DriveSuprim: Towards Precise Trajectory Selection for End-to-End Planning

Wenhao Yao[1, 2], **Zhenxin Li**[1, 2], **Shiyi Lan**[3], **Zi Wang**[3],
Xinglong Sun[3], **Jose M. Alvarez**[3], **Zuxuan Wu**[1, 2*]
[1]Shanghai Key Lab of Intell. Info. Processing, School of CS, Fudan University
[2]Shanghai Collaborative Innovation Center of Intelligent Visual Computing
[3]NVIDIA
{whyao23, lizx23}@m.fudan.edu.cn, zxwu@fudan.edu.cn

Abstract

Autonomous vehicles must navigate safely in complex driving environments. Imitating a single expert trajectory, as in regression-based approaches, usually does not explicitly assess the safety of the predicted trajectory. Selection-based methods address this by generating and scoring multiple trajectory candidates and predicting the safety score for each. However, they face optimization challenges in precisely selecting the best option from thousands of candidates and distinguishing subtle but safety-critical differences, especially in rare and challenging scenarios. We propose **DriveSuprim** to overcome these challenges and advance the selection-based paradigm through a coarse-to-fine paradigm for progressive candidate filtering, a rotation-based augmentation method to improve robustness in out-of-distribution scenarios, and a self-distillation framework to stabilize training. **DriveSuprim** achieves state-of-the-art performance, reaching 93.5% PDMS in NAVSIM v1 and 87.1% EPDMS in NAVSIM v2 without extra data, with 83.02 Driving Score and 60.00 Success Rate on the Bench2Drive benchmark, demonstrating superior planning capabilities in various driving scenarios.

Code — https://github.com/William-Yao-2000/DriveSuprim
Extended version — https://arxiv.org/abs/2506.06659

Introduction

End-to-end autonomous driving has traditionally relied on regression-based approaches that predict a single trajectory to mimic expert behavior (Jiang et al. 2023; Hu et al. 2023; Wang et al. 2023) . While regression is a common approach, it fundamentally lacks the ability to evaluate multiple alternatives in safety-critical scenarios where subtle trajectory differences can significantly impact outcomes.

In recent years, selection-based methods (Chen et al. 2024; Li et al. 2024c,a; Wang et al. 2025) have clearly outperformed regression approaches. The key is their capability to generate and evaluate diverse trajectory candidates using comprehensive safety metrics such as collision risk and driving rule compliance (Dauner et al. 2024). This explicit comparison enables the system to select the safest and most appropriate trajectory from multiple alternatives, addressing safety-critical issues that regression-based methods cannot address.

Top-K	NC↑	DAC↑	EP↑	TTC↑	C↑	PDMS↑
1	99.0	98.7	86.5	96.2	100	91.9
4	99.4	99.6	89.6	98.0	100	94.5
16	99.7	99.8	92.0	99.1	100	96.1
256	100	100	97.1	99.9	100	98.7
Human	100	100	87.5	100	99.9	94.8

Table 1: PDM score of the best trajectory in the top-K candidates on ranked predicted scores.

Our oracle study in Tab 1 demonstrates the substantial potential of selection-based methods: when making ideal optimal selection, these approaches can even surpass human demonstrations in NAVSIM (Dauner et al. 2024) safety-critical metrics. This performance ceiling highlights why selection-based planning has become the preferred paradigm for autonomous driving systems requiring robust safety guarantees.

However, selection-based methods still face three critical limitations, which make them difficult to reach the ideal perfect trajectory selection displayed in Tab 1 and limit the model performance. First, selection-based methods struggle to distinguish the optimal trajectory from similar but sub-optimal alternatives. During training, the model encounters thousands of trajectory candidates where the vast majority are unsafe or impractical ("easy negatives", the red trajectory in Fig 1). These easy-to-reject options dominate the training process and gradient, causing the model to focus primarily on avoiding obviously incorrect choices. In contrast, the model receives insufficient supervision to select the most suitable trajectory from reasonable-looking trajectories with subtle but important differences ("hard negatives", the orange trajectory in Fig 1). The overwhelming number of obvious negative examples hinders the model's ability to develop fine-grained discrimination capability, which is crucial for selecting optimal trajectories when presented with multiple plausible options, to avoid route deviation or collision.

Second, selection-based methods suffer from directional bias in trajectory distribution. This bias manifests as an imbalance in training data that, while reflecting real-world driving patterns where straight driving predominates, leads to models that perform relatively poorly in turning scenarios. Training with such imbalanced data naturally results in models that

*Corresponding author.

