearest neighbors use soft labels to promote diversity in model learning, enabling multi-mode planning. Here, the manually assigned expert probability distribution is denoted as $\bar{p}_{\text{cls}}$, while the predicted probability is $\hat{p}_{\text{cls}}$, enabling diverse planning representations. We then apply KL divergence [2] to effectively supervise the predictive distribution.

$$L_{cls} = \sum_{i=1}^{M_E} \bar{p}_{\text{cls}}(i) \log \left(\frac{\bar{p}_{\text{cls}}(i)}{\hat{p}_{\text{cls}}(i)} \right). \tag{8}$$

The final imitation learning error is shown below, with its loss weights following the base setting of SparseDrive [7], and no further ablation analysis provided.

$$L_{IL} = 2 \times L_{reg} + 3 \times L_{status} + 0.5 \times L_{cls}. \tag{9}$$

Reward Function. First, the state error e_{st} is defined similarly to Eq. (7) to measure the difference in state estimates between the predicted state and the expert trajectory. Similarly, the trajectory mean error e_{traj}^{mean} is calculated as in Eq. (6), while the trajectory start error and end point error e_{traj}^{start}, e_{traj}^{end} are calculated only at specific points to measure the closeness of the model's prediction to the expert.

Next, the speed predicted $\hat{s}_{vx}$ is monitored with thresholds to prevent obvious out-of-bounds behavior, as follows.

$$e_{speed} = \begin{cases} 1, & \text{if } 0 < \hat{s}_{vx} < 20, \\ 0. & \text{otherwise.} \end{cases} \tag{10}$$

Additionally, for state predictions $\hat{s}$ and planning trajectories $\hat{p}_{traj}$, we evaluate their consistency error to encourage the model to establish associations between learned states and planning as following:

$$e_{consist} = \hat{s}_{vx}^{t_1} \times \Delta_t - \hat{p}_{traj}^{t_1}. \tag{11}$$

Meanwhile, for the collision error $e_{collision}$, the predicted trajectory and the real target are used to compute a value of 0 for collision and 1 for safety.

Finally, the negative exponent is used as a reward value, as described in the main text, and reward-weighted summation is implemented using linearly weighted reinforcement learning to enhance the reward representation of the scene.

$$r_t = \sum_{i=1}^{N} \omega_i e^{-x_i}. \tag{12}$$

Unlike the experience replay strategy in DQN [6], our multi-batch setup enables a unified representation across multiple scenarios. The proposed Target Network Update in DQN addresses target value fluctuations during training using a delayed update technique. However, in our setup, the heterogeneous state values of the inputs to the main and target networks cause the delayed update policy to fail. Therefore, we use the actual decision behavior $\bar{a}$ for the individually supervised target network.

$$L_{Target} = \sum_{i=1}^{\bar{a}} \|Q_T(\bar{s}, \bar{a}_i) - \bar{a}_i\|. \tag{13}$$

A.2. Motion-Guided Student Model

Generative Encoder. Separate encoders are used to map expert trajectories and agent instances into Gaussian-distributed feature spaces. A multi-layer 1D convolution with ReLU activation is applied to each token to parameterize the agent's target distribution, enabling distribution-level interaction through fusion and supervision, as described in the main text.

A.3. Knowledge Distillation

Adapter. As described in the paper, one set of agent and map features in the teacher and student models originates from the vectorized representation of annotation results, while the other is extracted from image features using deformable attention [8] to capture key information. To address feature heterogeneity after the decoder, we employ an adapter ψ for alignment.

$$\psi = Linear(ReLU(Linear)). \tag{14}$$

Motion Property Distillation. The categorized kinematic attribute distillation loss was introduced in the main text; however, its performance was found to be limited in certain experiments. To address this, an additional regression-based attribute distillation loss was incorporated into the section. Specifically, the strategy is similar to Eq. (6) in that it uses the trajectory predictions p_{reg}^{tc} from the teacher model to supervise the student model trajectory p_{reg}^{st}.

$$L_{reg}^{KD} = \sum_{i=1}^{T} \|p_{reg}^{st} - p_{reg}^{tc}\|. \tag{15}$$

B. Supplementary Experiments

Here, we provide a complementary account of ablation experiments and visual analyses to illustrate the improvements in the model's performance.

