BENCH2DRIVE-PLAN: BRIDGING PRIVILEGED PLANNER AND END-TO-END AUTONOMOUS DRIVING UNDER SHARED PROTOCOLS

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning-based autonomous driving has developed within two separate communities based on the input information. Privileged planners receive ground-truth object states and HD maps while end-to-end models use raw sensor inputs. Despite sharing many similar modeling ideas, differences in scene inputs, simulation,and metrics make it difficult to compare results and transfer insights across them. In this work, we introduce **Bench2Drive-Plan** to bridge this gap by evaluating privileged planners under current end-to-end protocols. On Bench2Drive, we adapt vectorized, diffusion-based, and language-based planning paradigms,including PLUTO, Diffusion Planner and InstructDriver. We also implement two raster-based CNN baselines using RGB semantic BEV images and multi-channel semantic masks, respectively.

Comparisons with end-to-end baselines results show that ground-truth scene inputs do not inherently yield a performance advantage for the privileged planners. This finding raises a question about the information needed for learning based driving. Ground-truth boxes and maps provide accurate scene states but may omit useful context from raw sensors. End-to-end models could utilize visual context, but their learned perception may be inaccurate. To examine whether planners could benefit from visual information beyond boxes and maps, we augment privileged planners with camera inputs. On Bench2Drive, extra camera inputs provide no overall benefit. While on NAVSIM, extra camera inputs yield a positive effect, which might indicate that under real world complex environment, scene details beyond boxes could be helpful. Together, these studies show how a shared benchmark can benefit the studies of the two communities and we hope Bench2Drive-Plan could pave the way for future studies about synergistically training planner and end-to-end models.

1 INTRODUCTION

Privileged planning and end-to-end driving have largely developed as separate research communities. The privileged planning community (Cheng et al., 2024; Zheng et al., 2025) studies how to learn driving trajectories from structured scene inputs, typically using ground-truth agent states and HD maps as inputs. The end-to-end driving community learns driving policies from sensor observations. Despite addressing similar problems and sharing modeling ideas, their methods are usually developed and evaluated on different datasets under different datasets and evaluation protocols .Privileged planners are commonly evaluated on nuPlan (Caesar et al., 2021), whereas end-to-end driving methods are more often evaluated on Bench2Drive and NAVSIM(Dauner et al., 2024; Jia et al., 2024). This separation makes comparison between the two paradigms difficult.

To bridge end-to-end driving and privileged planning,we introduce **Bench2Drive-Plan**, a benchmark to evaluate privileged planning models under the same scenarios, simulation and metrics as end-to-end methods. We first extract scene information covering the inputs required by all current privileged planners. Then we construct planner adapters that convert extracted scene information into the input format of each planner. The predicted trajectories are executed by the official Bench2Drive controller and scored using the benchmark metrics (Jia et al., 2024).

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

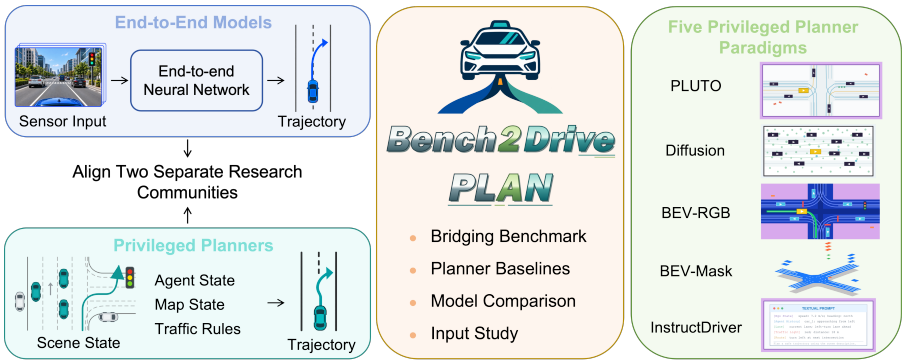

Figure 1: **Overview of Bench2Drive-Plan.** Our work bridges end-to-end driving and privileged planning, evaluates five privileged planners on Bench2Drive.

Bench2Drive-Plan includes baselines that adopt several architecture and scene representations. We adapt PLUTO (Cheng et al., 2024), Diffusion Planner (Zheng et al., 2025), and Instruct-Driver (Zhang et al., 2024) to cover vectorized, diffusion-based, and language-based planning. Two raster-based CNN baselines are also constructed, BEV-RGB and BEV-Mask, which encode scene states as RGB semantic BEV images and multi-channel semantic masks (Bansal et al., 2018). Comparing their results with end-to-end baselines brings these methods into a shared benchmark setting.

Our comparison shows that privileged planning methods do not exhibit a clear advantage over end-to-end baselines. One possible explanation is that, although ground-truth boxes and HD maps provide precise and reliable scene information, they can not provide the diverse visual cues that available in raw sensor observations (Li et al., 2025a). Motivated by this hypothesis, we augment privileged planners with front-view camera observations on Bench2Drive-Plan to investigate whether additional visual information can improve planning performance. However, we do not observe the expected improvement under our protocol. We hypothesize that this may result from the limitations in the scene complexity of the simulated environment. (Pasios & Nikolaidis, 2025; Park et al., 2023; Hu et al., 2023a) Therefore, the study is extended to NAVSIM(Dauner et al., 2024) and real camera inputs yield a positive effect. Together, these results support our hypothesis that in complex real-world driving environments, the visual details may be helpful for our planners. Importantly, this result does not diminish the role of Bench2Drive as a closed-loop driving benchmark,as recent studies suggest that they may emphasize different aspects of driving performance, with memory alone being sufficient for strong performance on NAVSIM without requiring planners to react to the current traffic scene (Löwens et al., 2026). We hope Bench2Drive-Plan can provide a common basis for future studies across different planner families, scene representations, and multimodal fusion strategies.

We summarize our contributions as follows:

- **Bridging benchmark:** We introduce Bench2Drive-Plan to connect privileged planning and end-to-end driving communities through shared evaluation protocols. The framework adapts scene inputs and planning outputs to bring privileged planners into the Bench2Drive setting, which enables methods previously studied on separate benchmarks to be assessed on the same protocol.
- **Planner baselines:** We adapt PLUTO, Diffusion Planner, and InstructDriver, covering established approaches in the planner community. We additionally construct BEV-RGB and BEV-Mask, two raster-based CNN baselines that share a backbone and training objective but encode scene information differently. These implementations broaden the methods evaluated on Bench2Drive and support the study of different architectures and scene representations.
- **Model comparison:** We systematically evaluate these planners on Bench2Drive and compare their scores with selected published end-to-end results. The leading planners using ground-truth scene inputs do not show a large performance margin over end-to-end models, while substantial differences remain among the evaluated scene-state planners. These findings show how methods from the two communities perform within the same benchmark setting and motivate closer study of how planners use the information available to them.

- **Input study:** We investigate whether privileged planners can benefit from additional visual information beyond ground-truth scene states by introducing front-view camera inputs. The results reveal different effects across simulated and real-world driving benchmarks, with no overall improvement on Bench2Drive and generally positive effects on NAVSIM. This study provides a diagnostic perspective on the role of visual information in privileged planning.

2 Related Work

2.1 End-to-End Driving and Privileged Planning

End-to-end autonomous driving maps onboard sensor observations to future trajectories or control commands (Chen et al., 2024; Chib & Singh, 2023). Early studies focused on image-based control and conditional imitation learning (Bojarski et al., 2016; Codevilla et al., 2018; Kendall et al., 2019). Subsequent methods introduced privileged distillation, intermediate representations, and multimodal fusion to improve learning and closed-loop driving (Chen et al., 2020; Prakash et al., 2021; Renz et al., 2022; Chen & Krähenbühl, 2022). Recent systems commonly transform camera or LiDAR features into object-centric, BEV, or occupancy representations and jointly perform perception, prediction, and planning (Shao et al., 2023; Chitta et al., 2022; Li et al., 2022; Wu et al., 2022; Hu et al., 2023b; Jiang et al., 2023; Sun et al., 2025; Zhu et al., 2025; Tong et al., 2023).

Focusing on the planning stage, existing methods employ diverse scene representations and trajectory generation strategies. Rasterized methods encode maps, actors, traffic signals, and routes as semantic BEV inputs (Bansal et al., 2018; Phan-Minh et al., 2020; Casas et al., 2021), whereas vectorized methods represent lane geometry and actor histories as tokens processed by transformers or graph networks (Gao et al., 2020; Huang et al., 2023; Zhang et al., 2025a). PLUTO further combines vectorized scene features with candidate reference lines and robust imitation learning (Cheng et al., 2024). Diffusion-based planners generate trajectories through conditional denoising and have been applied to decision making, control, and autonomous driving planning (Winter et al., 2025; Zhu et al., 2023; Zou et al., 2025; Janner et al., 2022; Chi et al., 2025; Zheng et al., 2025; Liao et al., 2025). Large language models and vision-language models further formulate driving as a reasoning problem (Li et al., 2024; Zhou et al., 2024; Jiang et al., 2025; Tian et al., 2024; Jiang et al., 2026). Representative methods study coordinate-based trajectory generation, instruction tuning, language-conditioned driving, visual reasoning, and closed-loop control (Mao et al., 2023; Zhang et al., 2024; Shao et al., 2024; Xu et al., 2024; Sima et al., 2024; Jia et al., 2026).

2.2 Driving Datasets and Simulation Benchmarks

Autonomous driving benchmarks have evolved from perception datasets such as KITTI (Geiger et al., 2012) to large scale datasets that support perception and prediction, including nuScenes (Caesar et al., 2020), Argoverse (Chang et al., 2019; Wilson et al., 2023), and the Waymo Open datasets (Sun et al., 2020; Ettinger et al., 2021). However, evaluation on recorded data alone does not fully characterize driving behavior during execution (Chen et al., 2024; Li et al., 2026). Existing planning benchmarks provide different scene interfaces and evaluation protocols. nuPlan supplies structured scene states and simulation tools commonly used to evaluate privileged planners, while NAVSIM evaluates predicted trajectories over logged driving scenes using the PDM Score (Caesar et al., 2021; Dauner et al., 2024). CARLA provides interactive closed-loop simulation with configurable sensor observations (Dosovitskiy et al., 2017). Built on CARLA, Bench2Drive provides official training data and 44 scenario types covering diverse driving capabilities, with recent extensions addressing robustness and speed-conditioned evaluation (Jia et al., 2024; Zhang et al., 2026; Shao et al., 2026). Bench2Drive-Plan bridges these evaluation ecosystems by benchmarking privileged planners and sensor-based end-to-end methods under the same Bench2Drive protocol.