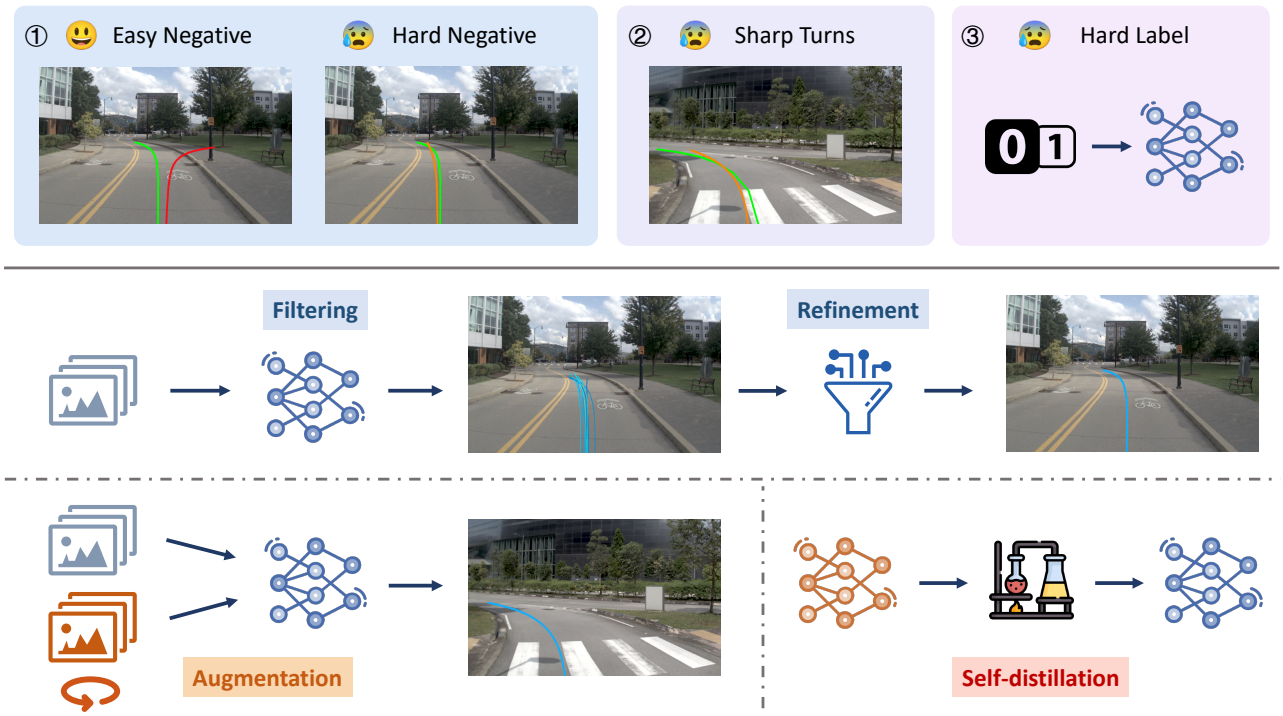

Figure 1: Overall pipeline of our method. Selection-based methods struggle to distinguish suboptimal trajectories, perform poorly in turning, and utilize hard binary labels in training. DriveSuprim introduces coarse-to-fine refinement and a rotation-based data augmentation method with self-distillation to address these weaknesses. The green trajectory is the ground-truth trajectory, and the red and orange trajectories are obviously unsafe and seemingly correct trajectory candidates in the trajectory vocabulary.

excel at straight-line driving but struggle with turns and complex maneuvers. Even advanced autonomous driving datasets like NAVSIM exhibit this limitation. We find that only 18% of ground-truth trajectories in NAVSIM involve turns exceeding 30 degrees. While this distribution may reflect typical driving patterns, it creates a significant challenge for learning models, which require sufficient examples of all maneuver types to develop robust capabilities. This directional bias significantly impairs the model's ability to select the most precise large-angle turning trajectories, particularly in navigation-critical scenarios where turns are essential.

Third, selection-based methods typically rely on binary classification for safety-related decisions, labeling trajectories as "safe" or "unsafe" based on specific thresholds for collision risk or rule compliance. This binary approach creates hard decision boundaries where trajectories just above or below a safety threshold could be treated entirely differently. Slight variations in score could suddenly thoroughly flip a trajectory label from being selected to being rejected. As a result, models risk becoming overly sensitive to minor changes in trajectory features, which causes inconsistent behavior.

We present **DriveSuprim**, a novel method that tackles these three critical challenges in selection-based trajectory prediction and makes more nuanced and precise trajectory selection. Our contributions include:

- We propose a coarse-to-fine refinement method that addresses the challenge of distinguishing between similar trajectories. Our method filters promising candidates and applies fine-grained scoring to the most challenging options, significantly improving discrimination between similar but subtly different trajectories.
- We propose an integrated training pipeline combining rotation-based data augmentation with self-distillation to address directional bias and hard decision boundaries. Our approach synthesizes challenging turning scenarios and leverages teacher-generated soft pseudo-labels, effectively balancing trajectory distributions and realizing better optimization for training a more robust model.
- DriveSuprim achieves state-of-the-art performance on the NAVSIM and Bench2Drive benchmark, demonstrating the effectiveness of our model in handling challenging driving scenarios. DriveSuprim significantly outperforms the previous methods by 3.6% and 1.5% on NAVSIMv1 and NAVSIMv2 without introducing other training data. On the Bench2Drive benchmark, our method achieves 83.02 and 60.00 on driving score and success rate.

Related Works

End-to-end Planning

Autonomous driving has traditionally relied on modular pipelines that separate perception from planning. However, UniAD (Hu et al. 2023) highlights several limitations of this approach, including information loss and error propagation. To address these challenges, end-to-end driving methods (Chen et al. 2019; Chen, Koltun, and Krähenbühl 2021; Chitta

et al. 2022; Hu et al. 2023; Jiang et al. 2023; Li et al. 2024d,c; Wang et al. 2025) unify the perception-to-planning pipeline within a single optimizable network. They process raw sensor inputs and directly output driving trajectories. While some methods (Chen, Koltun, and Krähenbühl 2021; Zhang et al. 2021) use reinforcement learning (RL) to learn through interaction with simulated environments, the majority adopt imitation learning (IL), training from expert demonstrations without interaction. Most IL-based approaches (Chitta et al. 2022; Hu et al. 2023; Jiang et al. 2023; Li et al. 2025a; Liao et al. 2025) generate a single trajectory using regression or diffusion-based methods to mimic expert behavior.

More recently, selection-based methods (Chen et al. 2024; Li et al. 2024c,a; Wang et al. 2025) have emerged. These models evaluate a diverse set of candidate trajectories by scoring them on safety-focused metrics (e.g., PDM scores (Dauner et al. 2024)). A prominent example is Hydra-MDP (Li et al. 2024c), which employs multiple rule-based teachers and distills them into the planner to create diverse trajectory candidates tailored to different evaluation metrics.

Our proposed model also falls under the selection-based paradigm. However, unlike prior methods that perform a single-shot selection from a fixed candidate set—probably leading to suboptimal decisions—we introduce a coarse-to-fine selection and refinement strategy. This approach significantly improves selection precision by progressively narrowing down the trajectory set to the most optimal candidates.

Iterative & Multi-stage Refinement

Iterative refinement has been widely adopted to improve results in optical flow (Ilg et al. 2017; Hui, Tang, and Loy 2018; Teed and Deng 2020; Xu et al. 2022) and motion estimation (Sun, Harley, and Guibas 2024; Zheng et al. 2023). A common strategy in these works is to iteratively propagate features or trajectory estimates through a shared module to progressively refine the predictions. Inspired by this, iterative refinement is also applied in object detection (Zhu et al. 2020; Cai and Vasconcelos 2018) to improve performance. For instance, in Deformable DETR, each decoder layer refines bounding boxes based on predictions from the previous layer.

