



AutoVLA: A Vision-Language-Action Model for End-to-End Autonomous Driving with Adaptive Reasoning and Reinforcement Fine-Tuning

Zewei Zhou* Tianhui Cai* Seth Z. Zhao Yun Zhang Zhiyu Huang† Bolei Zhou Jiaqi Ma

University of California, Los Angeles

<https://autovla.github.io/>

Abstract

Recent advancements in Vision-Language-Action (VLA) models have shown promise for end-to-end autonomous driving by leveraging world knowledge and reasoning capabilities. However, current VLA models often struggle with physically infeasible action outputs, complex model structures, or unnecessarily long reasoning. In this paper, we propose **AutoVLA**, a novel VLA model that unifies reasoning and action generation within a single autoregressive generation model for end-to-end autonomous driving. AutoVLA performs semantic reasoning and trajectory planning directly from raw visual inputs and language instructions. We tokenize continuous trajectories into discrete, feasible actions, enabling direct integration into the language model. For training, we employ supervised fine-tuning to equip the model with dual thinking modes: fast thinking (trajectory-only) and slow thinking (enhanced with chain-of-thought reasoning). To further enhance planning performance and efficiency, we introduce a reinforcement fine-tuning method based on Group Relative Policy Optimization (GRPO), reducing unnecessary reasoning in straightforward scenarios. Extensive experiments across real-world and simulated datasets and benchmarks, including nuPlan, nuScenes, Waymo, and CARLA, demonstrate the competitive performance of AutoVLA in both open-loop and closed-loop settings. Qualitative results showcase the adaptive reasoning and accurate planning capabilities of AutoVLA in diverse scenarios.

1 Introduction

Autonomous driving systems typically adopt a modular paradigm, decomposing the driving task into different sub-modules, such as perception [1–3], prediction [4–6], and planning [7–9]. While this design enables structured development, it may cause error accumulation and a lack of joint optimization across modules, leading to suboptimal performance [10, 11]. End-to-end autonomous driving has gained prominence with a unified model architecture that maps raw sensor inputs directly to final driving actions. These models are trained on human driving data, enhancing scalability and human-like behavior. Vision-based approaches have garnered significant interest due to their affordability and ease of deployment [12–15].

However, conventional end-to-end methods [16–19] primarily focus on imitating expert trajectories, lacking essential world knowledge for understanding and reasoning about surrounding environments, particularly in long-tail or challenging scenarios. Recent advances in Vision-Language Models (VLMs) [20–22] have gained significant interest by introducing models capable of leveraging extensive world knowledge and powerful reasoning. These models have shown strong potential in

*Equal contribution. Email: {zeweizhou, tianhui}@ucla.edu

†Corresponding author. Email: zhiyuh@ucla.edu

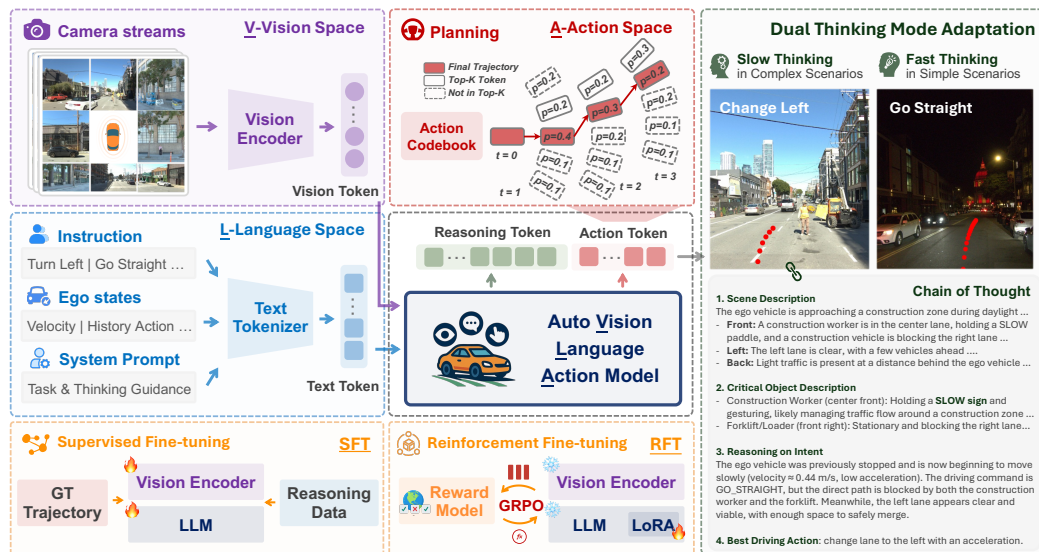


Figure 1: AutoVLA is an end-to-end autonomous driving framework based on vision-language models that integrates world knowledge into the driving policy. It takes visual observations, vehicle states, and language instructions as input and incorporates CoT reasoning and physical action tokenization to generate planning trajectories directly. The model is trained using supervised fine-tuning to jointly learn reasoning and action, and reinforcement fine-tuning is further applied to enable adaptive reasoning through fast and slow thinking modes, improving performance and efficiency.

improving adaptability and scalability across diverse driving scenarios [23–29]. Building upon VLMs, Vision-Language-Action (VLA) models extend this capability to action generation, enabling embodied agents, such as robots [30–32] and autonomous vehicles [33, 34], to produce feasible physical actions based on visual observations and language instructions.

Despite recent progress, existing VLA models face two critical limitations in autonomous driving, as illustrated in Fig. 2. 1) *Physically-infeasible or complex structure for action generation*. Some models generate textual actions or waypoints directly using VLMs [35–37], but these outputs can be physically infeasible and suffer from mode collapse. To address this, recent approaches introduce intermediate meta-actions [38–40] or latent action tokens [41–43], which are then processed by downstream planners or decoders to produce physically feasible trajectories. However, the intermediate representations either break the end-to-end optimization paradigm or increase model complexity and training overhead. 2) *Inflexible and inefficient reasoning across diverse scenarios*. Most existing models [44, 45] employ a fixed reasoning strategy, lacking the ability to adaptively switch between direct action outputs for straightforward scenarios and chain-of-thought (CoT) reasoning for complex ones. Although DriveVLM [46] introduces a dual-process paradigm, it relies on separate modules (i.e., a VLM for slow reasoning and a conventional end-to-end model for fast responses), which results in a complicated architecture, increased training overhead, and limited scalability [47].