3 Bench2Drive-Plan

3.1 Task Definition

In this task, we define a unified planner interface over shared scene semantics. Given a scene z and route specification r, the adapter $\mathcal{A}_m$ for method m materializes its native representation, and planner π_m predicts a future ego trajectory:

$$\hat{\tau}^{(m)} = \pi_m(\mathcal{A}_m(z, r)) \,. \tag{1}$$

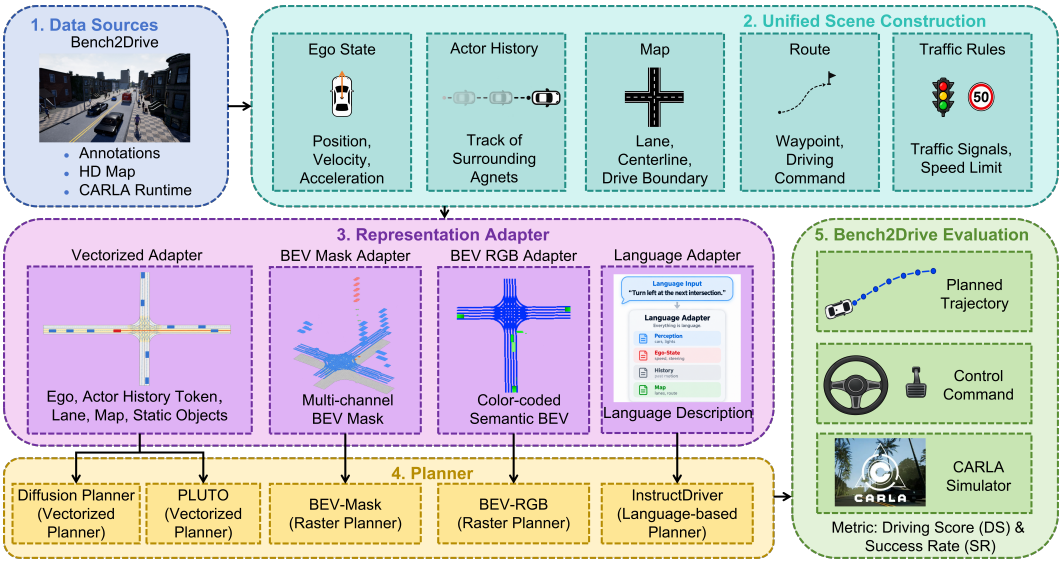

Figure 2: **Pipeline of the Bench2Drive-Plan.** Ground-truth scene states are converted into each planner's input through shared scene construction and representation-specific adapters, and the predicted ego trajectory is executed and scored under the official Bench2Drive protocol.

This interface aligns scene and output contracts while preserving their native input modality. Privileged planners operate on provided ground-truth scene states, including ego kinematics, dynamic actors, map geometry and topology, and traffic rules, whereas end-to-end methods infer the corresponding geometric and semantic information directly from raw sensor observations.

Under this unified framework, we train and evaluate five planners on Bench2Drive: PLUTO, Diffusion Planner, BEV-RGB, BEV-Mask and InstructDriver. Together, they span vectorized, diffusion-based, raster-based and language-based paradigms, enabling systematic comparison across heterogeneous scene representations and model architectures.

3.2 Planner-Oriented Scene Construction

We first extract the ground-truth scene information, including the ego vehicle and its history, surrounding agents and their histories, static objects, lanes, crosswalks, traffic-light states, and route information. Together, they cover the all scene information needed by privileged planners. We then use planner adapters to convert the extracted information into the input format required by each planner. Section 3.3 describes these adapters. For training, we also extract agents future coordinates as supervision, then convert to the output format of each planner. Below, we explain how to extract the scene information.

Ego, actors, and static elements. Dynamic actors, including the ego vehicle, are represented by their position, heading, velocity, acceleration, extent, and category, whereas static objects omit velocity and acceleration. To cover the diverse Bench2Drive scenarios, we retain additional actor categories, including emergency vehicles and vehicles with an open door. We obtain this information from benchmark-provided ground-truth scene states and transform the geometric features into the current ego coordinate frame. To construct actor histories, we associate observations across frames using persistent actor identifiers. Missing observations are padded and marked with validity masks.

Map vectorization. We extract lane centerlines, boundaries, crosswalks, and lane topology from each CARLA HD map. At each frame, we query map elements in an ego-centered region extending $120\,\mathrm{m}$ in each direction and transform their geometry from the CARLA world frame to a right-handed ego frame. Retained map polyline is uniformly resampled into 20 points. Lane and crosswalk polygons are indexed by STRtrees for efficient local queries (Leutenegger et al., 1997).

Most of the planners in our study were originally developed on nuPlan. However, Bench2Drive contains a substantially larger proportion of long lanes: 22.3% of Bench2Drive lanes exceed $100\,\mathrm{m}$, compared with 5.0% in nuPlan. With a fixed number of sample points, sampling a long CARLA lane

leaves too few points near the ego vehicle, making it harder for the planner to learn a useful local route representation. We therefore crop each nearby lane around the ego vehicle before sampling. We first crop the centerline, then use the same relative range to crop both boundaries. Finally, we sample corresponding points from the cropped centerline and boundaries. Further details are provided in Appendix B.1.

Route and traffic rules. Route information specifies the sequence of lanes traversed by the ego vehicle. During training, we obtain the route directly from the annotations, whereas during evaluation, we construct a feasible lane-level path from the start to the goal using graph-based search (Tarjan, 1972). We further augment each lane with traffic-rule information. Specifically, the traffic-light state is determined from the corresponding trigger region. Since CARLA provides speed limits only at sparse map locations, we index these samples with a cKDTree, query those near each lane centerline, and assign their median value to the lane (Virtanen et al., 2020).

3.3 Planner Adapters

To adapt the different input format of privileged planners, we construct corresponding planner adapters.

Vectorized planners. The vectorized adapter converts the extracted scene information into structured vectors for Diffusion Planner and PLUTO. For Diffusion Planner, the corresponding encoders transform these vectors into scene tokens that condition the trajectory denoising process (Zheng et al., 2025). For PLUTO, the adapter organizes actor histories and local map elements into agent and polygon features. It matches the route to the lane graph and traverses connected route branches to construct reference lines. The agent and map encoders provide scene context, while the encoded reference lines define the trajectory queries for PLUTO's planning decoder (Cheng et al., 2024).

Raster planners. Unlike camera-based BEV construction that lifts multi-view images into a BEV (Philion & Fidler, 2020; Huang et al., 2021; Li et al., 2023), our raster baselines directly rasterize the ground-truth scene state into a semantic BEV. The BEV-RGB adapter renders the extracted scene information as a semantic RGB image. Colors distinguish scene elements, while brightness represents actor history. The BEV-Mask adapter instead separates static map structure, actor categories, traffic rules, and sampled history into 38 binary channels.

Language-based planners. The language adapter converts the scene state into an ego-centric textual prompt (Zhang et al., 2024), providing a structured description of the relevant scene and traffic context. The LLM is prompted to generate a reasoning trace and ego-relative future poses represented as $(x, y, \text{heading})$. The parser converts these poses to the Bench2Drive trajectory interface.

3.4 Planner Evaluation Protocol

Each method is deployed as a CARLA Leaderboard agent under the same route partitions and simulator settings. At each simulation tick, current and historical CARLA states, route and map records, and traffic-rule states pass through the scene-construction pipeline and the planner adapter described above to form the model input. The planner then performs inference through its benchmark agent wrapper. All planners produce trajectories that are executed by the official Bench2Drive trajectory-tracking controller, which produces throttle, brake, and steering commands.

3.5 Camera Information Study

We study whether adding front-view camera input improves privileged planners on both Bench2Drive and NAVSIM. The front-view images come from simulation in Bench2Drive and real-world driving logs in NAVSIM. Camera geometry and image preprocessing are kept consistent between training and evaluation within each benchmark.

Camera input is integrated into each planner in its own representation. Every camera-augmented planner first encodes the front-view image I_{front} with an ImageNet-pretrained ResNet-34 and maps the pooled visual feature through a planner-specific learnable projection to a compact camera representation $\mathbf{c}_m$ in that planner's feature space. Due to differences in model architectures, planners then consume $\mathbf{c}_m$ differently. PLUTO and Diffusion Planner append it as an extra token to their scene-token sequence before the fusion Transformer; BEV-RGB and BEV-Mask concatenate it with the pooled BEV feature and pass the result through a fusion MLP; InstructDriver prepends it to the text token sequence and feeds them into the language model. Full details are given in Appendix D..

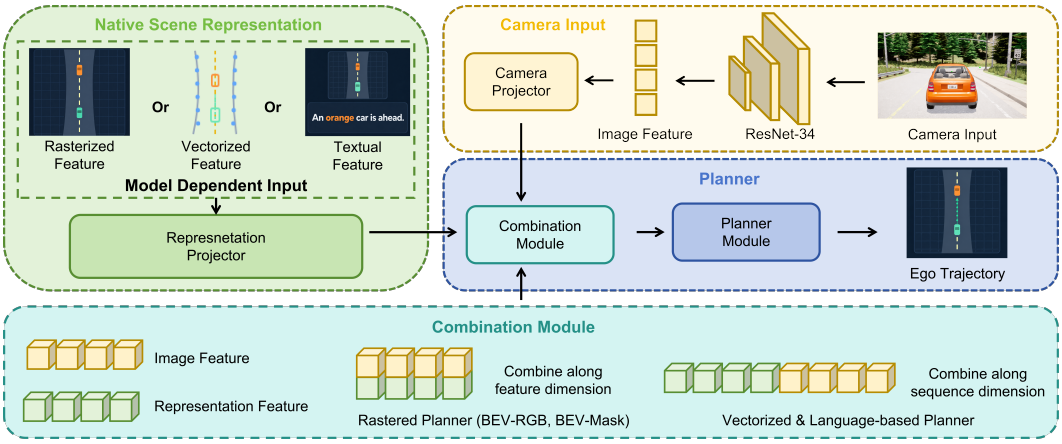

Figure 3: **Camera fusion.** A front-view camera feature is combined with each planner's original scene representation.

Before conducting the camera study on NAVSIM, we first adapt the evaluated privileged planners to the NAVSIM evaluation pipeline. We construct agents, map, route, and traffic rules features from the logged annotations and the map API, convert these features into the input format required by each planner, and map its output to NAVSIM's four-second trajectory representation for evaluation with the official PDM scoring pipeline. More details of the privileged planner adaptations to NAVSIM and the camera augmentation experiments are provided in Appendices C and D, respectively.

4 Experiments

4.1 Baselines

PLUTO (Cheng et al., 2024) encodes agent histories, static obstacles, and vectorized map elements into a scene representation, then combines lane-derived lateral queries with learnable longitudinal queries to decode and score multimodal ego-trajectory proposals.

Diffusion Planner (Zheng et al., 2025) generates trajectories through an iterative denoising process conditioned on a vectorized representation of the scene information.