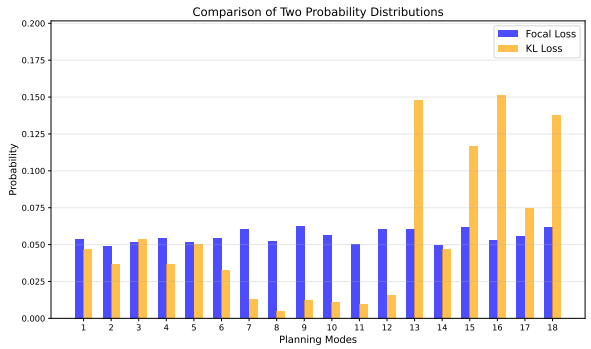

Figure 1. Visualization of probability for target in same scene.

Distillation	Generative	RL	EP ↑	PDMS ↑
✗	✗	✗	78.6	83.5
✓	✗	✗	80.0	85.8
✓	✓	✗	80.3	**86.0**
✓	✓	✓	**80.5**	85.5

Table 1. **Ablation study on the NAVSIM val dataset.**

Agent	Temporal	Map	Avg Collision(%)	Avg L2(m)	Best Epochs
✓	✗	✓	0.111	1.3915	60
✗	✓	✓	0.107	0.6632	60
✓	✓	✓	**0.066** -40%	**0.5716** -60%	30

Table 2. Ablation study of the planning decoder in the teacher model on nuScenes[1] val dataset.

KL Loss	Focal Loss	Avg Collision(%)	Avg L2(m)	Best Epochs
✓	✗	0.072	**0.5497**	30
✗	✓	**0.061**	0.5674	20

Table 3. Ablation study of the classification loss in the teacher model on nuScenes[1] val dataset.

B.1. Ablation Studies

Ablation Study in Closed-Loop Evaluation While the paper focused on ablation studies with nuScenes, we have since conducted further experiments on the NAVSIM dataset [4] to assess the effectiveness of each module. As shown in Tab. 1, distillation with the teacher model improves EP and PDMS, leading to more diverse and complete trajectories. The generative model enhances planning by interacting with the underlying motion distribution. However, since Transfuser is inherently trained without temporal modeling and relies on explicit ego status inputs, it conflicts with the implicit ego status representation and temporal learning in RL optimization, resulting in no significant performance improvement.

Selection of Planning Decoder. To validate the decoders design, we assess their performance in Tab. 2. As the evaluation mainly focuses on collision and planning, we analyze the Map Decoder and further compare the roles and interactions of the Temporal and Agent Decoders. Introducing the Agent Decoder enables basic interaction and planning, though with some performance fluctuations. Replacing it with the Temporal Decoder for sequential agents significantly enhances planning, especially by reducing L2 error. Using both decoders together enhances planning by balancing agent interactions and temporal information. This approach reduces collision rates and L2 error while improving training efficiency, requiring half the epochs.

L_{reg}^{KD}	L_{cls}^{KD}	L_{en}^{KD}	L_{de}^{KD}	Avg Collision(%)	Avg L2(m)
✓	✗	✗	✗	0.183	0.5901
✓	✓	✗	✗	0.108	0.5856
✓	✗	✓	✗	0.096	0.5918
✓	✗	✗	✓	**0.091**	**0.5842**

Table 4. Ablation Study of the regression distillation on the student model on nuScenes val dataset.