To overcome these limitations, we propose **AutoVLA**, an end-to-end autonomous driving framework that directly integrates physical action tokens into a pretrained VLM backbone, enabling direct learning of an autoregressive planning policy, as illustrated in Fig. 1. Our unified architecture seamlessly integrates reasoning and action generation, allowing adaptive switching between direct trajectory generation and CoT reasoning. In supervised fine-tuning (SFT), we leverage both trajectory-only data and CoT reasoning data to equip the model with dual-process capabilities (fast and slow thinking). Furthermore, we propose reinforcement fine-tuning (RFT) [48], utilizing Group Relative Policy Optimization (GRPO) [49] with verifiable planning reward functions. This enables adaptive reasoning that balances planning accuracy and efficiency. The RFT method not only improves planning performance but also runtime efficiency by minimizing unnecessary reasoning.

We extensively evaluate AutoVLA using real-world datasets, including nuPlan [50, 51], Waymo [52], nuScenes [53], and simulation datasets such as CARLA [54, 55]. Experimental results demonstrate that AutoVLA achieves superior performance across various end-to-end autonomous driving benchmarks under both open-loop and closed-loop tests. Empirical results validate that our RFT approach

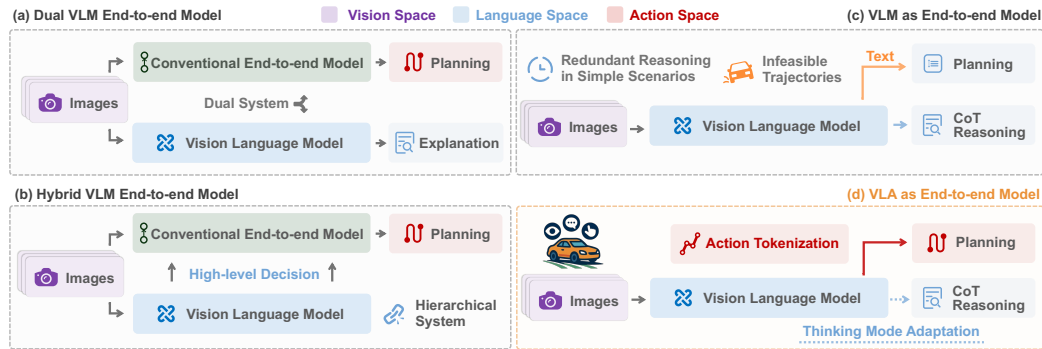


Figure 2: Four paradigms of VLMs for end-to-end autonomous driving. Compared to other methods, our proposed VLA-based paradigm enables direct trajectory planning and adaptive reasoning from visual inputs. By incorporating physical action tokenization into the language model, our model effectively integrates high-level scene reasoning and low-level trajectory planning.

markedly improves planning performance, enables adaptive fast and slow thinking capabilities, and reduces runtime by minimizing redundant reasoning. The main contributions of this paper are summarized as follows:

1. We introduce AutoVLA, an end-to-end autonomous driving framework leveraging a pretrained VLM backbone integrated with physical action tokens, enabling direct policy learning and semantic reasoning from raw visual observations and language instructions.
2. We propose an RL-based post-training method using GRPO to enable adaptive reasoning and further enhance the model's performance on end-to-end driving tasks.
3. We demonstrate that AutoVLA achieves superior performance across multiple autonomous driving benchmarks, including both open-loop and closed-loop testing.

2 Related Work

End-to-end Autonomous Driving. End-to-end autonomous driving approaches have made significant advances in recent years [10, 11, 56–64]. Methods such as UniAD [65] and VAD [66] explicitly integrate multiple driving tasks from perception to planning in a unified Transformer architecture, thereby enhancing planning performance. ParaDrive [67] discusses the necessary components within end-to-end driving architectures. Additionally, GenAD [68] and DiffusionDrive [69] adopt generative models to maintain trajectory continuity and produce multi-modal driving trajectories. However, integrating world knowledge into end-to-end driving systems remains challenging due to bottlenecks in semantic reasoning [34] and limited adaptability in complex environments [70].

VLA and VLM for Autonomous Driving. The gap between semantic reasoning and physical actions remains a critical challenge for VLA and VLM in end-to-end autonomous driving. Current research broadly follows three directions. The first directly formulates driving as a language-centric problem, utilizing VLMs for scenario understanding through caption generation [71–73] or question answering [74, 75]. The second direction leverages VLA or VLM to produce high-level meta-actions or driving decisions [17, 38–40], which are used to either supervise [12, 76–78] or guide [39, 79] traditional planners or end-to-end models. Although these approaches facilitate integration, they prevent full end-to-end optimization. Thus, a third direction directly integrates VLMs with action generation into VLA models, enabling the direct prediction of latent action tokens [34–36, 43] or final driving trajectories [37, 44, 80–83]. However, simple trajectory decoders employed in these methods (e.g., MLP [41, 84] or GRU [45]) may produce impractical trajectories and suffer from modal collapse. To address this issue, ORION [42] incorporates generative planners into VLM architectures, enhancing trajectory feasibility but increasing model complexity and computational demands. In our work, we integrate a physical action codebook for vehicle motion into a pretrained VLM to effectively bridge the semantic reasoning and physical action space.

Reinforcement Fine-tuning. RFT [48] has shown considerable promise in enhancing the performance and adaptability of LLMs, as demonstrated in DeepSeek-R1 [22]. In autonomous driving,



Figure 3: Overview of the AutoVLA model and its training process. A pretrained small VLM is used as the backbone of AutoVLA. The model receives multi-view camera streams, system prompts, driving instructions, and vehicle status as input, and outputs textual reasoning and physical action tokens. In SFT, a large VLM model with strong visual understanding capabilities is employed to collect reasoning data, which is used alongside trajectory data in SFT for training the AutoVLA model. In RFT, we utilize GRPO to train the model for improved alignment with verified reward functions, while enabling adaptive reasoning by penalizing excessive reasoning.

Gen-Drive [7] and TrajHF [85] employed the RFT to align the trajectory generation model with safety constraints and human driving preferences. RAD [86] combined 3D Gaussian splatting to generate scenarios and conduct closed-loop RL training. However, the application of RFT in end-to-end VLM/VLA-based autonomous driving remains nascent. While previous methods, such as AlphaDrive [38], utilize GRPO instead of direct preference optimization (DPO) [87] to enhance planning performance and ensure training efficiency and stability, they are still limited to simplified settings involving only high-level meta-actions. In this work, we advance this direction by applying RFT to optimize the end-to-end VLA framework in both scene reasoning and low-level planning, and we adopt GRPO to accelerate convergence and enhance training stability.