BEV-RGB and BEV-Mask are constructed as two raster planners. Both use a ResNet-50 (He et al., 2016) pretrained on ImageNet (Deng et al., 2009), followed by global average pooling and a linear trajectory head that directly regresses future ego poses. BEV-RGB represents the scene as a RGB image, using different colors to distinguish scene elements. BEV-Mask represents the scene information in 38 binary channels that separate semantic categories and sampled history steps. They are trained with the same imitation objective.

InstructDriver (Zhang et al., 2024) represents structured scene information as text and uses a language model trained on driving instructions to predict future ego waypoints.

End-to-end Baselines. We use VAD (Jiang et al., 2023), UniAD-Base (Hu et al., 2023b), TCP-traj (Wu et al., 2022),BridgeAD (Zhang et al., 2025b), DriveTransformer (Jia et al., 2025),Think-Twice (Jia et al., 2023b), DriveDPO (Shang et al., 2025),WoTE (Li et al., 2025b), DriveAdapter (Jia et al., 2023a), Drive-JEPA (Wang et al., 2026),SimLingo (Renz et al., 2025) and HiP-AD (Tang et al., 2025) as twelve sensor-based end-to-end reference baselines.

4.2 Experimental Setup

Bench2Drive evaluation. We train the privileged planners on the official Bench2Drive training split, which contains 950 scenes across 12 towns, and evaluate all five planners on the held-out 220-route base split. Each method is executed through the official Bench2Drive control and scoring pipeline. We report Driving Score (DS), which combines route completion with penalties for driving infractions, and Success Rate (SR), the route completion rate without a disqualifying infraction. Following the Bench2Drive multi-ability protocol (Jia et al., 2024), we additionally report Overtaking, Merging, Emergency Brake, Give Way, and Traffic Signs scores and their unweighted mean.

Camera input study. We compare the base and front-view camera configurations on both Bench2Drive and NAVSIM using the augmentation protocol described in Sec. 3.5. The Bench2Drive experiments retain the training split, route set, controller, and DS/SR evaluation used above. We further study camera input in real-world scenes by evaluating the privileged planners on NAVSIM v1.1 (Dauner et al., 2024). Each trainable configuration is optimized on navtrain to predict eight future poses over four seconds and is evaluated on the 12,146 navtest scenarios using the official PDM scoring pipeline. NAVSIM camera variants use the real front-view image associated with each logged scene. We report PDM Score (PDMS) as the primary metric, together with no-at-fault collision, drivable-area compliance, ego progress, time-to-collision, and comfort, all on a 0–100 scale. Complete NAVSIM component results are given in Appendix E.1.

4.3 Main Results on Bench2Drive

Overall performance. Table 1 reports DS and SR for the five privileged planners and twelve sensor-based end-to-end baselines on Bench2Drive. Among the privileged planners, the DS ranking is PLUTO, Diffusion Planner, InstructDriver, BEV-RGB, and BEV-Mask; SR exhibits the same broad trend. Among the two matched raster baselines, BEV-RGB performs substantially better than BEV-Mask, showing how the scene is encoded has a substantial effect. PLUTO achieves the highest

Table 1: Bench2Drive results and model characteristics. The first five methods are privileged planners; the remaining twelve are sensor-based end-to-end baselines. Paradigm indicates the dominant modeling paradigm rather than specific network components. All methods are evaluated on the official Bench2Drive evaluation routes; * denotes expert-feature distillation (Jia et al., 2024).

Method	Input	Paradigm	Output	DS↑	SR↑
PLUTO	GT scene vectors	Transformer	Trajectory	**64.74**	**35.91**
Diffusion Planner	GT scene vectors	DiT	Trajectory	58.78	26.36
BEV-RGB	GT semantic RGB BEV	CNN	Trajectory	51.10	21.36
BEV-Mask	GT multi-channel BEV	CNN	Trajectory	41.30	13.64
InstructDriver	GT text prompt	LLM	Text trajectory	57.16	24.09
VAD	Multi-view cameras	Transformer	Trajectory	38.65	17.27
UniAD-Base	Multi-view cameras	Transformer	Trajectory	38.69	19.09
TCP-traj*	Front camera + ego state	CNN/MLP/GRU	Trajectory	51.44	26.36
ThinkTwice*	Multi-view cameras	Transformer	Trajectory	52.84	27.73
BridgeAD	Multi-view cameras	Transformer	Trajectory	50.06	22.73
DriveTransformer	Multi-view cameras	Transformer	Trajectory	60.20	35.91
DriveDPO	Cameras + LiDAR	Transformer	Trajectory	62.02	30.62
WoTE	Cameras + LiDAR	Transformer	Trajectory	61.71	31.36
DriveAdapter*	Cameras + LiDAR	Transformer	Control	54.96	30.45
Drive-JEPA	Front camera + ego state	Transformer	Trajectory	64.52	36.82
SimLingo	Multi-view cameras	VLA	Trajectory	85.07	67.27
HiP-AD	Multi-view cameras	Transformer	Trajectory	**86.77**	**69.09**

DS of 64.74 and SR of 35.91 among the tested privileged planners, this score is competitive with the sensor-based end-to-end methods, but it still remains below the best baselines. These results suggest that access to ground-truth scene information alone does not guarantee stronger closed-loop performance. Direct access to ground-truth scene states removes much of the uncertainty in scene estimation, but it does not by itself create the expected advantage over sensor-based end-to-end models. One possible reason is that ground-truth boxes and maps provide accurate structured scene representations but may not include visual cues available in raw sensor observations. End-to-end models, despite having to learn scene understanding from imperfect perception, have direct access to such observations and may exploit visual cues that are not explicitly represented by structured scene states.

This motivates our camera information study: we augment privileged planners with front-view camera observations to examine whether visual information beyond ground-truth scene states can improve planning performance. We evaluate this augmentation on both Bench2Drive and NAVSIM to further examine whether its effect depends on the characteristics of the driving environment.

Driving ability analysis. Table 2 reports performance across five driving abilities. Among privileged planners, PLUTO achieves the highest mean and leads in four abilities, while InstructDriver performs best on Traffic Signs. Across the two communities, similar overall performance does not imply similar abilities across driving scenarios. For example, PLUTO and DriveTransformer achieve nearly identical ability means, yet PLUTO performs better on Overtaking and Give Way, while DriveTransformer is stronger on Emergency Brake and Traffic Signs. Comparing methods from the two communities should therefore consider both their overall scores and the driving scenarios in which each method performs well or poorly.

Table 2: Driving ability scores (%) on the 220-route Bench2Drive evaluation. The first five methods are privileged planners, while the remaining six are sensor-based end-to-end baselines. Mean is the unweighted average of the five abilities. * denotes expert-feature distillation. (Jia et al., 2024).

Method	Overtaking↑	Merging↑	Emergency Brake↑	Give Way↑	Traffic Signs↑	Mean↑
PLUTO	**44.44**	**33.75**	**28.33**	**40.00**	34.21	**36.15**
Diffusion Planner	31.11	27.50	21.67	20.00	30.53	26.16
BEV-RGB	13.33	25.00	18.33	20.00	31.05	21.54
BEV-Mask	17.78	16.25	8.33	10.00	21.58	14.79
InstructDriver	17.78	25.00	23.33	20.00	**37.37**	24.70
VAD	13.04	17.19	29.82	16.67	19.77	19.30
UniAD-Base	11.59	10.94	40.35	**33.33**	8.14	20.87
TCP-traj*	14.49	14.06	59.64	33.33	63.95	37.10
ThinkTwice*	20.29	20.31	59.64	33.33	66.86	40.09
DriveTransformer	**30.43**	**39.06**	49.12	0.00	61.63	36.05
DriveAdapter*	26.09	21.88	**64.90**	**33.33**	**69.18**	**43.08**

4.4 Camera Information Results

Table 3 and Figure 4 summarize the effect of front-view camera augmentation on different planners across Bench2Drive and NAVSIM. On Bench2Drive, the camera inputs provide no consistent benefit: most planners exhibit slight degradation. This indicates that in the simulated environment, the additional visual observations probably do not provide helpful information for the privileged planners.

Table 3: Effect of front-view camera augmentation on Bench2Drive and NAVSIM. Base and +Camera denote configurations without and with front-view camera input.

	Bench2Drive			NAVSIM		
Method	Base DS	+Camera DS	ΔDS	Base PDMS	+Camera PDMS	ΔPDMS
PLUTO	64.74	62.60	-2.14	87.88	89.49	+1.61
Diffusion Planner	58.78	58.05	-0.73	87.51	88.36	+0.85
BEV-RGB	51.10	48.30	-2.80	76.65	75.46	-1.19
BEV-Mask	41.30	41.76	+0.46	64.14	71.02	+6.88
InstructDriver	57.16	57.10	-0.06	77.66	77.19	-0.47

We further extend the camera augmentation study to real-world driving environment using NAVSIM, where incorporating camera observations leads to the performance improvement. This suggests that camera observations may provide complementary driving cues in real-world scenarios that are not fully captured by the privileged scene representation, while such benefits appear to be more limited in simulated environments with simpler structure.

We note that this comparison follows a unified fusion protocol under fixed experimental settings. We leave it to future work to investigate whether stronger fusion mechanisms or explicit visual supervision would change the magnitude of these gaps.

4.5 Qualitative Camera Analysis

Figure 5 visualizes the camera study for PLUTO and Diffusion Planner on NAVSIM. With camera input, both planners produce trajectories that avoid the vehicle ahead and eliminate the predicted

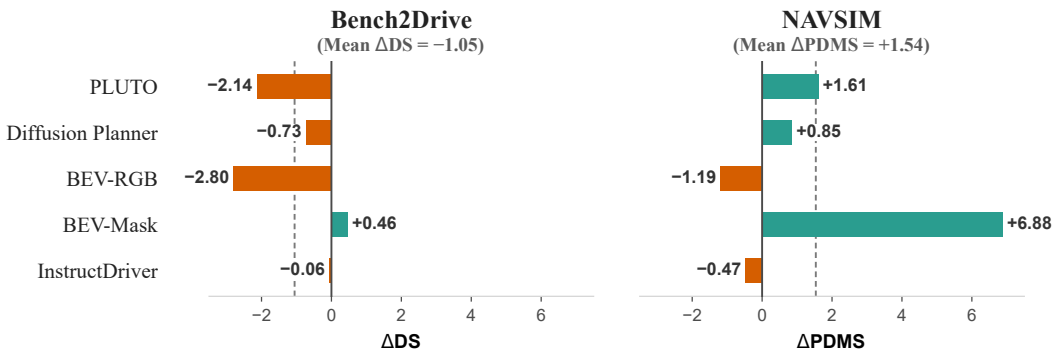

Figure 4: Effect of front-view camera augmentation on planning performance. Positive values indicate improvement. Dashed lines indicate the mean change for each benchmark.

collision. We infer that this trajectory change may be driven by the illuminated brake lights visible in the camera image, which indicate that the lead vehicle is likely to decelerate. This example illustrates how camera observations may provide useful visual cues for trajectory adjustment in interaction-sensitive scenarios.