We similarly adopt a multi-stage refinement strategy to improve trajectory selection accuracy. However, rather than repeatedly updating fixed-dimensional features, we implement selection from a fixed vocabulary of candidate trajectories. After selection, the search space is progressively narrowed to a more precise subset, improving the final prediction quality.

Augmentation for Enhanced Robustness

Robustness has long been a critical focus in computer vision research (Ganin et al. 2016; Croce et al. 2020; Rebuffi et al. 2021; Sun et al. 2022). Early studies demonstrated that image models are highly sensitive to minor domain shifts (Azulay and Weiss 2019) and adversarial perturbations (Szegedy 2013; Goodfellow, Shlens, and Szegedy 2014). For example, MNIST-C (Mu and Gilmer 2019) introduces 15 distinct corruption types to benchmark model performance against diverse failure modes. Motivated by these insights, several methods (Rusak et al. 2020; Mintun, Kirillov, and Xie 2021; Kar et al. 2022) utilize corruption-based augmentations, such as adding Gaussian and speckle noise, to enhance robustness. Inspired by these methods, our study explores the use of similar corruption-based augmentation techniques specifically for end-to-end driving models. We introduce targeted perturbations tailored to autonomous driving scenarios, addressing critical domain shifts—particularly the overrepresentation of straightforward driving trajectories—which pose challenges for scenarios involving complex maneuvers such as turns.

Methods

We introduce our method in this section. Firstly, we introduce preliminaries about the end-to-end planning and the selection-based method. Next, we introduce our proposed coarse-to-fine selection paradigm, rotation-based data augmentation method and self-distillation framework.

Preliminaries

End-to-End Planning & Selection-based Planning In autonomous driving, the end-to-end planning requires the planning system to output a future trajectory T based on input sensor data, like RGB image or Lidar point cloud:

$$T = \text{Planner}\left(Img, Lidar\right), \tag{1}$$

where the trajectory T can be represented as a sequence of vehicle locations $(u_1, u_2, ..., u_l)$ or a sequence of controller actions $(a_1, a_2, ..., a_l)$, and l denotes the sequence length.

Among the end-to-end planning methods, the selection-based paradigm predefines a trajectory vocabulary $\{\tau_i\}_{i=1}^{N}$ covering N planning trajectories. Given a specific driving scenario, the quality of each trajectory is evaluated by several metrics, like the l_2 distance to the human teacher trajectory, or metrics considering driving safety and traffic rule adherence. The model learns a scorer that generates trajectory scores $\{s_i\}_{i=1}^{N}$ revealing trajectory quality. The trajectory τ_k with the highest score is chosen as the predicted result in inference.

Coarse-to-Fine Trajectory Selection

DriveSuprim proposes a coarse-to-fine trajectory selection paradigm comprising coarse filtering and fine-grained scoring, improving model capability in distinguishing hard negative trajectories. As shown in Fig 2, in the coarse filtering stage, the model selects several trajectory candidates based on predicted scores, similar to classic selection-based approaches. The fine-grained scoring stage then produces more accurate scores for the filtered trajectories.

Coarse Filtering The coarse filtering stage scores all trajectories in the vocabulary and filters a smaller set of candidates for the next stage. Here we apply the same strategy as the previous selection-based method (Li et al. 2024c). The trajectory feature cross-attends with the image feature to extract planning-related information, then several prediction heads are applied on the refined trajectory feature to regress the normalized l_2 distance to the ground-truth human trajectory and the rule-based metric scores:

$$\mathcal{E}_{\text{img}} = \text{Enc}_{\text{i}}(I), f_j = \text{Enc}_{\text{t}}(\tau_j), \tag{2}$$

$$g_j = \text{TransDec}(\mathcal{E}_{\text{img}}, f_j) \tag{3}$$

$$s_j^{(m)} = \text{Sigmoid}\left(\text{head}^{(\text{m})}(g_j)\right), \tag{4}$$

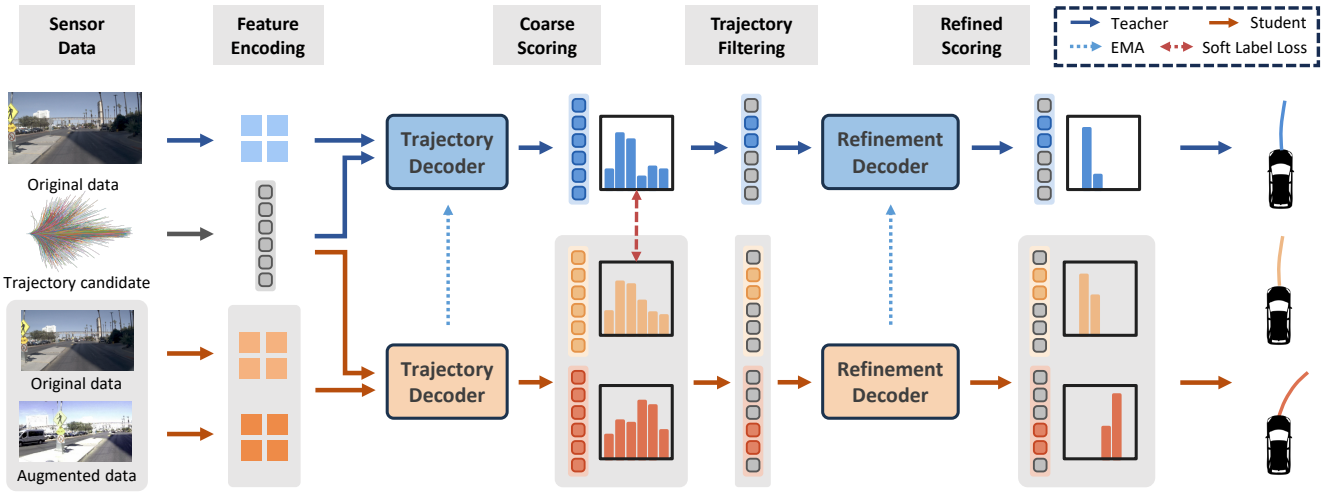

Figure 2: Model architecture. DriveSuprim adopts a coarse-to-fine paradigm to better distinguish hard negatives. According to the scoring distribution, the Trajectory Decoder filters potential candidates, and the Refinement Decoder further outputs fine-grained trajectory scores. The model introduces rotation-based augmented data to ease the directional bias and applies a self-distillation framework for stable training. The teacher outputs serve as soft labels for auxiliary supervision for the student.