3 AutoVLA

The proposed AutoVLA framework consists of two main components, as shown in Fig. 1. 1) *VLM Backbone*: It is capable of processing visual and textual input and generating corresponding tokens (reasoning and action), employing a unified autoregressive Transformer decoder. 2) *Physical Action Token Generation*: We extend the language model decoder to output physical action tokens that directly correspond to vehicle movements. These tokens are designed to comply with physical constraints and can be reliably translated into physically feasible planning trajectories.

Training of AutoVLA is conducted in two stages, as illustrated in Fig. 3. 1) *Supervised Fine-Tuning* uses ground-truth trajectory data and distills high-quality reasoning data from a large-scale VLM. 2) *Reinforcement Fine-Tuning* uses task-specific reward functions to optimize planning performance while improving the running efficiency by minimizing unnecessary reasoning. The details of our model and training process are illustrated below.

3.1 Framework

Model Inputs. AutoVLA takes as input multi-view, multi-frame camera data C from onboard cameras, high-level navigation instructions I , and ego vehicle states S , and performs scene reasoning

and trajectory planning. Specifically, we utilize three RGB cameras positioned at the front, front-left, and front-right sides of the vehicle. Each camera stream $c^i = [c_{t-3}^i, c_{t-2}^i, c_{t-1}^i, c_t^i]$ captures four sequential frames at a frequency of 2 Hz, including the current and three preceding frames, supplying temporal information for scene dynamics. Additionally, the model employs high-level navigation instructions I (e.g., Turn Left and Go Straight) to specify intended directions explicitly. The ego vehicle's state S encompasses current velocity, acceleration, and historical actions.

Base VLM Model. We adopt Qwen2.5-VL-3B [21] as the vision-language backbone of AutoVLA. Qwen2.5-VL is a series of powerful multimodal large language models that possess strong visual understanding capabilities, and the open-source nature of the Qwen2.5-VL model facilitates task-specific fine-tuning. The 3B variant offers a good trade-off between efficiency and performance, making it suitable for deployment in onboard devices.

Action Tokenization. To enable trajectory planning within the language model, we discretize continuous vehicle trajectories $\mathbf{P} \in \mathbb{R}^{T \times d}$ into a sequence of physical action tokens $\mathbf{a} = [a_1, \dots, a_T]$, where $a_t \in \mathcal{A}$, T is the length of the tokenized predicted trajectory and each token is represented by short-term spatial position and heading movement $(\Delta x, \Delta y, \Delta \theta)$. This transforms the planning task into a next-token prediction problem, which can be conducted within the language model. We build our action codebook $\mathcal{A} = \{a_1, a_2, \dots, a_K\}$ using a K-disk clustering method [88–90], which covers the majority of vehicle movement patterns. Finally, we obtain a vehicle motion codebook that consists of $K = 2048$ discrete action tokens. Following [30, 91], these action tokens are incorporated into the VLM as additional tokens (i.e., `<action_0>`, `<action_1>`, ...). During inference, the model outputs a sequence of these action tokens, which are subsequently decoded into a planning trajectory using the action codebook. More details about action tokenization and trajectory decoding are provided in the Supplementary Material.

Unified Reasoning and Action. AutoVLA unifies reasoning and action generation within a single autoregressive Transformer framework, enabling adaptive switching between fast and slow thinking depending on the driving scenario. In fast thinking mode, AutoVLA directly predicts physical action tokens without generating long CoT reasoning, enabling rapid responses in straightforward scenarios. In contrast, slow thinking mode involves structured CoT reasoning, where the model first analyzes the environment, identifies critical elements, and reasons through potential outcomes before deciding on the final driving action. To enable this dual thinking capability, AutoVLA is trained with a mixture of direct action supervision and reasoning-augmented data. We design system prompts and response formats to support both modes consistently.

3.2 Reasoning Data

Reasoning data provides high-quality CoT annotations that are essential for training VLMs with reasoning capabilities [42]. In driving tasks, reasoning involves understanding complex semantics and interactions in dynamic environments [92–95]. Despite its importance, the development of a high-quality, large-scale driving reasoning dataset remains a key challenge due to three major limitations: 1) limited scenario diversity and repetitive examples, 2) inadequate representation of critical perceptual cues, such as traffic signs and vehicle indicator signals, 3) low-quality reasoning process, such as repeatedly stopping at a stop sign without justification.

To address these issues, we propose an automated reasoning annotation pipeline using the advanced Qwen2.5-VL-72B model [21]. This pipeline enables automatic generation of high-accuracy reasoning annotations and supports knowledge distillation from a large capable model to a more compact target model. The pipeline generates structured reasoning annotations across four key components: detailed scene descriptions, identification of crucial objects, prediction of surrounding agents' intentions, and determination of appropriate driving actions. To regulate the reasoning outcomes, our approach incorporates ground-truth driving actions as hints, guiding the model to produce causal explanations that explicitly link driving decisions to scene context. This structured prompting method significantly reduces nonsensical outputs and minimizes the need for manual correction.

Employing this annotation pipeline, we compile a comprehensive reasoning dataset comprising approximately 45.6k CoT reasoning annotations for the nuPlan dataset and 7.2k annotations for the Waymo end-to-end driving dataset. Additionally, we reformat and integrate DriveLM [96], a VQA dataset built on nuScenes and CARLA simulation data, to augment our reasoning data. Additional details and illustrative examples are provided in the Supplementary Material.

3.3 Supervised Fine-tuning

Supervised fine-tuning (SFT) is employed to train the model to generate both reasoning and action sequences. Given multi-frame camera images C , a high-level navigation instruction I , and the ego vehicle state S , the model is trained to produce a sequence of output tokens. The output sequence consists of language tokens $\mathbf{l} = [l_1, \dots, l_L]$ for reasoning followed by action tokens $\mathbf{a} = [a_1, \dots, a_T]$. To enable both fast and slow thinking during SFT, we curate training data with ground-truth assistant responses that either include only the final action tokens or combine CoT reasoning with the corresponding action tokens. In the fast-thinking mode, $\mathbf{l}$ is a fixed, short template indicating that reasoning is not needed. Conversely, in the slow-thinking mode, $\mathbf{l}$ begins with a template that introduces the need for CoT reasoning, followed by a structured sequence of reasoning.