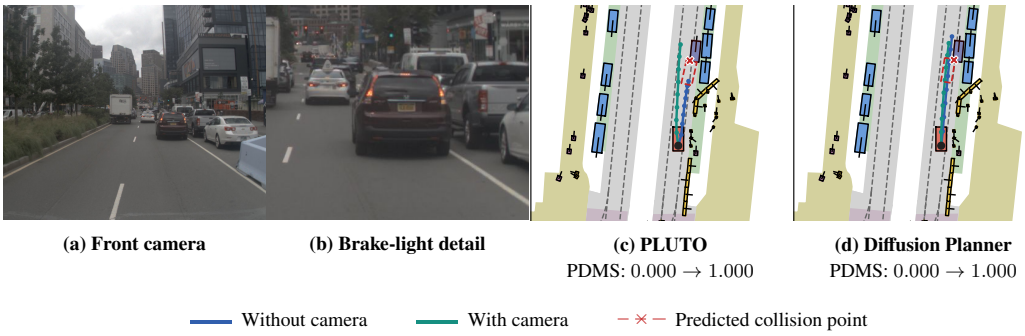

Figure 5: **Trajectory visualization for the camera study on NAVSIM.** Panel (b) enlarges the lead vehicle and shows its illuminated brake lights. The red dashed line and cross mark the collision without camera input. With camera input, both planners avoid the collision.

5 Conclusion

In this work, we introduce **Bench2Drive-Plan**, a unified benchmark that bridges privileged planning and end-to-end autonomous driving under a shared protocol. Through dedicated scene construction and planner adapters, Bench2Drive-Plan enables diverse planner paradigms, including vectorized, raster-based, diffusion-based, and language-based planners, to be evaluated within the same closed-loop driving environment. This framework provides a common basis for comparing methods that have traditionally been developed and assessed under separate evaluation settings. Our experiments show that ground-truth scene inputs do not inherently guarantee a performance advantage over end-to-end baselines. We further study the role of additional camera observations and find that their benefits vary across Bench2Drive and NAVSIM. We hope Bench2Drive-Plan can facilitate future research on how scene information, representation choices, and planning mechanisms interact in autonomous driving systems.

AI Use Statement

We have not used generative AI tools for any tasks with required disclosure, and all such tasks are not applicable to this work. Additionally, we used generative AI tools to edit the research paper to improve readability. We have reviewed all AI-assisted text and take responsibility for the final content of this work, including all text, claims, and artifacts.

REFERENCES

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv*, 2023.

Mayank Bansal, Alex Krizhevsky, and Abhijit Ogale. Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. *arXiv*, 2018.

Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv*, 2016.

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.

Holger Caesar, Juraj Kabzan, Kok Seang Tan, Whye Kit Fong, Eric Wolff, Alex Lang, Luke Fletcher, Oscar Beijbom, and Sammy Omari. nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles. *arXiv*, 2021.

Sergio Casas, Abbas Sadat, and Raquel Urtasun. Mp3: A unified model to map, perceive, predict and plan. In *CVPR*, 2021.

Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, et al. Argoverse: 3d tracking and forecasting with rich maps. In *CVPR*, 2019.

Dian Chen and Philipp Krähenbühl. Learning from all vehicles. In *CVPR*, 2022.

Dian Chen, Brady Zhou, Vladlen Koltun, and Philipp Krähenbühl. Learning by cheating. In *Conference on Robot Learning*, 2020.

Li Chen, Penghao Wu, Kashyap Chitta, Bernhard Jaeger, Andreas Geiger, and Hongyang Li. End-to-end autonomous driving: Challenges and frontiers. *TPAMI*, 2024.

Jie Cheng, Yingbing Chen, and Qifeng Chen. Pluto: Pushing the limit of imitation learning-based planning for autonomous driving. *arXiv*, 2024.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 2025.

Pranav Singh Chib and Pravendra Singh. Recent advancements in end-to-end autonomous driving using deep learning: A survey. *T-IV*, 2023.

Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *TPAMI*, 2022.

Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning. In *ICRA*, 2018.

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. *NeurIPS*, 2024.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *CoRL*, 2017.

Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R Qi, Yin Zhou, et al. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *ICCV*, 2021.

Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In *CVPR*, 2020.

Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv*, 2021.

Xuemin Hu, Shen Li, Tingyu Huang, Bo Tang, Rouxing Huai, and Long Chen. How simulation helps autonomous driving: A survey of sim2real, digital twins, and parallel intelligence. *IEEE Transactions on Intelligent Vehicles*, 2023a.

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*, 2023b.

Junjie Huang, Guan Huang, Zheng Zhu, Yun Ye, and Dalong Du. Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. *arXiv*, 2021.

Zhiyu Huang, Haochen Liu, and Chen Lv. Gameformer: Game-theoretic modeling and learning of transformer-based interactive prediction and planning for autonomous driving. In *ICCV*, 2023.

Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv*, 2022.

Xiaosong Jia, Yulu Gao, Li Chen, Junchi Yan, Patrick Langechuan Liu, and Hongyang Li. Driveadapter: Breaking the coupling barrier of perception and planning in end-to-end autonomous driving. In *ICCV*, 2023a.

Xiaosong Jia, Penghao Wu, Li Chen, Jiangwei Xie, Conghui He, Junchi Yan, and Hongyang Li. Think twice before driving: Towards scalable decoders for end-to-end autonomous driving. In *CVPR*, 2023b.

Xiaosong Jia, Zhenjie Yang, Qifeng Li, Zhiyuan Zhang, and Junchi Yan. Bench2drive: Towards multi-ability benchmarking of closed-loop end-to-end autonomous driving. *NeurIPS*, 2024.

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

Xiaosong Jia, Yuqian Shao, Zhenjie Yang, Qifeng Li, Zhiyuan Zhang, and Junchi Yan. Bench2drive-vl: Benchmarks for closed-loop autonomous driving with vision-language models, 2026. URL https://arxiv.org/abs/2604.01259.

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*, 2023.

Bo Jiang, Shaoyu Chen, Bencheng Liao, Xingyu Zhang, Wei Yin, Qian Zhang, Chang Huang, Wenyu Liu, and Xinggang Wang. Senna: Bridging large vision-language models and end-to-end autonomous driving. *International Journal of Computer Vision*, 2026.

Sicong Jiang, Zilin Huang, Kangan Qian, Ziang Luo, Tianze Zhu, Yang Zhong, Yihong Tang, Menglin Kong, Yunlong Wang, Siwen Jiao, et al. A survey on vision-language-action models for autonomous driving. In *ICCV*, 2025.

Alex Kendall, Jeffrey Hawke, David Janz, Przemyslaw Mazur, Daniele Reda, John-Mark Allen, Vinh-Dieu Lam, Alex Bewley, and Amar Shah. Learning to drive in a day. In *ICRA*, 2019.

Scott T Leutenegger, Mario A Lopez, and Jeffrey Edgington. Str: A simple and efficient algorithm for r-tree packing. In *Proceedings 13th international conference on data engineering*, 1997.

Ding Li, Qichao Zhang, and Dongbin Zhao. Ranging from prediction to planning via machine learning approaches for autonomous driving: a survey. *Artificial Intelligence Review*, 2026.

Yingyan Li, Lue Fan, Jiawei He, Yuqi Wang, Yuntao Chen, Zhaoxiang Zhang, and Tieniu Tan. Enhancing end-to-end autonomous driving with latent world model. In *ICLR*, 2025a.

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*, 2025b.

Yinhao Li, Zheng Ge, Guanyi Yu, Jinrong Yang, Zengran Wang, Yukang Shi, Jianjian Sun, and Zeming Li. Bevdepth: Acquisition of reliable depth for multi-view 3d object detection. In *Proceedings of the AAAI conference on artificial intelligence*, 2023.

Yun Li, Kai Katsumata, Ehsan Javanmardi, and Manabu Tsukada. Large language models for human-like autonomous driving: A survey. In *ITSC*, 2024.

Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. *arXiv*, 2022.

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*, 2025.

Christian Löwens, Thorben Funke, and Alexandru Paul Condurache. Driving on memory, 2026.

Jiageng Mao, Yuxi Qian, Junjie Ye, Hang Zhao, and Yue Wang. Gpt-driver: Learning to drive with gpt. *arXiv*, 2023.

Seongjeong Park, Jinu Pahk, Lennart Lorenz Freimuth Jahn, Yongseob Lim, Jinung An, and Gyeungho Choi. A study on quantifying sim2real image gap in autonomous driving simulations using lane segmentation attention map similarity. In *International Conference on Intelligent Autonomous Systems*, 2023.

Stefanos Pasios and Nikos Nikolaidis. Carla2real: A tool for reducing the sim2real appearance gap in carla simulator. *T-ITS*, 2025.

Tung Phan-Minh, Elena Corina Grigore, Freddy A Boulton, Oscar Beijbom, and Eric M Wolff. Covernet: Multimodal behavior prediction using trajectory sets. In *CVPR*, 2020.

Jonah Philion and Sanja Fidler. Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *European conference on computer vision*, 2020.

Aditya Prakash, Kashyap Chitta, and Andreas Geiger. Multi-modal fusion transformer for end-to-end autonomous driving. In *CVPR*, 2021.

Katrin Renz, Kashyap Chitta, Otniel-Bogdan Mercea, A Koepke, Zeynep Akata, and Andreas Geiger. Plant: Explainable planning transformers via object-level representations. *arXiv*, 2022.

Katrin Renz, Long Chen, Elahe Arani, and Oleg Sinavski. Simlingo: Vision-only closed-loop autonomous driving with language-action alignment. In *CVPR*, 2025.

ShuYao Shang, Yuntao Chen, Yuqi Wang, Yingyan Li, and Zhaoxiang Zhang. DriveDPO: Policy learning via safety DPO for end-to-end autonomous driving. In *NeurIPS*, 2025.

Hao Shao, Letian Wang, Ruobing Chen, Hongsheng Li, and Yu Liu. Safety-enhanced autonomous driving using interpretable sensor fusion transformer. In *CoRL*, 2023.

Hao Shao, Yuxuan Hu, Letian Wang, Guanglu Song, Steven L Waslander, Yu Liu, and Hongsheng Li. Lmdrive: Closed-loop end-to-end driving with large language models. In *CVPR*, 2024.

Yuqian Shao, Xiaosong Jia, Langechuan Liu, and Junchi Yan. Can users specify driving speed? bench2drive-speed: Benchmark and baselines for desired-speed conditioned autonomous driving. *arXiv*, 2026.