Effect of Classification Target. In SparseDrive [7], focal loss [5] is employed for supervised classification between dissimilar modalities. However, it relies solely on unique labels for supervision, lacking the capability for diverse imitation learning. As introduced in Eq. (8), we use KL divergence for supervision, so here we perform a simple performance analysis of the two categorical loss. As shown in Table Tab. 3, the two models exhibit comparable performance. However, the model with the KL Loss constraint demonstrates superior performance in terms of L2 metrics, while the model with the Focal Loss constraint achieves a lower collision rate. Distributions supervised by KL divergence exhibit more diverse probabilities in the scene, influenced by the number of TopKs, while label-supervised distributions have relatively uniform probabilities as show in Fig. 1. Therefore, we prefer the KL divergence-constrained classification layer for diversity.

Ablation Study of Regression Distillation. In the main text, we analyze classification distillation as less effective for practical motion planning due to its focus on category differences. Therefore, in Tab. 4, we additionally provide the effect of regression distillation, which can be found to have a negative effect on collisions when used alone, leading to regression head learning disorientation. While combining it with other distillation methods provides some mitigation, the overall performance remains inferior to multi-mode instance learning, which avoids the representation space gap between across heads.

Qualitative Visualization on NAVSIM Dataset. In Fig. 2, we present additional scenarios from the NAVSIM dataset to illustrate the results. In case (a), the Transfuser [3] struggles to complete a left turn, whereas DistillDrive successfully assists the model in navigating the turn according to the prescribed curvature. Similarly, DistillDrive effectively

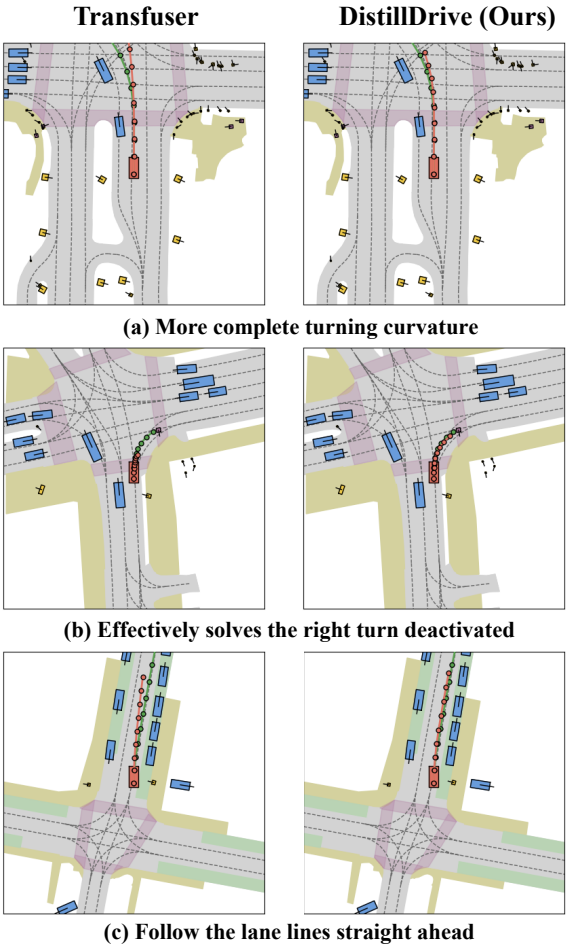

Figure 2. Qualitative visualization of the planning performance on the NAVSIM [4] navtest split,follow the logic of the paper.

Figure 3. **Qualitative visualisation on the CARLA dataset.** DistillDrive aligns better with driving demonstrations in turns, making it safer than TransFuser with lower collision rate.

addresses the right-turn deactivated issue of Transfuser, as shown in Fig. 2 (b). In the turn-to-straight scenario (c), Transfuser fails to efficiently adjust the yaw, whereas our proposed DistillDrive successfully outputs a planned path that aligns with the lane line as expected.

Qualitative Visualization on CARLA Dataset. To better evaluate the model's performance in closed-loop settings, we provide qualitative comparisons in Fig. 3, which show that DistillDrive enhances safety in turns and interactive scenarios through multi-mode distillation and reinforcement learning optimization.