The first supervision signal is the standard causal language modeling objective, which minimizes the negative log-likelihood of the target token sequence and facilitates the reasoning capability. The other supervision signal focuses on the planning accuracy, and we introduce an auxiliary loss over action tokens $\mathbf{a} = [a_1, \dots, a_T]$, which appear at positions x_{L+1} to x_{L+T} in the output sequence. Given an output sequence $\mathbf{x} = [l_1, \dots, l_L, a_1, \dots, a_T]$, the loss functions are defined as:

$$\mathcal{L}_{\text{LM}} = -\frac{1}{N} \sum_{i=1}^N \log p_{\theta}(x_i | x_{<i}, C, I, S), \quad \mathcal{L}_{\text{action}} = -\frac{1}{T} \sum_{i=L+1}^{L+T} \log p_{\theta}(x_i | x_{<i}, C, I, S), \quad (1)$$

where $N = L + T$, and p_{θ} denotes the model's predicted distribution parameterized by θ .

To jointly optimize reasoning and action generation, we combine the language modeling loss and the action loss into a single SFT loss function. To address the imbalance between reasoning data and action-only data, and to encourage the model to learn from examples that include CoT reasoning, we apply a per-sample weighting factor based on the presence of CoT in the ground truth. The overall loss for each training example is computed as follows:

$$\mathcal{L}_i^{\text{SFT}} = w_i \cdot (\mathcal{L}_{\text{LM},i} + \lambda_a \mathcal{L}_{\text{action},i}), \quad w_i = \begin{cases} \lambda_{\text{cot}} & \text{if CoT is present in GT} \\ 1 & \text{otherwise} \end{cases}, \quad (2)$$

where λ_a and λ_{cot} are hyperparameters that control the relative importance.

3.4 Reinforcement Fine-tuning

To further improve the performance of AutoVLA and align it with driving requirements and task-specific rewards, we introduce a reinforcement learning-based post-training method. This RFT stage enables the model to perform adaptive reasoning and optimize planning performance. We employ the GRPO algorithm [49], which stabilizes training and improves convergence efficiency. Moreover, the inherent multi-modality of planning, characterized by multiple feasible trajectories in the same scenario, naturally aligns with the group-based optimization framework of GRPO [38].

Given a scenario input query q , comprising sensor images, the ego vehicle's state, and driving instruction, we sample a set of G candidate outputs $O = \{o_1, o_2, \dots, o_G\}$ from the old policy $\pi_{\theta_{\text{old}}}$. The current policy π_{θ} is then optimized using the normalized group-relative advantage A_i , by maximizing the following objective:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{q, \{o_i\} \sim \pi_{\theta_{\text{old}}}(O|q)} \left[\frac{1}{G} \sum_{i=1}^G (\mathcal{J}_i^R - \beta \mathbb{D}_{\text{KL}}(\pi_{\theta} \parallel \pi_{\text{ref}})) \right], \quad (3)$$

$$\mathcal{J}_i^R = \min \left(\frac{\pi_{\theta}(o_i|q)}{\pi_{\theta_{\text{old}}}(o_i|q)} A_i, \text{clip} \left(\frac{\pi_{\theta}(o_i|q)}{\pi_{\theta_{\text{old}}}(o_i|q)}, 1 - \epsilon, 1 + \epsilon \right) A_i \right), \quad A_i = \frac{r_i - \text{mean}(\{r_j\}_{j=1}^G)}{\text{std}(\{r_j\}_{j=1}^G)}, \quad (4)$$

where θ and θ_{old} denote the current and old policy parameters, r_i is the reward for sample o_i , ϵ and β are hyperparameters controlling the clipping range and the weight of the KL divergence regularization term, and π_{ref} is the reference policy from the SFT stage.

The final reward function is defined as $r = r_{\text{Driving}} - \lambda_r r_{\text{CoT}}$, where the λ_r denotes the balance weight. The term r_{Driving} varies across benchmarks. For the nuPlan dataset, we employ the Predictive Driver Model Score (PDMS) [51] as the driving reward, which captures aspects such as safety, comfort,

Table 1: Testing Results on the NAVSIM (nuPlan) End-to-end Driving Benchmark

Method	PDMS $\uparrow$	Collision $\uparrow$	Area $\uparrow$	Direction $\uparrow$	Progress $\uparrow$	TTC $\uparrow$	Comfort $\uparrow$
Ego Status MLP	66.40	93.09	78.26	90.45	63.20	84.02	99.97
TransFuser [58]	83.88	97.78	92.63	97.97	78.88	92.89	99.98
DRAMA [56]	86.87	98.19	95.18	98.03	81.33	94.17	100.00
Hydra-MDP [57]	91.26	99.07	98.29	95.79	85.20	96.56	100.00
Centaur [59]	92.10	99.23	98.72	96.77	85.96	97.17	99.97
TrajHF [85]	93.95	99.30	97.51	91.72	90.39	98.02	99.81
AutoVLA (One-shot)	80.54	96.89	92.42	94.43	75.82	88.06	99.94
AutoVLA (Post-RFT)	89.11	98.41	95.64	95.40	81.87	98.04	99.94
AutoVLA (Best-of-N)	92.12	99.14	97.08	95.51	87.55	97.12	99.98

travel efficiency, and other driving quality metrics. For the Waymo end-to-end driving dataset, due to the limited availability of Rater Feedback Score (RFS) annotations [52], we use the Average Displacement Error (ADE) as the driving reward. To discourage unnecessary long reasoning chains, we incorporate a CoT length penalty r_{CoT} into the reward function. Additional implementation details are provided in the Supplementary Material.

4 Experiments

4.1 Experimental Setup

Datasets. We train the AutoVLA model using a diverse set of real-world and simulation datasets. The nuPlan (Open-Scene) dataset [50, 97] contains 120 hours of large-scale driving data with eight streams of camera data and object annotations. The Waymo end-to-end driving dataset (Waymo E2E) [52] comprises 4,021 20-second driving segments with eight streams of camera views and ego vehicle trajectories, especially focusing on challenging and long-tail scenarios, such as driving through construction areas or risky situations. The nuScenes dataset [53] provides 1,000 urban driving scenes with six camera views. The CARLA-Garage dataset [55] provides over 500,000 frames of simulation camera data. In addition to the collected reasoning data, we utilize the DriveLM dataset [96] for nuScenes and CARLA datasets, by reformatting the VQA pairs to facilitate CoT reasoning.