Chonghao Sima, Katrin Renz, Kashyap Chitta, Li Chen, Hanxue Zhang, Chengen Xie, Jens Beißwenger, Ping Luo, Andreas Geiger, and Hongyang Li. Drivelm: Driving with graph visual question answering. In *ECCV*, 2024.

Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*, 2020.

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.

Yingqi Tang, Zhuoran Xu, Zhaotie Meng, and Erkang Cheng. Hip-ad: Hierarchical and multi-granularity planning with deformable attention for autonomous driving in a single decoder. In *ICCV*, 2025.

Robert Tarjan. Depth-first search and linear graph algorithms. *SIAM journal on computing*, 1972.

Xiaoyu Tian, Junru Gu, Bailin Li, Yicheng Liu, Yang Wang, Zhiyong Zhao, Kun Zhan, Peng Jia, Xianpeng Lang, and Hang Zhao. Drivevlm: The convergence of autonomous driving and large vision-language models, 2024.

Wenwen Tong, Chonghao Sima, Tai Wang, Li Chen, Silei Wu, Hanming Deng, Yi Gu, Lewei Lu, Ping Luo, Dahua Lin, et al. Scene as occupancy. In *ICCV*, 2023.

Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 2020.

Linhan Wang, Zichong Yang, Chen Bai, Guoxiang Zhang, Xiaotong Liu, Xiaoyin Zheng, Xiao-Xiao Long, Chang-Tien Lu, and Cheng Lu. Drive-jepa: Video jepa meets multimodal trajectory distillation for end-to-end driving. *arXiv*, 2026.

Xinshuo Weng, Boris Ivanovic, Yan Wang, Yue Wang, and Marco Pavone. Para-drive: Parallelized architecture for real-time autonomous driving. In *CVPR*, 2024.

Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting, 2023.

Katharina Winter, Abhishek Vivekanandan, Rupert Polley, Yinzhe Shen, Christian Schlauch, Mohamed-Khalil Bouzidi, Bojan Derajic, Natalie Grabowsky, Annajoyce Mariani, Dennis Rochau, et al. Generative ai for autonomous driving: A review. *arXiv*, 2025.

Penghao Wu, Xiaosong Jia, Li Chen, Junchi Yan, Hongyang Li, and Yu Qiao. Trajectory-guided control prediction for end-to-end autonomous driving: A simple yet strong baseline. *NeurIPS*, 2022.

Zhenhua Xu, Yujia Zhang, Enze Xie, Zhen Zhao, Yong Guo, Kwan-Yee K Wong, Zhenguo Li, and Hengshuang Zhao. Drivegpt4: Interpretable end-to-end autonomous driving via large language model. *IEEE Robotics and Automation Letters*, 2024.

Bo Zhang, Heye Huang, Chunyang Liu, Yaqin Zhang, and Zhenhua Xu. Invdriver: Intra-instance aware vectorized query-based autonomous driving transformer. *Journal of Intelligent and Connected Vehicles*, 2025a.

Bozhou Zhang, Nan Song, Xin Jin, and Li Zhang. Bridging past and future: End-to-end autonomous driving with historical prediction and planning. In *CVPR*, 2025b.

Ruijun Zhang, Xianda Guo, Wenzhao Zheng, Chenming Zhang, Kurt Keutzer, and Long Chen. Instruct large language models to drive like humans. *arXiv*, 2024.

Zhiyuan Zhang, Zhenghao Jin, Yanlun Peng, Xianda Guo, Haoran Liu, Shaofeng Zhang, Xingjun Ma, Zuxuan Wu, Junchi Yan, Xiaosong Jia, et al. Bench2drive-robust: Benchmarking closed-loop autonomous driving under deployment perturbations. *arXiv*, 2026.

Yinan Zheng, Ruiming Liang, Kexin Zheng, Jinliang Zheng, Liyuan Mao, Jianxiong Li, Weihao Gu, Rui Ai, Shengbo Li, Xianyuan Zhan, et al. Diffusion-based planning for autonomous driving with flexible guidance. In *ICLR*, 2025.

Xingcheng Zhou, Mingyu Liu, Ekim Yurtsever, Bare Luka Zagar, Walter Zimmer, Hu Cao, and Alois C Knoll. Vision language models in autonomous driving: A survey and outlook. *T-IV*, 2024.

Runwen Zhu, Jianbo Zhao, Diankun Zhang, Guoan Wang, Xiwu Chen, Siyu Zhang, Jiahao Gong, Qibin Zhou, Wenyuan Zhang, Ningzi Wang, et al. Sparsead: Sparse query-centric paradigm for efficient end-to-end autonomous driving. *TAI*, 2025.

Zhengbang Zhu, Hanye Zhao, Haoran He, Yichao Zhong, Shenyu Zhang, Haoquan Guo, Tingting Chen, and Weinan Zhang. Diffusion models for reinforcement learning: A survey. *arXiv*, 2023.

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*, 2025.

A Training Hyperparameters

Tables 4, 5, 6, and 7 summarize the training hyperparameters of each planner on Bench2Drive and NAVSIM, the camera-augmented Bench2Drive setting and the camera-augmented NAVSIM setting, respectively. Batch denotes the effective batch size. All models are trained in FP32 with gradient clipping at 5.0. On NAVSIM, BEV-RGB and BEV-Mask train the head and backbone with separate learning rates.

Table 4: Training hyperparameters on Bench2Drive. LR denotes the initial learning rate; WD denotes weight decay.

Method	Epochs	Batch	Optimizer	LR	WD	Schedule
PLUTO	25	128	AdamW	10^{-3}	10^{-4}	3-epoch warmup + cosine
Diffusion Planner	400	2048	AdamW	5×10^{-4}	10^{-2}	5-epoch warmup + cosine
BEV-RGB	25	128	Adam	10^{-3}	10^{-4}	constant
BEV-Mask	25	128	Adam	10^{-3}	10^{-4}	constant
LLM planner	3	8	AdamW	10^{-4}	0	cosine + 0.1 warmup

Table 5: Training hyperparameters for the camera-augmented Bench2Drive setting. PLUTO, Diffusion Planner, and the LLM planner follow a two-stage schedule. Stage 1 freezes the ResNet-34 backbone. Stage 2 unfreezes the backbone and jointly fine-tunes the camera branch and planner. BEV-RGB and BEV-Mask use a single-stage schedule that jointly trains their unfrozen ResNet-50 BEV and ResNet-34 camera encoders. All camera backbones are initialized from ImageNet-pretrained weights.

Method	Stage	Epochs	Batch	LR	Backbone
PLUTO	1	25	128	10^{-3}	ResNet-34 (frozen)
	2	5	128	10^{-4}	ResNet-34 (unfrozen)
Diffusion Planner	1	400	2048	5×10^{-4}	ResNet-34 (frozen)
	2	100	128	10^{-4}	ResNet-34 (unfrozen)
BEV-RGB	single	25	64	10^{-3}	ResNet-34 (unfrozen)
BEV-Mask	single	25	64	10^{-3}	ResNet-34 (unfrozen)
LLM planner	1	3	8	10^{-4}	ResNet-34 (frozen)
	2	1	16	10^{-4}	ResNet-34 (unfrozen)

Table 6: Training hyperparameters for the trainable NAVSIM planners. LR denotes the initial learning rate; WD denotes weight decay.

Method	Epochs	Batch	Optimizer	LR	WD	Schedule
PLUTO	50	32	AdamW	10^{-3}	10^{-4}	3-epoch warmup + cosine
Diffusion Planner	72	64	AdamW	10^{-4}	10^{-2}	5-epoch warmup + cosine
BEV-RGB	50	128	AdamW	$10^{-4}/10^{-5}$	10^{-4}	3-epoch warmup + cosine
BEV-Mask	50	64	AdamW	$10^{-4}/10^{-5}$	10^{-4}	3-epoch warmup + cosine
LLM planner	3	8	AdamW	10^{-4}	0	cosine + 0.1 warmup

Table 7: Training hyperparameters for the camera-augmented NAVSIM setting. PLUTO, Diffusion Planner, and the LLM planner follow a two-stage schedule. Stage 1 freezes the ResNet-34 backbone. Stage 2 unfreezes the backbone and jointly fine-tunes the camera branch and planner. BEV-RGB and BEV-Mask use a single-stage schedule that jointly trains their unfrozen ResNet-50 BEV and ResNet-34 camera encoders. For BEV-RGB and BEV-Mask, the trajectory head is trained at the higher learning rate and the two backbones at the lower rate. All camera backbones are initialized from ImageNet-pretrained weights.

Method	Stage	Epochs	Batch	LR	Backbone
PLUTO	1	50	32	10^{-3}	ResNet-34 (frozen)
	2	10	32	10^{-4}	ResNet-34 (unfrozen)
Diffusion Planner	1	60	64	10^{-4}	ResNet-34 (frozen)
	2	15	16	2×10^{-5}	ResNet-34 (unfrozen)
BEV-RGB	single	50	128	$10^{-4}/10^{-5}$	ResNet-34 (unfrozen)
BEV-Mask	single	50	64	$10^{-4}/10^{-5}$	ResNet-34 (unfrozen)
LLM planner	1	3	8	10^{-4}	ResNet-34 (frozen)
	2	1	16	10^{-4}	ResNet-34 (unfrozen)

B Bench2Drive Implementation Details

B.1 Scene Feature Construction and planner adaptation

Bench2Drive driving information are preprocessed into per-frame feature files for training. During closed-loop evaluation, the agent reconstructs the same feature definitions online from the current CARLA state. Both paths use a right-handed coordinate convention and an SE(2) transform anchored at the current ego pose. Actor identities are associated across frames, and future frames are accessed only when constructing training targets.

Traffic participants. We first extract ground-truth position, velocity and category for the ego vehicle and traffic participants from the Bench2Drive annotations, and associate observations across frames using persistent actor identifiers. All states are transformed into a right-handed SE(2) frame anchored at the current ego pose. Benchmark-specific categories, including ordinary vehicles, pedestrians, two-wheelers, emergency vehicles, and vehicles with an open door, are retained in the shared scene representation. Missing history is padded and marked by validity masks, while static objects are recorded separately from dynamic actors. Each planner adapter then converts these shared states into its model-specific input format, using the required number of actors, history sampling scheme, and feature fields.

The Diffusion Planner retains the 32 nearest non-ego actors. Each actor is represented over 21 frames (two seconds of history and the current frame at 10 Hz) by ego-relative position, heading sine and cosine, velocity, footprint, and a five-way category encoding. Its training targets contain 80 future ego poses and trajectories for up to 10 neighboring agents over eight seconds, together with explicit validity masks. Five nearby static objects are represented by pose, footprint, and type.