where $\mathrm{Enc_i}$ and $\mathrm{Enc_t}$ are image encoder and trajectory encoder, I and $\mathcal{E}_{\mathrm{img}}$ are the input image and image feature, τ_j and f_j denote the trajectory and the encoded trajectory feature, $\mathrm{TransDec}$ denotes the Trajectory Decoder, which is a Transformer decoder (Vaswani et al. 2017), g_j denotes the refined trajectory feature, $\mathrm{head}^{(\mathrm{m})}$ denotes the prediction head of evalutation metric m, $s_j^{(m)}$ denotes the prediction score of trajectory τ_j on metric m.

At the end of the coarse filtering stage, each trajectory τ_j corresponds to a score s_j revealing its quality on end-to-end planning. We select the trajectories with top-k scores as the filtered trajectories $T_{\mathrm{filter}} = \{\tau_j \mid j \in \mathcal{I}_{\mathrm{top}k}\}$, where $\mathcal{I}_{\mathrm{top}k} = \mathrm{argsort}_k(\{s_j\}_{j=1}^N)$, and the refined features $G_{\mathrm{filter}} = \{g_j \mid j \in \mathcal{I}_{\mathrm{top}k}\}$ are utilized for fine-grained fitting.

Fine-grained Scoring In this stage, a Transformer decoder similar to the first stage is applied to make fine-grained scoring and further distinguish the trajectories in the first-stage filtered candidates, which contain a large proportion of hard negatives. Specifically, we obtain the refined score from the l-th decoder layer output feature, and optimize with the ground truth trajectory score:

$$\{h_{j,l}\}_{l=1}^{n_{\mathrm{ref}}} = \mathrm{RefineDec}\left(\mathcal{E}_{\mathrm{img}}, g_j\right) \tag{5}$$

$$s_{j,l}^{(m)} = \mathrm{Sigmoid}\left(\mathrm{head}^{(\mathrm{m})}\left(h_{j,l}\right)\right) \tag{6}$$

where $\mathrm{RefineDec}$ is a Transformer decoder with n_{ref} layers, $h_{j,l}$ is the refined feature of τ_j from the l-th layer of the Refinement Transformer, $s_{j,l}^{(m)}$ denotes the score of τ_j on metric m output by the l-th decoder layer. The output trajectory τ_k is selected based on the highest last-layer output score $s_{k,n_{\mathrm{ref}}}^{(m)}$.

The loss for the coarse-to-fine module on the original dataset is represented as:

$$L_{\mathrm{ori}} = L_{\mathrm{coarse}} + L_{\mathrm{refine}}, \tag{7}$$

where L_{coarse} and L_{refine} respectively train the coarse filtering stage and the fine-grained scoring stage. L_{coarse} consists of an imitation loss L_{imi} (Li et al. 2024c,a) and a binary cross-entropy between the predicted trajectory score and the ground truth metric score:

$$L_{\mathrm{coarse}} = L_{\mathrm{imi}} + \sum_{m,i} \mathrm{BCE}(s_i^{(m)}, y_i^{(m)}), \tag{8}$$

where $y_i^{(m)}$ and $s_i^{(m)}$ are the ground-truth metric score and the predicted score of trajectory τ_i on metric m. L_{refine} is similar to L_{coarse}, while it only considers the filtered trajectories T_{filter} rather than the entire vocabulary, and is applied on each decoder layer output.

Rotation-based Data Augmentation

To mitigate the data imbalance, we introduce an end-to-end rotation-based augmentation pipeline, where a 2D horizontal view transformation is applied to the sensor input data to simulate the ego-vehicle rotation in the 3D space. This approach easily synthesizes more challenging scenarios and diversifies the driving scenarios, enabling the model to precisely select trajectories regardless of the vehicle orientation.

As shown in Fig 3, for each scenario, we sample a random angle θ from a uniform distribution $U[-\Theta, \Theta]$, where Θ is the angle boundary. The positive angle indicates the leftward rotation of the ego vehicle. Camera images corresponding to the original field of view (FOV) along with images from two extended views are concatenated to simulate a "pseudo panoramic view", then the input image is cropped from the concatenated image according to a shifting window based on θ. For example, for a 1-camera FOV input setting, we choose these three cameras as our input: f (front), l_0 (front-left), and r_0 (front-right), where f corresponds to the original field of view, and l_0 and r_0 correspond to the extended view.

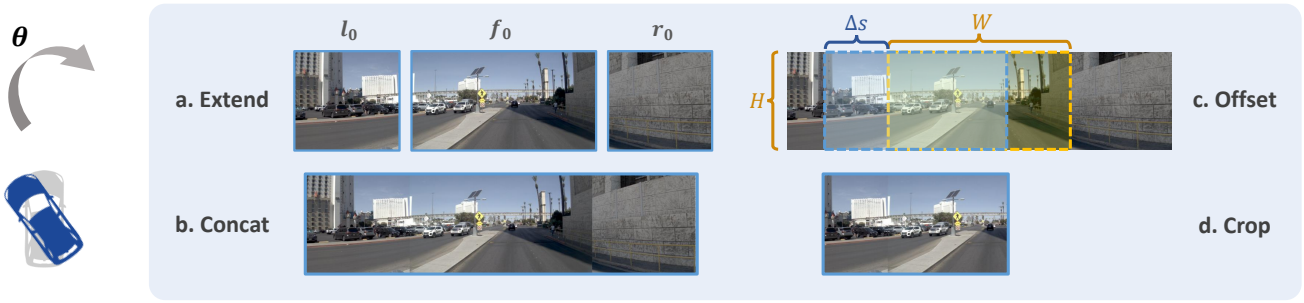

Figure 3: Rotation demonstration for input image in a 1-camera FOV setting. Our rotation-based data augmentation method generates the rotated image through horizontal shifting and cropping.