Benchmarks. We evaluate AutoVLA on both open-loop and closed-loop benchmarks across real-world and simulated environments. Open-loop performance is assessed on two public benchmarks: the NAVSIM benchmark [51] from the nuPlan dataset and the nuScenes benchmark [65]. The NAVSIM benchmark employs PDMS to assess key aspects of driving behavior, such as collision and ego progress. The nuScenes benchmark uses L2 distance and collision rate as evaluation metrics. Additionally, we report our model’s performance on the Waymo end-to-end driving benchmark using the RFS metric, which reflects human-judged planning quality. Closed-loop performance is evaluated on the Bench2Drive benchmark [54] in the CARLA simulator. Bench2Drive contains 44 interactive, closed-loop scenarios under varying locations and weather conditions, using metrics such as success rate, driving score, efficiency, and comfort.

Implementation Details. Each action token corresponds to 0.5 seconds of movement, and the planning horizon is set to 5 seconds. Consequently, the model outputs 10 action tokens, from which a 5-second trajectory can be decoded. For SFT, we use a learning rate of 1×10^{-5} and the FSDP training strategy. The model is trained for 5 epochs using 8 NVIDIA L40S GPUs. We use a per-GPU batch size of 1 and accumulate gradients over 4 steps, resulting in an effective batch size of 32. The weighting parameters in the SFT loss function are set to $\lambda_a = 1$ and $\lambda_{cot} = 40$. For RFT, we employ the LoRA adapter [98] for parameter-efficient training. The learning rate for RFT is set to 3×10^{-5} , and the KL regularization weight β is set to 0.04. We perform a single policy update at each step, allowing the use of a simplified objective without the need for clipping or tracking the old policy. The model is fine-tuned for 6,000 steps, and the best-performing checkpoint is selected for evaluation. Additional details are provided in the Supplementary Material.

4.2 Main Results

This section reports the main results of the AutoVLA model for various datasets and benchmarks, with additional results included in the Supplementary Material.



Figure 5: Reinforcement fine-tuning results on the nuPlan dataset. (a) Comparison of PDMS and runtime before and after RFT; (b) Training reward curves for different GRPO group sample sizes; (c) Qualitative comparison of planning and reasoning performance before and after RFT.

Data Scaling Results. We train AutoVLA on a mixture of the nuPlan and nuScenes datasets with varying training set sizes (10k, 50k, 100k, and the full 185k samples), with action-only supervision or with additional CoT reasoning supervision. The models are evaluated on the respective standard test sets, and the results are shown in Fig. 4. We observe that increasing the amount of training data consistently improves planning performance on both datasets. In the nuPlan dataset, when using fewer than 50k training samples, CoT reasoning does not outperform action-only in terms of PDMS and Collision Score. This is likely due to the increased difficulty of learning structured reasoning from limited data. However, as the training set size increases, models trained with CoT reasoning surpass those with action-only supervision, highlighting the scalability advantages of reasoning-augmented learning. A similar trend is observed on the nuScenes dataset: as the training set size increases, models trained with CoT reasoning consistently outperform those trained with action-only data in terms of L2 distance and collision rate.

RFT Performance. The slow-thinking mode incurs a significantly higher runtime due to the generation of CoT reasoning compared to the fast-thinking mode, as shown in Table 2. To mitigate this overhead, we introduce RFT to enhance AutoVLA's adaptive thinking capability and avoid unnecessary reasoning in straightforward scenarios. Specifically, we apply RFT to the full-data CoT reasoning model trained via SFT to enhance its planning performance. As shown in Fig. 5(a), RFT yields a 10.6% improvement in PDMS (on the NAVSIM testing set) and a 66.8% reduction in runtime (average over 500 testing scenarios). The reward curve in Fig. 5(b) illustrates the progressive improvement of the model's policy during RFT. Experiments with different GRPO group

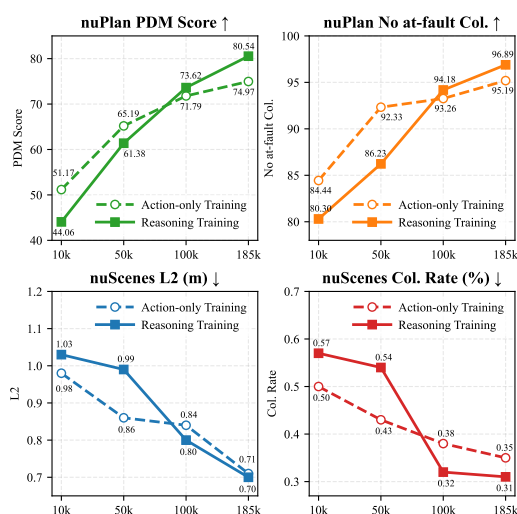


Figure 4: Data scaling effect on planning performance for nuPlan and nuScenes datasets (log-scaled x-axis). Increasing the amount of training data consistently enhances planning performance.

Table 2: Runtime Analysis of Fast & Slow Thinking Modes in AutoVLA

Thinking Mode	Min. (s)	Max. (s)	Avg. (s)
Fast Thinking	0.997	1.116	1.072
Slow Thinking	7.607	13.706	10.518

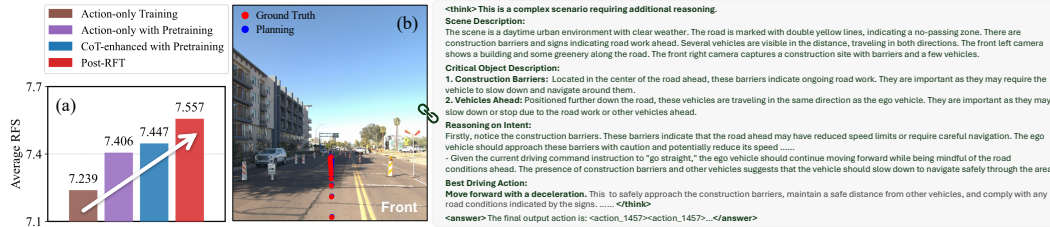


Figure 6: Performance comparison of AutoVLA with different training settings on the Waymo end-to-end driving dataset, along with an example illustrating the model’s reasoning capabilities.

sample sizes indicate that larger groups lead to better performance by promoting broader exploration of training samples. As illustrated in Fig. 5(c), RFT also reduces unnecessary and slow reasoning in simple scenarios, driven by the CoT length penalty that encourages fast thinking for straightforward driving cases. A qualitative comparison shows that the SFT model produces suboptimal plans due to error accumulation in generation, whereas the RFT model (optimized via PDMS-based reward) generates better trajectories.

nuPlan Benchmark Results. We evaluate AutoVLA against state-of-the-art end-to-end driving models on the NAVSIM benchmark [51] and present results in Table 1. In best-of-N planning, we use an oracle scorer to select the optimal trajectory from six generated candidates. After RFT, AutoVLA demonstrates significantly improved performance, aligning more closely with the NAVSIM reward signal. The best-of-N strategy further enhances performance, achieving the highest PDMS. Overall, AutoVLA achieves competitive results while demonstrating scalability across diverse datasets.