The PLUTO input retains up to 64 actors on the same 21-history/80-future timeline. It stores position, heading, velocity, footprint, category, and validity fields in PLUTO's agent-feature schema.

Its current ego state additionally includes speed, acceleration, steering, and yaw rate. Future actor states are retained for training targets and reference-line projection only.

The two raster baselines render the scene into an ego-centered 300×300 raster at 5 px/m (0.2 m/px, a 60 m field of view). Actor and traffic rule geometry is drawn directly from benchmark annotations. BEV-RGB distinguishes semantic categories by color and encodes the full 21-frame history through brightness, whereas BEV-Mask samples five frames and preserves each category and time step in a separate binary channel. Both models regress the 80-pose trajectory over eight seconds.

The LLM planner serializes up to 10 nearest dynamic agents as text, each described by a category (vehicle, walker, two-wheeler, emergency vehicle, or open-door vehicle), bounding-box shape, ego-relative pose, velocity, and a one-second position prediction, with static obstacles listed separately. The ego is described by its shape, pose, velocity, and acceleration together with a four-pose (two-second) historical trajectory;For prediction,the model decodes the ego-relative 16-pose trajectory over eight seconds as text.

CARLA map vectorization. For each town, we preprocess the exported CARLA HD map into separate collections of lane and crosswalk objects. For each lane, fragmented centerline and boundary segments are merged according to endpoint proximity. Boundary segments are classified as left or right relative to the centerline, aligned to a consistent direction, and combined into a lane polygon while retaining the lane topology. Crosswalk records are converted into polygons. Separate STRtree indices are built for the lane and crosswalk collections, enabling efficient spatial queries within the local map region. Dense route points are matched to their containing lane polygons, and the lane topology constrains transitions to recover an ordered route-lane sequence.

Speed limits and traffic lights are associated with lanes in separate steps. Town-level speed-limit samples are converted to the right-handed map frame and indexed with a cKDTree. For each queried lane, we retrieve speed-limit samples within 3 m of its centerline and use their median value. At each frame, we select traffic-light annotations marked as affecting the ego vehicle, query each trigger location against the lane STRtree, and write its one-hot light state into the corresponding lane tokens.

Ego-centered local-window extraction. Most privileged planners evaluated in this work were originally developed on nuPlan, but the lane length distribution differs substantially between nuPlan and CARLA. In our processed maps, 22.3% of Bench2Drive lanes exceed 100 m, compared with 5.0% in nuPlan; above 200 m, the proportions are 12.1% and 0.8%, respectively. This gap is important for planners that encode each lane with a fixed budget of 20 points. Uniformly sampling a fixed number of points over an entire long CARLA lane places many samples far from the ego vehicle and represents the local road geometry too sparsely. This differs from the denser local lane representation typically encountered in nuPlan.

We address this gap with an ego-centered local-window procedure. For each queried lane, we intersect its centerline with the local map region and select the connected segment closest to the ego vehicle. If the intersection does not yield a valid segment, we project the ego position onto the complete centerline and construct a local interval around the projected arc length. We convert the start and end positions of the selected segment into normalized progress along the complete centerline. The same progress interval is then used to crop and resample the centerline and both lane boundaries into 20 aligned points. This preserves the correspondence between the centerline and boundaries while concentrating the fixed point budget on geometry relevant to the current planning context. In our Diffusion Planner experiment, replacing whole-lane sampling with this procedure substantially improved performance on Bench2Drive, showing the importance of correcting this map representation gap when transferring planners from nuPlan to CARLA.

After preprocessing, each adapter combines the extracted map information with the remaining scene features and converts them into the input format required by its planner.

The Diffusion adapter orders local map elements by distance and retains up to 70 lane or crosswalk tokens and 25 route-lane tokens, with 20 points per token. Lane tokens contain centerline position and direction, boundary offsets, traffic-light state, lane/crosswalk type, and speed-limit value and validity; route-lane tokens reuse the corresponding ordered local-lane geometry.

The PLUTO adapter resamples each lane into 20 aligned segments along its centerline and two boundaries, while crosswalks use the same point sequence format. Each map element contains point

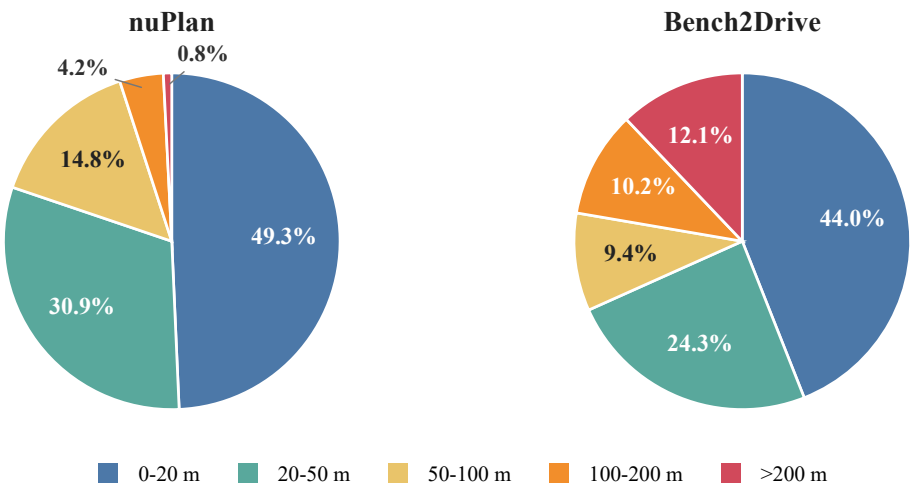

Figure 6: **Distribution of lane lengths in nuPlan and Bench2Drive.** The substantial difference in the prevalence of long lanes between CARLA and nuPlan motivates our ego-centered local-window extraction procedure.

positions, direction vectors, orientations, and side labels, together with polygon attributes for map type, route membership, traffic light state, and road block identity. Candidate reference lines are constructed by starting from nearby route lanes and traversing the route topology. They are limited to 100 m, duplicate candidates are removed, and the remaining lines are sampled at 1 m intervals. Future ego poses are projected onto each candidate to provide progress and distance targets for reference line supervision. To form the cost map, the adapter rasterizes drivable lane polygons, removes stationary actor footprints, and converts the result into a signed distance field.

For the raster baselines, we first rasterize each CARLA town map into a road surface layer and a lane geometry layer containing lane centerlines and boundaries. At each frame, both layers are cropped and rotated around the current ego pose to form a local BEV representation. We then add the near and far route command points, stop signs, and the stop lines of traffic lights affecting the ego vehicle from the current annotations.

BEV-RGB stores these map elements in one color image: road, route, and lane geometry are rendered in dark gray, light gray, and magenta, while traffic light stop lines use green, yellow, or red and stop signs use dark yellow. BEV-Mask instead represents road, route, and lane geometry with three binary channels. Green traffic lights, combined red and yellow traffic lights, and stop signs each use five temporal channels.

The LLM planner serializes the ego's current lane centerline and its left and right boundaries, each sampled forward up to 100 m at ten points, together with the current traffic-light status, the speed limit, and the upcoming stop line as a line segment with its distance.

B.2 Planner Implementations and Training Configurations

Diffusion Planner. We adapt the Diffusion Planner implementation (Zheng et al., 2025) to Bench2Drive while retaining its overall architecture and planning formulation. Bench2Drive contains more diverse interactive scenarios and semantic categories than the original planner input. For example, the *YieldToEmergencyVehicle* scenario requires the planner to distinguish an emergency vehicle from an ordinary vehicle. We therefore adapt the encoder to preserve these task-relevant semantics. The neighbor type embedding is expanded from three to five categories: vehicles, pedestrians, two-wheelers, emergency vehicles, and open-door vehicles. We also add a binary lane/crosswalk type embedding to the lane encoder, while leaving the remaining model components unchanged.

We train the adapted model for 400 epochs with four-GPU distributed data parallelism, a per-GPU batch size of 512, AdamW at an initial learning rate of 5×10^{-4}, five warmup epochs followed by cosine decay. Encoder and decoder drop-path rates are both 0.1. An exponential moving average of

model parameters is maintained with decay 0.999. We retain the original Diffusion Planner loss and state perturbation augmentation.

PLUTO. We adapt the PLUTO implementation (Cheng et al., 2024) to Bench2Drive while retaining its agent and map encoders, reference-line-conditioned planning decoder, trajectory-mode head, and associated planning losses. PLUTO faces the same actor-category mismatch as Diffusion Planner. We therefore expand its agent type embedding from four to eight entries. All other model components follow the original implementation.

We train the adapted model for 25 epochs on a single GPU with a batch size of 128, using AdamW at an initial learning rate of 10^{-3} and weight decay 10^{-4}, with a three-epoch linear warmup followed by cosine decay to 10^{-6}. It predicts 12 trajectory modes per reference line from a 21-frame (two-second) history and outputs 80 poses over eight seconds.

BEV-RGB and BEV-Mask. Both raster baselines use a 300×300 ego-centered input at 0.2 m/px, corresponding to a 60 m field of view. BEV-RGB combines map, route, traffic rules, actors, and the ego vehicle in a three-channel semantic image. Actor categories are distinguished by color, and the complete 21-frame history is encoded by shifting older footprints toward white while retaining the full category color at the current frame. The image is scaled to $[0, 1]$ and normalized with ImageNet statistics. BEV-Mask represents the scene as a 38-channel binary tensor. Three channels encode road, route, and lane geometry. Vehicles, including the ego vehicle and open-door indicators, emergency vehicles, static objects, and walkers each use five channels sampled uniformly from the 21-frame history. Green traffic lights, combined red and yellow traffic lights, and stop signs use another five channels each. Missing history is padded with zeros.

Both models use an ImageNet-pretrained ResNet-50 backbone. BEV-RGB retains the standard three-channel input stem, whereas the input convolution of BEV-Mask is expanded to 38 channels. The remaining backbone structure is identical. After global average pooling, a linear trajectory head outputs 240 values, which are reshaped into 80 future ego poses over eight seconds, with each pose represented by $(x, y, \text{heading})$. We train both models for 25 epochs with Adam, a batch size of 128, an initial learning rate of 10^{-3}, weight decay of 10^{-4}. The imitation loss combines position MSE and wrapped angular L1 loss for heading with equal weights.

InstructDriver. We adapt InstructDriver (Zhang et al., 2024) to Bench2Drive while preserving its Perception, Ego-States, Historical Trajectory, and Map Insights prompt structure and LoRA recipe(Hu et al., 2021). We serialize the Bench2Drive scene representation into the corresponding prompt fields and use 16 ego-relative poses as trajectory targets. Qwen2.5-1.5B-Instruct is fine-tuned for three epochs with rank-8 LoRA applied to all linear modules(Bai et al., 2023). At inference, greedy decoding (temperature 0, maximum generation length 3072) produces the (x, y, yaw) trajectory as text, which is converted to the Bench2Drive trajectory interface.