In the synthesized rotated scenario, the ground truth human trajectory $(u_1, u_2, \ldots, u_l)$ is generated by a trivial rotation approach: under a specific rotation angle θ, each location u_j of the human trajectory T_h applies a 2D-rotation transformation surrounding the initial vehicle position u_0, and the rotation angle is $-\theta$, ensuring the world coordinate of the trajectory remains unchanged. Given augmented camera views and the prediction of our model, we calculate a loss L_{aug} for training, which has the same formulation as L_{ori}.

Self-distillation with Soft-labeling

Instead of training the selection-based model to fit a hard decision boundary of trajectory evaluation, which can impair training stability, we propose a self-distillation framework with teacher-generated soft labels to stabilize model training. The self-distillation framework consists of a teacher and a student model, both sharing the same architecture. The student is updated via standard gradient descent, whereas the teacher is updated using an exponential moving average (EMA) of the student's parameters.

During training, the student receives noisier input, including both original and augmented data to calculate L_{ori} and L_{aug}, as shown in Fig 2. The teacher only receives original data to generate scores served as soft labels, where a clipping threshold δ_m is introduced to control the gap between the teacher output and the ground-truth:

$$\hat{y}_i^{(m)} = y_i^{(m)} + \text{clip}\left(s_{i,\text{teacher}}^{(m)} - y_i^{(m)}, -\delta_m, \delta_m\right) \quad (9)$$

$$L_{\text{soft}} = L_{\text{imi-soft}} + \sum_{m,i} \text{BCE}(s_i^{(m)}, \hat{y}_i^{(m)}), \quad (10)$$

where $s_{i,\text{teacher}}^{(m)}$ is the score of trajectory i on metric m predicted by the teacher, $y_i^{(m)}$ is the ground-truth score of trajectory i on metric m, $\text{clip}(\cdot, \alpha, \beta)$ denotes clipping the input value to the interval $[\alpha, \beta]$, and $\hat{y}_i^{(m)}$ is the soft label. $L_{\text{imi-soft}}$ is a soft version of L_{imi}, where the human trajectory is shifted toward the teacher model's output trajectory for at most 1 meter, and is adopted to calculate the imitation loss. The overall training loss of the student model is:

$$L = L_{\text{ori}} + L_{\text{aug}} + L_{\text{soft}} \quad (11)$$

During inference, the teacher model is utilized to output the planning trajectories.

Experiments

In this section, we first introduce our implementation details. Next, we show the superior performance of DriveSuprim on NAVSIM and Bench2Drive, and ablation studies on NAVSIM are listed to validate the effectiveness of the proposed modules. Finally, we produce some visualization results to intuitively show the advantages of our method.

Implementation Details

Dataset and Metrics We conduct experiments mainly on the NAVSIM (Dauner et al. 2024) and Bench2Drive (Jia et al. 2024) benchmark. NAVSIM includes two evaluation metrics, leading to NAVSIM v1 and NAVSIM v2. The evaluation metric of NAVSIM v1 is the PDM Score (PDMS). Each predicted trajectory is sent to a simulator to collect different rule-based metrics, which are weighted aggregated to get the final PDMS. NAVSIM v1 includes 5 metrics: no collisions (NC), drivable area compliance (DAC), ego progress (EP), time-to-collision (TTC), and comfort (C). NAVSIM v2 further introduces 4 extended metrics, including driving direction compliance (DDC), traffic light compliance (TLC), lane keeping (LK) and extended comfort (EC). Aggregating all these subscores leads to the EPDMS metric.

Bench2Drive is a closed-loop benchmark evaluating 220 short routes in CARLA v2. The evaluation metrics include Success Rate, Driving Score, Efficiency and Comfortness. More details about datasets and metrics are in Appendix A.

Model Details and Training Details We conduct our methods on three different backbones as the image encoder Enc_i, including ResNet34 (He et al. 2016), VoVNet (Lee et al. 2019), and ViT-Large (Dosovitskiy et al. 2021). A 2-layer MLP is leveraged as Enc_t to encode each trajectory in the vocabulary. The trajectory decoder TransDec and refinement decoder RefineDec are both 3-layer Transformer Decoders with 256 hidden dimensions. The MLP prediction head $\text{head}^{(m)}$ predicts the normalized l_2 distance to the human ground-truth trajectory, and each subscore in NAVSIM except for EC. The number of trajectories in the vocabulary and filtered trajectories of the coarse filtering stage are set to 8192 and 256. We adopt a 3-camera FOV setting, with images from l_0, f, and r_0 cameras as the input. The rotation angle boundary Θ is set to $\pi/6$ in the augmentation pipeline. The threshold δ_m in soft-labeling is set to 0.15.

Method	Backbone	NC ↑	DAC ↑	EP ↑	TTC ↑	C ↑	PDMS ↑
Human Agent	—	100	100	87.5	100	99.9	94.8
Transfuser (2022)	ResNet34	97.7	92.8	79.2	92.8	100	84.0
UniAD (2023)	ResNet34	97.8	91.9	78.8	92.9	100	83.4
VADv2 (2024)	ResNet34	97.9	91.7	77.6	92.9	100	83.0
LAW (2024b)	ResNet34	96.4	95.4	81.7	88.7	99.9	84.6
DRAMA (2024)	ResNet34	98.0	93.1	80.1	94.8	100	85.5
Hydra-MDP (2024c)	ResNet34	98.3	96.0	78.7	94.6	100	86.5
DiffusionDrive (2025)	ResNet34	98.2	96.2	82.2	94.7	100	88.1
DriveSuprim	ResNet34	97.8	97.3	86.7	93.6	100	**89.9 (+1.8)**
Hydra-MDP (2024c)	V2-99	98.4	97.8	86.5	93.9	100	90.3
DriveSuprim	V2-99	98.0	98.2	90.0	94.2	100	**92.1 (+1.9)**
Hydra-MDP (2024c)	ViT-L	98.4	97.7	85.0	94.5	100	89.9
DriveSuprim	ViT-L	98.6	98.6	91.3	95.5	100	**93.5 (+3.6)**

Table 2: Evaluation on NAVSIM v1. Results are grouped by backbone types.