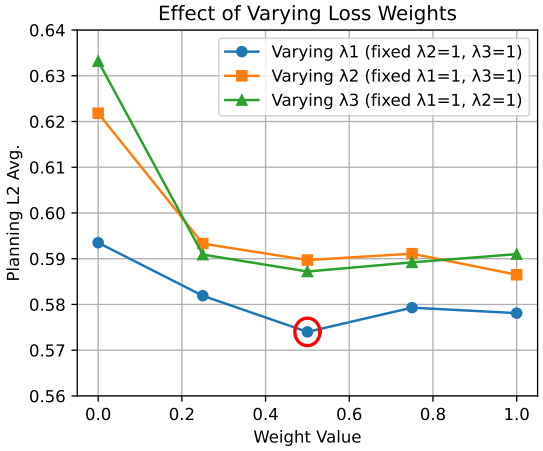

Figure 4. Visualization of loss weight impact on performance.

Ablation Study of Hyperparameters We have experimented with the hyperparameter ζ of the distributional generative model, and its impact on the model is not significant. As for the decaying sparse γ of the reward function in reinforcement learning, we did not do additional experiments and set it to 0.95 by default. The impact of the loss weight $\lambda_1, \lambda_2, \lambda_3$ on the model performance is analyzed in Fig. 4. The model achieves optimal performance when λ_2 and λ_3 are to 1, and λ_1 is set to 0.5. Conversely, performance deteriorates significantly when both reinforcement learning and generative models fail, which occurs when λ_2 and λ_3 are set to 0. At the same time, the choice of these two parameters (λ_2 and λ_3) has minimal impact on the model performance. Instead, the multi-mode instance imitation is primarily controlled by the distillation loss weight λ_1.

Effect Analysis of Knowledge Distillation. To effectively verify the reasonableness of knowledge distillation, we first visualize our teacher model and SparseDrive in Fig. 5. We observe that SparseDrive frequently encounters issues when turning, often leading to collisions with the road edge. In contrast, our designed teacher model effectively avoids such situations, further demonstrating its superior ability to perceive lane line information. Moreover, the designed diverse end-to-end imitation learning model not only captures the distributional representation of multi-modal motion features but also surpasses the teacher model in collision performance in certain cases, as shown in Fig. 6 (a, b).

Qualitative Visualization on nuScenes Dataset. In Fig. 7, we visualize DistillDrive's overall performance on the nuScenes dataset, covering both perception and planning.

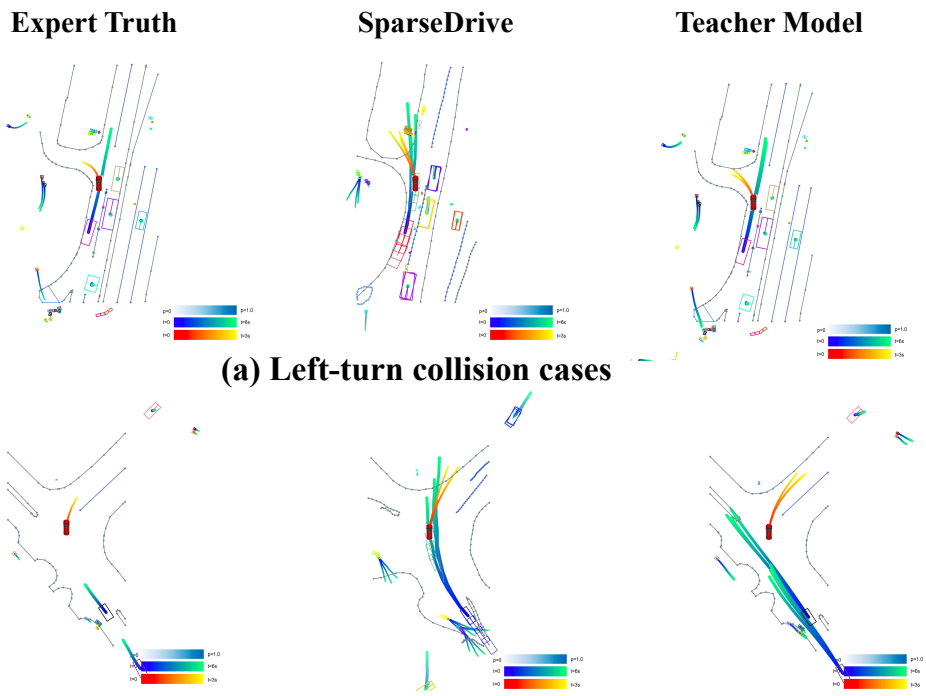

Figure 5. Qualitative visualization comparing the performance of the teacher model and the end-to-end model SparseDrive, verifying their differences in performance before knowledge distillation.