B.3 Closed-Loop Execution

All methods are evaluated on the official 220-route Bench2Drive split through CARLA Leaderboard agents. The predicted ego trajectory is transformed once from the model's right-handed ego frame to CARLA's left-handed local frame. All such methods then use the official Bench2Drive trajectory-tracking controller with identical parameters for every planner.

Given predicted waypoints $\{p_i\}_{i=1}^{N}$, the controller computes the desired speed as $v_{\text{des}} = \frac{10}{N-1} \sum_i \|p_{i+1} - p_i\|_2$ and selects the waypoint associated with the segment midpoint nearest 4 m as the steering aim point. The lateral PID gains (K_P, K_I, K_D) are $(0.75, 0.75, 0.3)$, while the longitudinal gains are $(5.0, 0.5, 1.0)$; both use a 40-step error window. Braking is activated when the desired speed is below $0.4\,\text{m/s}$ or the current-to-desired speed ratio exceeds 1.1. The longitudinal speed error is clipped at 0.25 and throttle at 0.75. Final steering, throttle, and brake commands are clipped to $[-1, 1]$, $[0, 0.75]$, and $[0, 1]$, respectively, and throttle is suppressed above an ego speed of $5\,\text{m/s}$.

C NAVSIM IMPLEMENTATION DETAILS

C.1 SCENE FEATURE CONSTRUCTION AND PLANNER ADAPTATION

NAVSIM scenes are converted into planner inputs from the dataset annotations and map API. All geometric features are expressed in a local SE(2) frame anchored at the current ego pose. The standard observation contains four recorded history frames. Future frames are used only to build training targets. Each planner outputs eight ego-relative poses at 0.5-second intervals over four seconds, following the NAVSIM trajectory interface.

Traffic participants and map information. We extract the ego state, surrounding actors, local map geometry, route, traffic-light states, and speed limits from each scene. Each adapter then organizes this information according to the actor capacity, temporal sampling, and feature definitions required by its planner.

The Diffusion Planner input contains the current ego state, four-frame histories for up to 32 neighboring agents, and five static objects. Its vectorized map input retains up to 70 local lanes and 25 route lanes, with 20 sampled points per polyline, together with traffic-light and speed-limit attributes. The training targets contain the future ego trajectory and trajectories for up to 10 neighboring agents.

The PLUTO input retains up to 48 actors and 10 static obstacles. Lane and route geometry is converted into polygon features with 20 sampled points, and the adapter additionally constructs candidate reference lines and the cost map used by PLUTO's planning objective.

The two raster baselines render the four-frame scene history into an ego-centered 300×300 BEV at 5 px/m. BEV-RGB represents the map, route, traffic rules, actors, and ego vehicle in a three-channel semantic image. Actor classes are distinguished by color, and arrows encode current velocities. BEV-Mask represents the same scene information in 24 binary channels. Three channels encode road, route, and lane geometry; four temporal channels are used for each of vehicles, static objects, walkers, green lights, and red lights; and one channel represents stop signs.

The LLM planner follows the InstructDriver prompt structure (Zhang et al., 2024). The adapter serializes the ego state, up to 10 surrounding actors, and nearby map and traffic information into an ego-centric text prompt. The model generates eight ego-relative (x, y, yaw) poses as text, which are parsed into the NAVSIM trajectory format.

C.2 PLANNER IMPLEMENTATIONS AND TRAINING CONFIGURATIONS

Diffusion Planner. We adapt the Diffusion Planner implementation (Zheng et al., 2025) to NAVSIM while retaining its overall architecture and planning formulation. The adapter reuses the original agent, static object, lane, and route encoders, scene fusion, diffusion decoder, and sampler. We retain the original state perturbation augmentation and adapt the training loss to the NAVSIM setting.

We train the adapted model for 72 epochs with AdamW, a batch size of 64, and an initial learning rate of 10^{-4}. The learning rate uses five warmup epochs followed by cosine decay to 10^{-5}. An exponential moving average of the model parameters is maintained with decay 0.999.

PLUTO. We adapt PLUTO (Cheng et al., 2024) while retaining its map encoder, planning decoder, trajectory-mode head, and planning losses. Four NAVSIM history frames yield only three temporal difference tokens, which are too short for the neighborhood kernel in PLUTO's original temporal encoder. We therefore replace this block with a two-layer self-attention encoder and retain the remaining model components.

We train the adapted model for 50 epochs with AdamW, a batch size of 32, and an initial learning rate of 10^{-3}. The learning rate uses three warmup epochs followed by cosine decay to 10^{-6}. We retain PLUTO's planning, neighboring-agent prediction, and collision losses, together with its positive-sample perturbation adapted to the NAVSIM input.

BEV-RGB and BEV-Mask. Both raster planners use an ImageNet-pretrained ResNet-50 backbone and a linear trajectory head. BEV-RGB retains the standard three-channel input stem, whereas the BEV-Mask input convolution is expanded to 24 channels. The head outputs 24 values, which are reshaped into eight future $(x, y, \mathrm{heading})$ poses.

We train both models for 50 epochs with AdamW and three warmup epochs followed by cosine decay. BEV-RGB and BEV-Mask use batch sizes of 128 and 64, respectively. For both models, the trajectory head uses an initial learning rate of 10^{-4} and the pretrained backbone uses 10^{-5}. Their shared imitation loss combines position MSE and wrapped angular L1 loss for heading with equal weights.

InstructDriver. We fine-tune Qwen2.5-1.5B-Instruct with rank-8 LoRA applied to all linear modules. Training uses the token-level supervised fine-tuning objective and the Qwen chat template for three epochs, with an batch size of 8, an initial learning rate of 10^{-4}, and a cosine schedule with a warmup ratio of 0.1. At evaluation time, greedy decoding generates at most 1024 new tokens before the predicted poses are parsed from the response.

C.3 Trajectory Evaluation

All methods are evaluated on the 12,146 scenarios of the official navtest split. For each scene, the adapter reconstructs the same input representation used during training and returns eight ego-relative poses spanning four seconds.

The NAVSIM v1.1 scorer interpolates the 2 Hz planner output to a 10 Hz, 40-pose proposal and runs PDM simulation under the fixed logged scene evolution. We use a shared precomputed metric cache and report the official PDM Score (PDMS), together with no-at-fault collision, drivable-area compliance, ego progress, time-to-collision, and comfort.

D Camera Augmentation Details

D.1 Visual Encoder and Fusion

All planners follow the encoder-level concatenation protocol of Sec. 3.5. The front-view RGB image is resized to 900×256 pixels and encoded by an ImageNet-pretrained ResNet-34, and a learned projection (a linear layer followed by GELU and LayerNorm) maps the ResNet output into the planner's feature space. Image normalization uses the ImageNet mean and standard deviation in both training and evaluation. The projected camera feature is then combined with each planner's scene representation. Section D.2 describes the model-specific fusion designs, and Table 5 reports the training hyperparameters.

D.2 Planner-Specific Camera Fusion

Diffusion Planner. Diffusion Planner first encodes the front-view image with ResNet-34. A linear layer followed by GELU and LayerNorm maps the 1000-dimensional image feature to the planner's hidden dimension. The resulting camera token is scaled by a learnable factor initialized to 0.1 and appended to the encoded agent, static-object, and lane tokens before the scene fusion Transformer. Training uses two stages. Stage 1 trains the projection and planner on precomputed features from a frozen ResNet-34 for 400 epochs, with batch size 512 and learning rate 5×10^{-4}. Stage 2 loads the first-stage weights, unfreezes the ResNet-34, and jointly fine-tunes the complete model on raw images for 100 epochs, with batch size 128 and learning rate 10^{-4}. Both stages use four GPUs, five warmup epochs followed by cosine decay, and EMA decay 0.999.

PLUTO. PLUTO also encodes the front-view image with ResNet-34. A linear layer followed by GELU and LayerNorm maps the image feature to PLUTO's hidden dimension. The resulting camera token is scaled by a learnable factor initialized to 0.1 and appended to the agent, map, and reference-line tokens before the fusion Transformer. Training uses two stages. Stage 1 trains the projection and planner on precomputed features from a frozen ResNet-34 for 25 epochs, with batch size 128 and learning rate 10^{-3}. Stage 2 unfreezes the ResNet-34 and jointly fine-tunes the complete model on raw images for five epochs, with batch size 128 and learning rate 10^{-4}.

BEV-RGB and BEV-Mask. BEV-RGB and BEV-Mask encode the front-view image into a 512-dimensional feature using ResNet-34. A linear layer followed by GELU and LayerNorm maps this output to a 256-dimensional camera feature. This feature is concatenated with the global feature produced by the ResNet-50 BEV encoder. A fusion MLP combines the two features before the trajectory head. The ResNet-34, ResNet-50, fusion module, and trajectory head are jointly trained on raw images for 25 epochs, with batch size 64 and learning rate 10^{-3}.

InstructDriver. InstructDriver encodes the front-view image with ResNet-34. A linear layer followed by GELU and LayerNorm maps the image feature to the hidden dimension of Qwen2.5-1.5B-Instruct. The resulting camera token is scaled by a learnable factor initialized to 0.1 and prepended to the text token sequence. Training uses two stages. Stage 1 trains the LoRA adapter on precomputed features from a frozen ResNet-34 for three epochs, with batch size 8 and learning rate 10^{-4}. Stage 2 unfreezes the ResNet-34 and jointly fine-tunes it with the LoRA adapter on raw images for one epoch, with batch size 16 and the same learning rate.

E Additional Results

E.1 NAVSIM Diagnostic Results

Tables 8 and 9 report all NAVSIM metrics before and after camera augmentation. Scores use a 0–100 scale over 12,146 test scenes, and the metrics follow Sec. 4.1.

Table 8: Complete NAVSIM results without front-view camera input. The end-to-end results are officially reported NAVSIM references. Bold marks the highest score within each method group (Dauner et al., 2024).

Method	PDMS ↑	NAC ↑	DAC ↑	EP ↑	TTC ↑	Comfort ↑
Privileged planners						
PLUTO	**87.88**	97.43	97.33	**82.18**	93.39	99.93
Diffusion Planner	87.51	**97.95**	**97.55**	80.90	**93.75**	98.66
BEV-RGB	76.65	94.53	88.86	72.66	87.48	99.12
BEV-Mask	64.14	88.44	82.04	61.61	77.89	99.88
InstructDriver	77.66	96.63	88.30	71.03	91.03	**99.99**
Official NAVSIM end-to-end references						
LTF (Chitta et al., 2022)	83.80	97.40	**92.80**	79.00	92.40	**100.00**
TransFuser (Chitta et al., 2022)	**84.00**	97.70	**92.80**	79.20	92.80	**100.00**
UniAD (Hu et al., 2023b)	83.40	97.80	91.90	78.80	92.90	**100.00**
PARA-Drive (Weng et al., 2024)	**84.00**	**97.90**	92.40	**79.30**	**93.00**	99.80

PLUTO and Diffusion Planner achieve 87.88 and 87.51 PDMS, respectively, compared with 84.00 for the strongest included end-to-end reference. The difference is most visible in drivable-area compliance and ego progress, while the best collision and TTC scores are closer between the two groups. Comfort is saturated for both groups. These results place the leading privileged and end-to-end methods in a similar performance range, while showing that direct scene states provide a moderate rather than decisive advantage on NAVSIM.