Method	Backbone	NC ↑	DAC ↑	DDC ↑	TL ↑	EP ↑	TTC ↑	LK ↑	HC ↑	EC ↑	EPDMS ↑
Human Agent	—	100	100	99.8	100	87.4	100	100	98.1	90.1	90.3
Ego Status MLP	—	93.1	77.9	92.7	99.6	86.0	91.5	89.4	98.3	85.4	64.0
Transfuser (2022)	ResNet34	96.9	89.9	97.8	99.7	87.1	95.4	92.7	98.3	87.2	76.7
HydraMDP++ (2024a)	ResNet34	97.2	97.5	99.4	99.6	83.1	96.5	94.4	98.2	70.9	81.4
DriveSuprim	ResNet34	97.5	96.5	99.4	99.6	88.4	96.6	95.5	98.3	77.0	**83.1 (+1.7)**
HydraMDP++ (2024a)	V2-99	98.4	98.0	99.4	99.8	87.5	97.7	95.3	98.3	77.4	85.1
DriveSuprim	V2-99	97.8	97.9	99.5	99.9	90.6	97.1	96.6	98.3	77.9	**86.0 (+0.9)**
HydraMDP++ (2024a)	ViT-L	98.5	98.5	99.5	99.7	87.4	97.9	95.8	98.2	75.7	85.6
DriveSuprim	ViT-L	98.4	98.6	99.6	99.8	90.5	97.8	97.0	98.3	78.6	**87.1 (+1.5)**

Table 3: Evaluation on NAVSIM v2. Results are grouped by backbone types.

Method	DS ↑	SR ↑	Eff. ↑	Comf. ↑
DriveAdapter (2023)	64.22	33.08	70.22	16.01
Hydra-NeXt (2025b)	73.86	50.00	197.76	20.68
Orion (2025)	77.74	54.62	151.48	17.38
AutoVLA (2025)	78.84	57.73	146.93	**39.33**
DriveSuprim	**83.02**	**60.00**	**238.78**	20.89

Table 4: Evaluation on Bench2Drive.

We train our model on 8 NVIDIA A100. We use Adam for model training, the batch size on a GPU is 8, and the learning rate is set to 7.5×10^{-5}. More model details and training details are shown in Appendix B.

Main Results

Result on NAVSIM Tab 2 shows the performance of DriveSuprim on the NAVSIM benchmark. our method reaches 89.9% PDMS with the ResNet34 backbone, surpassing DiffusionDrive by 1.8%. Moreover, DriveSuprim with a stronger Vit-Large backbone can reach 93.5% PDMS. Results in Table 3 show that our model can also reach the SOTA result on the more challenging NAVSIM v2 benchmark. On the EPDMS metric, DriveSuprim surpasses previous SOTA methods by 1.7%, 0.9%, and 1.5%, respectively. DriveSuprim reaches 27.2 FPS and 12.5 FPS with ResNet34 and ViT-Large backbone, achieving real-time planning capability.

Result on Bench2Drive Evaluation on Bench2Drive shows the close loop planning capability of our method. Based on the CARLA-Garage dataset (Jaeger, Chitta, and Geiger 2023; Sima et al. 2024) and the TF++ framework (Jaeger, Chitta, and Geiger 2023), our approach adopts the two-stage trajectory prediction for longitudinal control. Tab 4 shows that DriveSuprim achieves 83.02 and 60.00 scores on Driving Score and Success Rate, with 238.78 Efficiency and 20.89 Comfortness, outperforming the previous SOTA significantly.

Ablation Studies

Ablation on Different Modules We conduct ablation studies on the modules and approaches we propose, and the result is shown in Tab 5. Compared to baseline, adopting multi-stage refinement leads to 1.0% performance gain on PDMS, and the improvements further gained by augmented data and the self-distillation framework are 0.3% and 0.4%.

Effectiveness of Coarse-to-fine Selection We evolve our model from conventional single-stage selection to coarse-to-fine selection, to eliminate the effect of parameter number increase and validate the effectiveness of the coarse-to-fine trajectory selection paradigm. Tab 6 reveals that increasing

Multi-stage	Aug Data	Self-distill	NC ↑	DAC ↑	DDC ↑	TL ↑	EP ↑	TTC ↑	LK ↑	HC ↑	EC ↑	EPDMS ↑
✗	✗	✗	97.2	97.5	99.4	99.6	83.1	96.5	94.4	98.2	70.9	81.4
✓	✗	✗	97.5	96.4	99.1	99.6	87.0	96.4	95.3	98.2	75.0	82.4
✓	✓	✗	96.9	96.9	99.4	99.6	87.9	96.0	95.5	98.3	76.5	82.7
✓	✓	✓	97.5	96.5	99.4	99.6	88.4	96.6	95.5	98.3	77.0	83.1

Table 5: Ablation study on different proposed modules. "Multi-stage" denotes using coarse-to-fine selection, "Aug Data" denotes introducing rotation-based augmentation data, and "Self-distill" denotes adopting self-distillation.

	EPDMS ↑
Single-stage	81.4
+ 6 layer decoder	81.7
+ Layer-wise scoring	81.9
+ Trajectory filtering	82.4

Table 6: Ablation study of the evolution from single-stage selection to coarse-to-fine selection.

Method	NAVTEST_l	NAVTEST_f	NAVTEST_r
Hydra-MDP++	68.7	87.2	77.7
DriveSuprim	**71.6 (+2.9)**	88.1 (+0.9)	**79.7 (+2.0)**

Table 7: EPDMS on three dataset splits. NAVTEST_l, NAVTEST_f, and NAVTEST_r involve left-turning scenarios, near-forward scenarios, and right-turning scenarios.

the number of decoder layers from 3 to 6 brings only 0.3% improvement in EPDMS, while introducing layer-wise scoring and trajectory filtering leads to a more substantial 0.7% gain, proving that the performance boost comes from the coarse-to-fine mechanism rather than model size increase.