Waymo E2E Performance. We evaluate AutoVLA on the Waymo end-to-end driving dataset [52], which features long-tail and complex driving scenarios. The model’s performance under various training settings on the test set is presented in Fig. 6. The results reveal that pretraining on a combination of nuPlan and nuScenes datasets significantly enhances performance, suggesting enhanced scene understanding through exposure to more diverse training data. Incorporating CoT reasoning in training further improves planning performance compared to action-only training. Post-training with RFT, using ADE as the reward function, achieves the best overall RFS metric. A qualitative example in a construction zone demonstrates the model’s ability to reason about occlusions and generate effective detour plans.

CARLA Closed-loop Performance. We evaluate the closed-loop driving performance of our AutoVLA model on the Bench2Drive benchmark [54] in the CARLA simulator. The model is trained using SFT with both trajectory-only and CoT data. During testing, the planning frequency is set to 2 Hz. The results, shown in Table 3, demonstrate that AutoVLA outperforms existing end-to-end driving models in terms of overall driving score and success rate in the closed-loop test.

Table 3: Testing Results on the Bench2Drive (CARLA) Closed-loop Driving Benchmark

Method	Driving Score $\uparrow$	Success Rate (%) $\uparrow$	Efficiency $\uparrow$	Comfortness $\uparrow$
AD-MLP [99]	18.05	0.00	48.45	22.63
UniAD-Base [65]	45.81	16.36	129.21	43.58
VAD [66]	42.35	15.00	157.94	46.01
TCP-traj [100]	59.90	30.00	76.54	18.08
DriveAdapter [101]	64.22	33.08	70.22	16.01
Orion [42]	77.74	54.62	151.48	17.38
AutoVLA	78.84	57.73	146.93	39.33

4.3 Ablation Studies

Text Waypoint Output. We use the same mixed training set from the nuPlan and nuScenes datasets to train a model that predicts waypoints in a text format, which are then converted into a trajectory of waypoints. We evaluate its performance in an open-loop planning setting using the standard test sets. The results, shown in Table 5, indicate that our action tokenization and generation method significantly outperforms the text-based waypoint prediction approach. Additionally, due to the need to decode

Table 4: Action Tokenization Accuracy with Different Codebook Sizes and Methods. (DCT: discrete cosine transform; MC: movement coverage; CU: codebook usage. **Bold** indicates best performance, underline the second-best, and tilde the third-best.)

Codebook Size	RT-1 (Action Bin) [102]		FAST (DCT) [91]		K-disk (Ours)			
	ADE(m) ↓	FDE(m) ↓	ADE(m) ↓	FDE(m) ↓	ADE(m) ↓	FDE(m) ↓	MC ↑	CU ↑
256	0.1440	0.2942	0.1708	0.2137	0.0687	0.1034	86.47%	100.0%
1024	0.1052	0.1883	0.0522	0.0588	0.0253	0.0282	97.41%	100.0%
2048	0.1014	0.1775	0.0281	0.0309	<u>0.0182</u>	<u>0.0203</u>	<u>99.42%</u>	100.0%
4096	0.1001	0.1739	<u>0.0149</u>	<u>0.0161</u>	0.0141	0.0155	100.0%	<u>91.46%</u>

numerical values, the text-based method incurs a substantially higher computational cost in generating the final trajectory. This shows the limitation of language models in handling precise numerical reasoning.

Action Tokenization Methods. We evaluate reconstruction accuracy across different codebook sizes for several tokenization methods, including RT-1 [102], FAST [91], and our proposed K-disk tokenization, as shown in Table 4. For RT-1, acceleration and steering rate are discretized using uniform action bins, and a kinematic model is employed to reconstruct the trajectory. However, because only trajectory-level data is available and control actions must be indirectly inferred, this binning approach yields the highest reconstruction error. The FAST tokenization method is more sensitive to codebook size than our K-disk approach, achieving comparable reconstruction accuracy only at $K = 4096$. In contrast, our proposed method consistently attains the highest reconstruction accuracy across all codebook sizes.

When the codebook size is small ($K = 256$), reconstruction errors are significantly higher due to limited movement coverage, as many movements cannot be adequately represented by the small set of codebook tokens. As the codebook size increases beyond $K = 2048$, the improvements in tokenization accuracy and movement coverage become marginal, while codebook usage decreases as many tokens remain redundant and unused. Considering this trade-off, we select $K = 2048$ to balance reconstruction accuracy, movement coverage, and efficient codebook usage.

Furthermore, Table 6 reports planning performance on NAVSIM (nuPlan) for different tokenization methods. Our proposed method outperforms FAST. In FAST, the discrete cosine transform converts fixed-horizon planning trajectories into variable-length token sequences, complicating the selection of appropriate token lengths and making fixed-horizon planning difficult.

5 Conclusions

We propose AutoVLA, an end-to-end autonomous driving framework that unifies scene reasoning and action generation within a single autoregressive model. We adopt SFT to enable the model to operate in both fast thinking (direct trajectory generation) and slow thinking (enhanced with long CoT reasoning) modes. In addition, we introduce RFT to enable adaptive reasoning by penalizing unnecessary reasoning and aligning action generation with reward functions, improving both performance and efficiency. Experimental results demonstrate that AutoVLA achieves competitive performance on both open-loop and closed-loop planning benchmarks and exhibits strong reasoning capabilities.

Limitation and Future Work. Although our model with dual-process adaptation achieves near-real-time inference (1 Hz), it remains highly GPU-dependent, requiring significant memory and computing. Future work will focus on real-time applications, optimizing runtime efficiency, and reducing computation overhead (e.g., through model quantization) to enable real-time deployment.

Table 5: Influence of Physical Action Tokenization

Metric	Text Waypoint	Physical Action
PDM Score ↑	71.31	80.54
Avg. L2 (m) ↓	0.89	0.70
Avg. Col. (%) ↓	0.36	0.31
Runtime (s) ↓	7.65	3.95

Table 6: Performance Comparison across Different Tokenization Methods on NAVSIM

Tokenization	PDMS ↑	Collision ↑	Progress ↑
FAST (DCT) [91]	67.63	92.74	64.09
K-disk (Ours)	80.54	96.89	75.82

Acknowledgements

This work was supported by the Federal Highway Administration Center of Excellence on New Mobility and Automated Vehicles, and by the National Science Foundation under Award No. 2346267, POSE: Phase II - DriveX: An Open Source Ecosystem for Automated Driving and Intelligent Transportation Research.