Table 9: Complete NAVSIM results with front-view camera input. Bold marks the highest score in each column.

Method	PDMS ↑	NAC ↑	DAC ↑	EP ↑	TTC ↑	Comfort ↑
PLUTO	**89.49**	**97.86**	**98.16**	**83.95**	**94.20**	99.96
Diffusion Planner	88.36	97.75	97.56	82.90	93.77	98.89
BEV-RGB	75.46	94.83	87.19	71.58	87.85	98.69
BEV-Mask	71.02	92.98	84.55	67.63	84.99	98.28
InstructDriver	77.19	95.71	87.55	72.19	89.55	**99.98**

Without camera input, PLUTO and Diffusion Planner form the leading group on NAVSIM, with only 0.37 PDMS between them. Their component profiles nevertheless differ: Diffusion Planner obtains the highest NAC, DAC, and TTC, whereas PLUTO achieves higher ego progress and the best overall PDMS. InstructDriver and BEV-RGB occupy the middle group, while BEV-Mask remains lower across most safety and progress metrics. Comfort is nearly saturated for all five planners and therefore provides limited separation among them.

The overall NAVSIM ranking, PLUTO, Diffusion Planner, InstructDriver, BEV-RGB, and BEV-Mask, matches the Bench2Drive DS ranking. Camera augmentation does not change this ordering on either benchmark, but its numerical effect differs. On NAVSIM, PLUTO and Diffusion Planner improve by 1.61 and 0.85 PDMS, and BEV-Mask improves by 6.88 PDMS, whereas BEV-RGB and InstructDriver decrease slightly. Bench2Drive shows little or negative change for most planners. The repeated ordering indicates a stable coarse performance trend across the two evaluations, while the different camera gains show that additional visual information interacts with the dataset and the planner representation rather than providing a uniform benefit.

E.2 Bench2Drive Scenario-Level Results

The 220-route Bench2Drive evaluation contains five routes for each of 44 atomic scenario types. Following the official SR criterion, a route is successful only if its status is Completed or Perfect and it contains no infraction other than a minimum-speed infraction. Figure 7 reports the scenario success rate in every cell. Figure 8 reports the mean DS of the corresponding routes.

The heatmaps reveal a broad agreement in scenario difficulty. Across planner families, highway and interurban flow, highway cut-in, merger-into-slow-traffic, sequential-lane-change, and vanilla-turn scenarios tend to receive higher scores, whereas accident or construction setups, non-signalized junction interactions, parking exits, and emergency-yield cases remain challenging for most methods. Although individual cells differ across planners, the mean-DS and success-rate maps exhibit broadly similar trends in scenario difficulty.

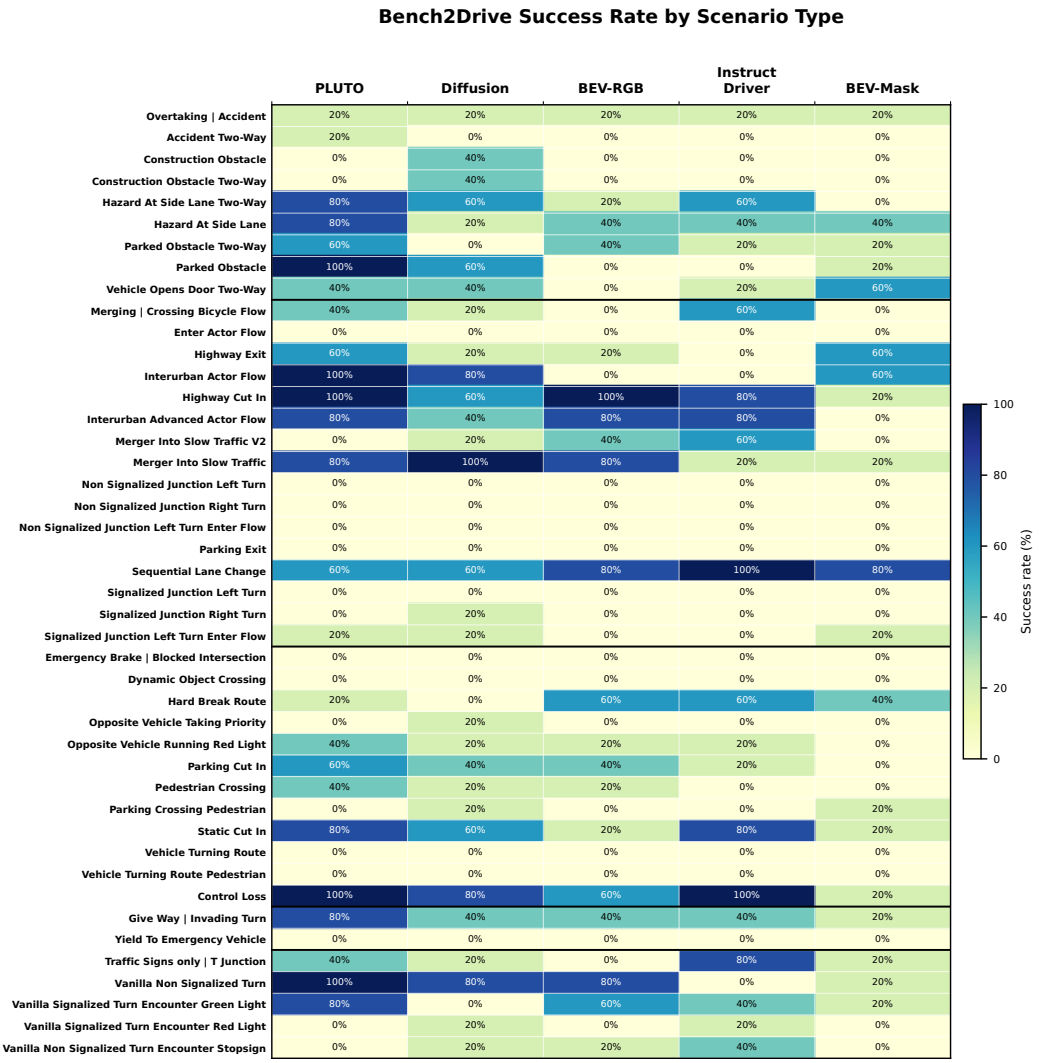

Figure 7: Scenario-level success rates on Bench2Drive. Each scenario contains five routes; both cell annotations and color encode the corresponding success rate. Thick horizontal lines separate the primary display groups.

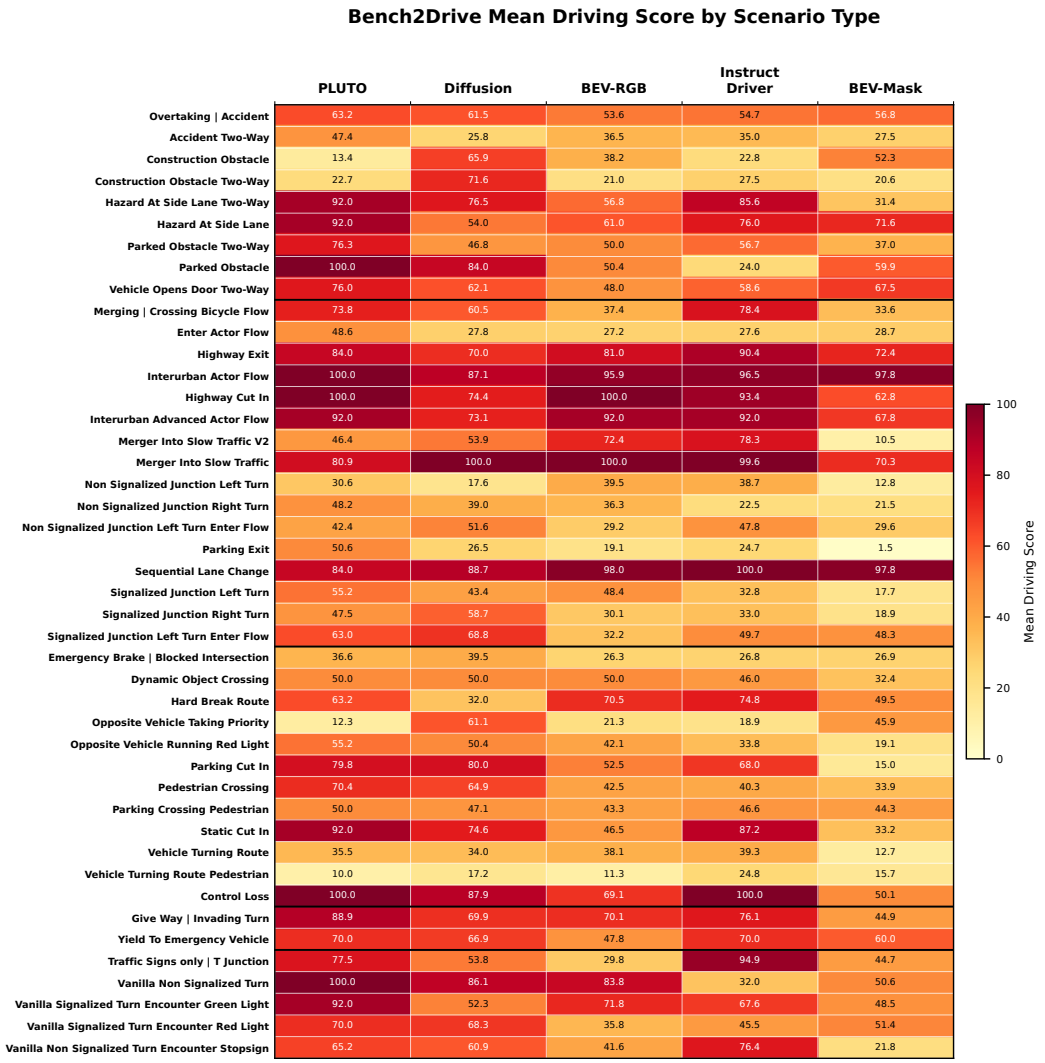

Figure 8: Mean Driving Score by Bench2Drive scenario type. Each cell averages DS over the five routes of the corresponding scenario and therefore retains both route completion and infraction penalties.