Ablation on Refinement Settings and Soft Label Threshold We further conduct comprehensive ablation studies on the refinement approach and soft-labeling. Utilizing a 3-layer refinement Transformer decoder and filtering 256 trajectories for the refinement stage leads to the best result. For the teacher soft label, choosing 0.15 as the threshold leads to the best EPDM score. Results are shown in Appendix C.

Visualization and Analysis

Visualization Results Fig 4 shows qualitative visualization results on NAVSIM comparing DriveSuprim with the selection-based approach Hydra-MDP++. The images in the first and second columns are the results of Hydra-MDP++ and DriveSuprim. The first row illustrates a challenging overtaking scenario near a crossroad, where DriveSuprim correctly overtakes while Hydra-MDP++ mistakenly turns left and risks collision. In the second row, DriveSuprim outperforms in a sharp turn with smooth and accurate trajectories. These results demonstrate DriveSuprim not only performs well in challenging scenarios by choosing precise trajectories, but also excels in handling sharp turns with high accuracy. More visualization results are shown in Appendix D.

Figure 4: Visualization results across various challenging scenarios. In each example, the green trajectory represents the ground truth from the human expert, the red trajectory is generated by Hydra-MDP++, and the blue trajectory is produced by DriveSuprim.

Superior Performance on Turning Scenarios Tab. 7 demonstrates the superior performance of DriveSuprim in turning scenarios. We divide the test dataset into three subsets based on the turning angle of the ground-truth trajectories: NAVTEST_l (left turns exceeding 30 degrees), NAVTEST_f (near-forward trajectories), and NAVTEST_r (right turns exceeding 30 degrees). Performance improvements of DriveSuprim are more pronounced in turning scenarios than in near-straight ones, highlighting enhanced ability to handle turning maneuvers.

Trajectory Distribution Comparison We illustrate the trajectory frequency distribution in Appendix D. Results show that in the original dataset, trajectories are predominantly concentrated in the forward or near-forward direction, while in the augmented dataset, trajectories across all directions appear with similar frequency.

Conclusion

We present DriveSuprim, a novel framework for end-to-end planning. By introducing coarse-to-fine selection and an integrated training pipeline with rotation-based data augmentation and soft-label self-distillation, DriveSuprim significantly enhances the model's ability to distinguish hard negatives and precisely select trajectories, and performs well in scenarios involving sharp turns. Extensive experiments on the NAVSIM and Bench2Drive benchmark demonstrate that our approach outperforms prior methods by a substantial margin.

Acknowledgments

This work was supported by National Natural Science Foundation of China (No. 62427819) and the Science and Technology Commission of Shanghai Municipality (No. 24511103100).

References

Azulay, A.; and Weiss, Y. 2019. Why do deep convolutional networks generalize so poorly to small image transformations? *Journal of Machine Learning Research*, 20(184): 1–25.

Cai, Z.; and Vasconcelos, N. 2018. Cascade R-CNN: Delving Into High Quality Object Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Chen, D.; Koltun, V.; and Krähenbühl, P. 2021. Learning to drive from a world on rails. In *ICCV*, 15590–15599.

Chen, D.; Zhou, B.; Koltun, V.; and Krähenbühl, P. 2019. Learning by Cheating. In *Conference on Robot Learning (CoRL)*.

Chen, S.; Jiang, B.; Gao, H.; Liao, B.; Xu, Q.; Zhang, Q.; Huang, C.; Liu, W.; and Wang, X. 2024. Vadv2: End-to-end vectorized autonomous driving via probabilistic planning. *arXiv preprint arXiv:2402.13243*.

Chitta, K.; Prakash, A.; Jaeger, B.; Yu, Z.; Renz, K.; and Geiger, A. 2022. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *TPAMI*, 45(11): 12878–12895.

Croce, F.; Andriushchenko, M.; Sehwag, V.; Debenedetti, E.; Flammarion, N.; Chiang, M.; Mittal, P.; and Hein, M. 2020. Robustbench: a standardized adversarial robustness benchmark. *arXiv preprint arXiv:2010.09670*.

Dauner, D.; Hallgarten, M.; Li, T.; Weng, X.; Huang, Z.; Yang, Z.; Li, H.; Gilitschenski, I.; Ivanovic, B.; Pavone, M.; Geiger, A.; and Chitta, K. 2024. NAVSIM: Data-Driven Non-Reactive Autonomous Vehicle Simulation and Benchmarking. In *NeurIPS*, volume 37, 28706–28719.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; Uszkoreit, J.; and Houlsby, N. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *ICLR*.

Fu, H.; Zhang, D.; Zhao, Z.; Cui, J.; Liang, D.; Zhang, C.; Zhang, D.; Xie, H.; Wang, B.; and Bai, X. 2025. Orion: A holistic end-to-end autonomous driving framework by vision-language instructed action generation. *arXiv preprint arXiv:2503.19755*.

Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; March, M.; and Lempitsky, V. 2016. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59): 1–35.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *CVPR*.

Hu, Y.; Yang, J.; Chen, L.; Li, K.; Sima, C.; Zhu, X.; Chai, S.; Du, S.; Lin, T.; Wang, W.; et al. 2023. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 17853–17862.

Hui, T.-W.; Tang, X.; and Loy, C. C. 2018. Liteflownet: A lightweight convolutional neural network for optical flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 8981–8989.

Ilg, E.; Mayer, N.; Saikia, T.; Keuper, M.; Dosovitskiy, A.; and Brox, T. 2017. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2462–2470.

Jaeger, B.; Chitta, K.; and Geiger, A. 2023. Hidden biases of end-to-end driving models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 8240–8249.

Jia, X.; Gao, Y.; Chen, L.; Yan, J.; Liu, P. L.; and Li, H. 2023. Driveadapter: Breaking the coupling barrier of perception and planning in end-to-end autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 7953–7963.

Jia, X.; Yang, Z.; Li, Q.; Zhang, Z.; and Yan, J. 2024. Bench2Drive: Towards Multi-Ability Benchmarking of Closed-Loop End-To-End Autonomous Driving. In *NeurIPS 2024 Datasets and Benchmarks Track*.