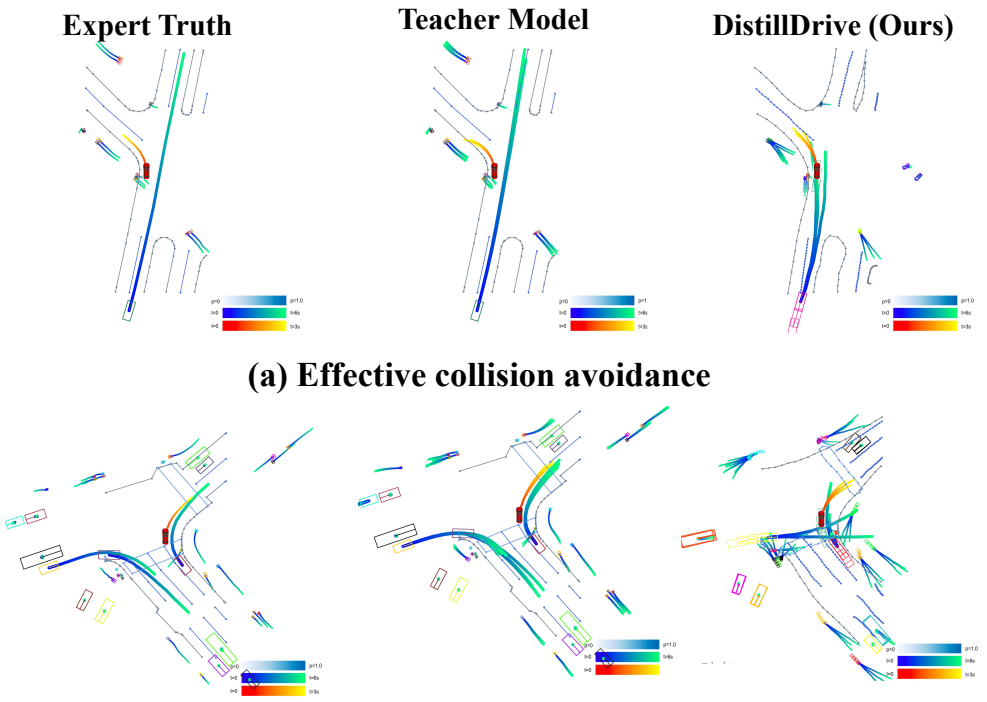

Figure 6. Qualitative visualization of performance for the teacher model and our proposed end-to-end model DistillDrive.