References

- [1] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Qiao Yu, and Jifeng Dai. Bevformer: learning bird’s-eye-view representation from lidar-camera via spatiotemporal transformers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.
- [2] Yue Wang, Vitor Campagnolo Guizilini, Tianyuan Zhang, Yilun Wang, Hang Zhao, and Justin Solomon. Detr3d: 3d object detection from multi-view images via 3d-to-2d queries. In *Conference on Robot Learning*, pages 180–191. PMLR, 2022.
- [3] Tingting Liang, Hongwei Xie, Kaicheng Yu, Zhongyu Xia, Zhiwei Lin, Yongtao Wang, Tao Tang, Bing Wang, and Zhi Tang. Bevfusion: A simple and robust lidar-camera fusion framework. *Advances in Neural Information Processing Systems*, 35:10421–10434, 2022.
- [4] Zikang Zhou, Zihao Wen, Jianping Wang, Yung-Hui Li, and Yu-Kai Huang. Qcnext: A next-generation framework for joint multi-agent trajectory prediction. *arXiv preprint arXiv:2306.10508*, 2023.
- [5] Shaoshuai Shi, Li Jiang, Dengxin Dai, and Bernt Schiele. Mtr++: Multi-agent motion prediction with symmetric scene modeling and guided intention querying. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(5):3955–3971, 2024.
- [6] Zhiyu Huang, Haochen Liu, and Chen Lv. Gameformer: Game-theoretic modeling and learning of transformer-based interactive prediction and planning for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3903–3913, 2023.
- [7] Zhiyu Huang, Xinshuo Weng, Maximilian Igl, Yuxiao Chen, Yulong Cao, Boris Ivanovic, Marco Pavone, and Chen Lv. Gen-drive: Enhancing diffusion generative driving policies with reward modeling and reinforcement learning fine-tuning. *arXiv preprint arXiv:2410.05582*, 2024.
- [8] Zhiyu Huang, Haochen Liu, Jingda Wu, and Chen Lv. Differentiable integrated motion prediction and planning with learnable cost function for autonomous driving. *IEEE transactions on neural networks and learning systems*, 2023.
- [9] Haochen Liu, Zhiyu Huang, Wenhui Huang, Haohan Yang, Xiaoyu Mo, and Chen Lv. Hybrid-prediction integrated planning for autonomous driving. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.
- [10] Hao Shao, Letian Wang, Ruobing Chen, Steven L Waslander, Hongsheng Li, and Yu Liu. Reasonnet: End-to-end driving with temporal and global reasoning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13723–13733, 2023.
- [11] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21983–21994, 2023.
- [12] Chenbin Pan, Burhaneddin Yaman, Tommaso Nesti, Abhirup Mallik, Alessandro G Allievi, Senem Velipasalar, and Liu Ren. Vlp: Vision language planning for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14760–14769, 2024.

- [13] Yuqi Wang, Jiawei He, Lue Fan, Hongxin Li, Yuntao Chen, and Zhaoxiang Zhang. Driving into the future: Multiview visual forecasting and planning with world model for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14749–14759, June 2024.
- [14] Shengchao Hu, Li Chen, Penghao Wu, Hongyang Li, Junchi Yan, and Dacheng Tao. St-p3: End-to-end vision-based autonomous driving via spatial-temporal feature learning. In *European Conference on Computer Vision*, pages 533–549. Springer, 2022.
- [15] Shaoheng Fang, Zi Wang, Yiqi Zhong, Junhao Ge, and Siheng Chen. Tbp-former: Learning temporal bird’s-eye-view pyramid for joint perception and prediction in vision-centric autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1368–1378, 2023.
- [16] 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. *IEEE Transactions on Intelligent Vehicles*, 2024.
- [17] Tianhui Cai, Yifan Liu, Zewei Zhou, Haoxuan Ma, Seth Z Zhao, Zhiwen Wu, and Jiaqi Ma. Driving with regulation: Interpretable decision-making for autonomous vehicles with retrieval-augmented reasoning via llm. *arXiv preprint arXiv:2410.04759*, 2024.
- [18] Li Chen, Penghao Wu, Kashyap Chitta, Bernhard Jaeger, Andreas Geiger, and Hongyang Li. End-to-end autonomous driving: Challenges and frontiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.
- [19] Daocheng Fu, Xin Li, Licheng Wen, Min Dou, Pinlong Cai, Botian Shi, and Yu Qiao. Drive like a human: Rethinking autonomous driving with large language models. In *2024 IEEE/CVF Winter Conference on Applications of Computer Vision Workshops (WACVW)*, pages 910–919. IEEE, 2024.
- [20] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.
- [21] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibao Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.
- [22] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.
- [23] Kai Chen, Yanze Li, Wenhua Zhang, Yanxin Liu, Pengxiang Li, Ruiyuan Gao, Lanqing Hong, Meng Tian, Xinhai Zhao, Zhenguo Li, et al. Automated evaluation of large vision-language models on self-driving corner cases. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 7817–7826. IEEE, 2025.
- [24] Boyi Li, Yue Wang, Jiageng Mao, Boris Ivanovic, Sushant Veer, Karen Leung, and Marco Pavone. Driving everywhere with large language model policy adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14948–14957, 2024.
- [25] 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.
- [26] Shihao Wang, Zhiding Yu, Xiaohui Jiang, Shiyi Lan, Min Shi, Nadine Chang, Jan Kautz, Ying Li, and Jose M Alvarez. Omnidrive: A holistic llm-agent framework for autonomous driving with 3d perception, reasoning and planning. *arXiv preprint arXiv:2405.01533*, 2024.
- [27] Yingzi Ma, Yulong Cao, Jiachen Sun, Marco Pavone, and Chaowei Xiao. Dolphins: Multimodal language model for driving. In *European Conference on Computer Vision*, pages 403–420. Springer, 2024.