Jiang, B.; Chen, S.; Xu, Q.; Liao, B.; Chen, J.; Zhou, H.; Zhang, Q.; Liu, W.; Huang, C.; and Wang, X. 2023. Vad: Vectorized scene representation for efficient autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 8340–8350.

Kar, O. F.; Yeo, T.; Atanov, A.; and Zamir, A. 2022. 3d common corruptions and data augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 18963–18974.

Lee, Y.; Hwang, J.-w.; Lee, S.; Bae, Y.; and Park, J. 2019. An Energy and GPU-Computation Efficient Backbone Network for Real-Time Object Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Li, D.; Ren, J.; Wang, Y.; Wen, X.; Li, P.; Xu, L.; Zhan, K.; Xia, Z.; Jia, P.; Lang, X.; et al. 2025a. Finetuning Generative Trajectory Model with Reinforcement Learning from Human Feedback. *arXiv preprint arXiv:2503.10434*.

Li, K.; Li, Z.; Lan, S.; Liu, J.; Xie, Y.; zhizhong zhang; Wu, Z.; Yu, Z.; and Alvarez, J. M. 2024a. Hydra-MDP++: Advancing End-to-End Driving via Hydra-Distillation with Expert-Guided Decision Analysis.

Li, Y.; Fan, L.; He, J.; Wang, Y.; Chen, Y.; Zhang, Z.; and Tan, T. 2024b. Enhancing end-to-end autonomous driving with latent world model. *arXiv preprint arXiv:2406.08481*.

Li, Z.; Li, K.; Wang, S.; Lan, S.; Yu, Z.; Ji, Y.; Li, Z.; Zhu, Z.; Kautz, J.; Wu, Z.; et al. 2024c. Hydra-MDP: End-to-end Multimodal Planning with Multi-target Hydra-Distillation. *arXiv preprint arXiv:2406.06978*.

Li, Z.; Wang, S.; Lan, S.; Yu, Z.; Wu, Z.; and Alvarez, J. M. 2025b. Hydra-next: Robust closed-loop driving with open-loop training. *arXiv preprint arXiv:2503.12030*.

Li, Z.; Yu, Z.; Lan, S.; Li, J.; Kautz, J.; Lu, T.; and Alvarez, J. M. 2024d. Is ego status all you need for open-loop end-to-end autonomous driving? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14864–14873.

Liao, B.; Chen, S.; Yin, H.; Jiang, B.; Wang, C.; Yan, S.; Zhang, X.; Li, X.; Zhang, Y.; Zhang, Q.; and Wang, X. 2025. DiffusionDrive: Truncated Diffusion Model for End-to-End Autonomous Driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Mintun, E.; Kirillov, A.; and Xie, S. 2021. On interaction between augmentations and corruptions in natural corruption robustness. *Advances in Neural Information Processing Systems*, 34: 3571–3583.

Mu, N.; and Gilmer, J. 2019. Mnist-c: A robustness benchmark for computer vision. *arXiv preprint arXiv:1906.02337*.

Rebuffi, S.-A.; Gowal, S.; Calian, D. A.; Stimberg, F.; Wiles, O.; and Mann, T. A. 2021. Data augmentation can improve robustness. *Advances in Neural Information Processing Systems*, 34: 29935–29948.

Rusak, E.; Schott, L.; Zimmermann, R. S.; Bitterwolf, J.; Bringmann, O.; Bethge, M.; and Brendel, W. 2020. A simple way to make neural networks robust against diverse image corruptions. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, 53–69. Springer.

Sima, C.; Renz, K.; Chitta, K.; Chen, L.; Zhang, H.; Xie, C.; Beißwenger, J.; Luo, P.; Geiger, A.; and Li, H. 2024. Drivelm: Driving with graph visual question answering. In *European conference on computer vision*, 256–274. Springer.

Sun, J.; Zhang, Q.; Kailkhura, B.; Yu, Z.; Xiao, C.; and Mao, Z. M. 2022. Benchmarking robustness of 3d point cloud recognition against common corruptions. *arXiv preprint arXiv:2201.12296*.

Sun, X.; Harley, A. W.; and Guibas, L. J. 2024. Refining pre-trained motion models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 4932–4938. IEEE.

Szegedy, C. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*.

Teed, Z.; and Deng, J. 2020. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, 402–419. Springer.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L. u.; and Polosukhin, I. 2017. Attention is All you Need. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*.

Wang, X.; Zhu, Z.; Huang, G.; Chen, X.; Zhu, J.; and Lu, J. 2023. Drivedreamer: Towards real-world-driven world models for autonomous driving. *arXiv preprint arXiv:2309.09777*.

Wang, Z.; Lan, S.; Sun, X.; Chang, N.; Li, Z.; Yu, Z.; and Alvarez, J. M. 2025. Enhancing Autonomous Driving Safety with Collision Scenario Integration. *arXiv preprint arXiv:2503.03957*.

Xu, H.; Zhang, J.; Cai, J.; Rezatofighi, H.; and Tao, D. 2022. Gmflow: Learning optical flow via global matching. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 8121–8130.

Yuan, C.; Zhang, Z.; Sun, J.; Sun, S.; Huang, Z.; Lee, C. D. W.; Li, D.; Han, Y.; Wong, A.; Tee, K. P.; et al. 2024. Drama: An efficient end-to-end motion planner for autonomous driving with mamba. *arXiv preprint arXiv:2408.03601*.

Zhang, Z.; Liniger, A.; Dai, D.; Yu, F.; and Van Gool, L. 2021. End-to-end urban driving by imitating a reinforcement learning coach. In *Proceedings of the IEEE/CVF international conference on computer vision*, 15222–15232.

Zheng, Y.; Harley, A. W.; Shen, B.; Wetzstein, G.; and Guibas, L. J. 2023. Pointodyssey: A large-scale synthetic dataset for long-term point tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 19855–19865.

Zhou, Z.; Cai, T.; Zhao, S. Z.; Zhang, Y.; Huang, Z.; Zhou, B.; and Ma, J. 2025. AutoVLA: A Vision-Language-Action Model for End-to-End Autonomous Driving with Adaptive Reasoning and Reinforcement Fine-Tuning. *arXiv preprint arXiv:2506.13757*.

Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; and Dai, J. 2020. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*.