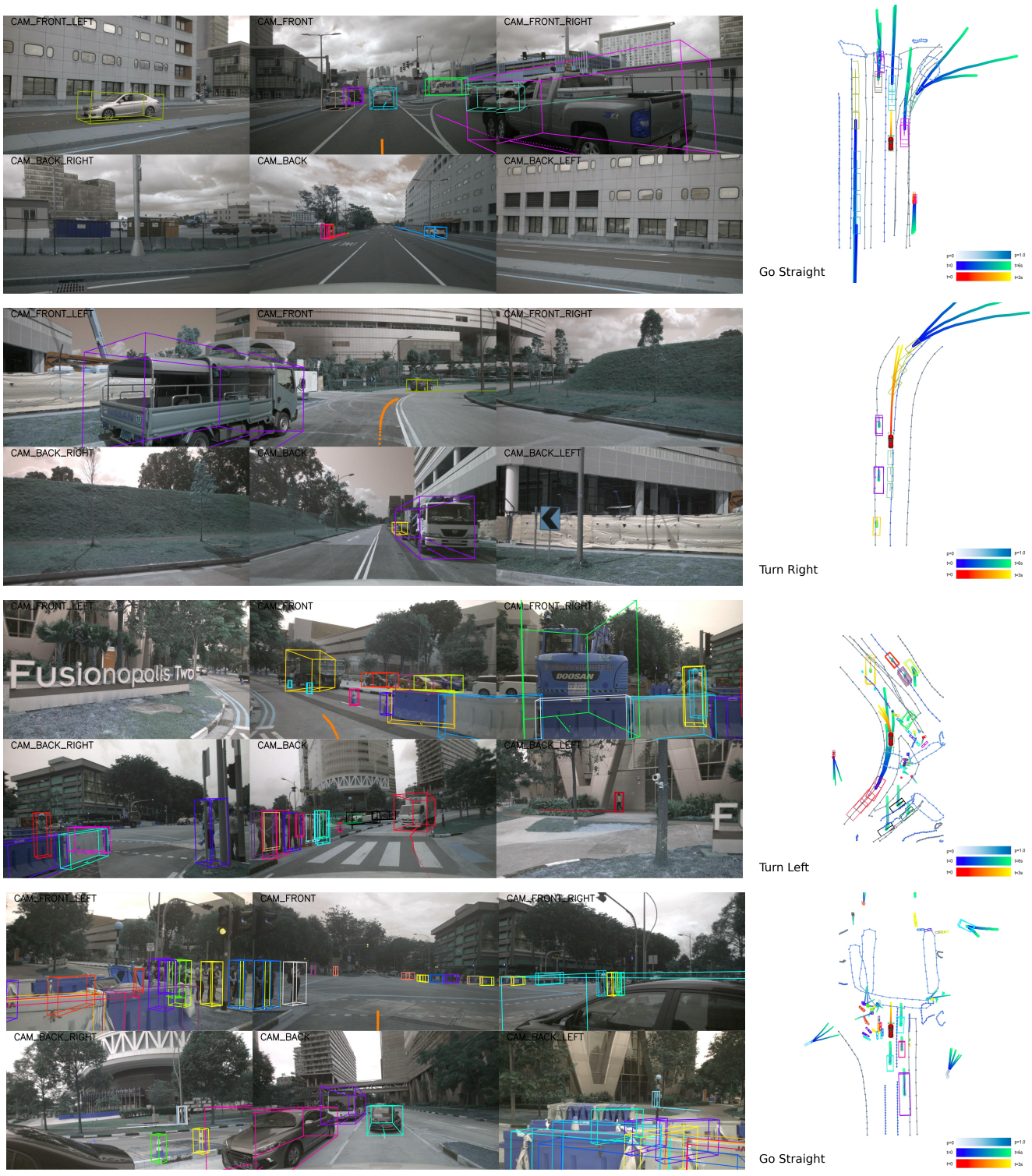

Figure 7. Qualitative visualization of DistillDrive's overall performance on the nuScenes val dataset.

References

[1] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern*

Recognition, pages 11621–11631, 2020. 3

[2] Shaoyu Chen, Bo Jiang, Hao Gao, Bencheng Liao, Qing Xu, Qian Zhang, Chang Huang, Wenyu Liu, and Xinggang Wang. Vadv2: End-to-end vectorized autonomous driving via probabilistic planning. *arXiv preprint arXiv:2402.13243*, 2024. 2

[3] 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. 3

[4] 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, 2025. 3, 4

[5] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE International Vonference on Vomputer BDision*, pages 2980–2988, 2017. 3

[6] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015. 2

[7] Wenchao Sun, Xuewu Lin, Yining Shi, Chuang Zhang, Haoran Wu, and Sifa Zheng. Sparsedrive: End-to-end autonomous driving via sparse scene representation. *arXiv preprint arXiv:2405.19620*, 2024. 2, 3

[8] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 2