- [28] Katharina Winter, Mark Azer, and Fabian B Flohr. Bevdriver: Leveraging bev maps in llms for robust closed-loop driving. *arXiv preprint arXiv:2503.03074*, 2025.
- [29] Kexin Tian, Jingrui Mao, Yunlong Zhang, Jiwan Jiang, Yang Zhou, and Zhengzhong Tu. Nuscenes-spatialqa: A spatial understanding and reasoning benchmark for vision-language models in autonomous driving. *arXiv preprint arXiv:2504.03164*, 2025.
- [30] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.
- [31] Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. *arXiv preprint arXiv:2503.22020*, 2025.
- [32] Chia-Yu Hung, Qi Sun, Pengfei Hong, Amir Zadeh, Chuan Li, U-Xuan Tan, Navonil Majumder, and Soujanya Poria. Nora: A small open-sourced generalist vision language action model for embodied tasks, 2025. URL <https://arxiv.org/abs/2504.19854>.
- [33] Hidehisa Arai, Keita Miwa, Kento Sasaki, Kohei Watanabe, Yu Yamaguchi, Shunsuke Aoki, and Issei Yamamoto. Covla: Comprehensive vision-language-action dataset for autonomous driving. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 1933–1943. IEEE, 2025.
- [34] Xingcheng Zhou, Xuyuan Han, Feng Yang, Yunpu Ma, and Alois C Knoll. Opendrivevla: Towards end-to-end autonomous driving with large vision language action model. *arXiv preprint arXiv:2503.23463*, 2025.
- [35] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15120–15130, 2024.
- [36] Katrin Renz, Long Chen, Elahe Arani, and Oleg Sinavski. Simlingo: Vision-only closed-loop autonomous driving with language-action alignment. *arXiv preprint arXiv:2503.09594*, 2025.
- [37] Jyh-Jing Hwang, Runsheng Xu, Hubert Lin, Wei-Chih Hung, Jingwei Ji, Kristy Choi, Di Huang, Tong He, Paul Covington, Benjamin Sapp, et al. Emma: End-to-end multimodal model for autonomous driving. *arXiv preprint arXiv:2410.23262*, 2024.
- [38] Bo Jiang, Shaoyu Chen, Qian Zhang, Wenyu Liu, and Xinggang Wang. Alphadrive: Unleashing the power of vlms in autonomous driving via reinforcement learning and reasoning. *arXiv preprint arXiv:2503.07608*, 2025.
- [39] 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. *arXiv preprint arXiv:2410.22313*, 2024.
- [40] Wenhai Wang, Jiangwei Xie, ChuanYang Hu, Haoming Zou, Jianan Fan, Wenwen Tong, Yang Wen, Silei Wu, Hanming Deng, Zhiqi Li, et al. Drivemlm: Aligning multi-modal large language models with behavioral planning states for autonomous driving. *arXiv preprint arXiv:2312.09245*, 2023.
- [41] Katrin Renz, Long Chen, Ana-Maria Marcu, Jan Hünermann, Benoit Hanotte, Alice Karnsund, Jamie Shotton, Elahe Arani, and Oleg Sinavski. Carllava: Vision language models for camera-only closed-loop driving. *arXiv preprint arXiv:2406.10165*, 2024.
- [42] Haoyu Fu, Diankun Zhang, Zongchuang Zhao, Jianfeng Cui, Dingkan Liang, Chong Zhang, Dingyuan Zhang, Hongwei Xie, Bing Wang, and Xiang Bai. Orion: A holistic end-to-end autonomous driving framework by vision-language instructed action generation. *arXiv preprint arXiv:2503.19755*, 2025.
- [43] Waywe Research Team et al. Lingo-2: Driving with natural language, 2024.

- [44] Shuo Xing, Chengyuan Qian, Yuping Wang, Hongyuan Hua, Kexin Tian, Yang Zhou, and Zhengzhong Tu. Openemma: Open-source multimodal model for end-to-end autonomous driving. In *Proceedings of the Winter Conference on Applications of Computer Vision*, pages 1001–1009, 2025.
- [45] Tianqi Wang, Enze Xie, Ruihang Chu, Zhenguo Li, and Ping Luo. Drivecot: Integrating chain-of-thought reasoning with end-to-end driving. *arXiv preprint arXiv:2403.16996*, 2024.
- [46] 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. *arXiv preprint arXiv:2402.12289*, 2024.
- [47] Wenjie Ma, Jingxuan He, Charlie Snell, Tyler Griggs, Sewon Min, and Matei Zaharia. Reasoning models can be effective without thinking. *arXiv preprint arXiv:2504.09858*, 2025.
- [48] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.
- [49] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.
- [50] Napat Karnchanachari, Dimitris Geromichalos, Kok Seang Tan, Nanxiang Li, Christopher Eriksen, Shakiba Yaghoubi, Noushin Mehdipour, Gianmarco Bernasconi, Whye Kit Fong, Yiluan Guo, et al. Towards learning-based planning: The nuplan benchmark for real-world autonomous driving. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 629–636. IEEE, 2024.
- [51] Daniel Dauner, Marcel Hallgarten, Tianyu Li, Xinshuo Weng, Zhiyu Huang, Zetong Yang, Hongyang Li, Igor Gilitschenski, Boris Ivanovic, Marco Pavone, et al. Navsim: Data-driven non-reactive autonomous vehicle simulation and benchmarking. *Advances in Neural Information Processing Systems*, 37:28706–28719, 2024.
- [52] Waymo Research. 2025 waymo open dataset challenge: Vision-based end-to-end driving. <https://waymo.com/open/challenges/2025/e2e-driving/>, 2025. Accessed: 2025-04-25.
- [53] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.
- [54] Xiaosong Jia, Zhenjie Yang, Qifeng Li, Zhiyuan Zhang, and Junchi Yan. Bench2drive: Towards multi-ability benchmarking of closed-loop end-to-end autonomous driving. *arXiv preprint arXiv:2406.03877*, 2024.
- [55] Bernhard Jaeger, Kashyap Chitta, and Andreas Geiger. Hidden biases of end-to-end driving models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8240–8249, October 2023.
- [56] Chengran Yuan, Zhanqi Zhang, Jiawei Sun, Shuo Sun, Zefan Huang, Christina Dao Wen Lee, Dongen Li, Yuhang Han, Anthony Wong, Keng Peng Tee, et al. Drama: An efficient end-to-end motion planner for autonomous driving with mamba. *arXiv preprint arXiv:2408.03601*, 2024.
- [57] Zhenxin Li, Kailin Li, Shihao Wang, Shiyi Lan, Zhiding Yu, Yishen Ji, Zhiqi Li, Ziyue Zhu, Jan Kautz, Zuxuan Wu, et al. Hydra-mdp: End-to-end multimodal planning with multi-target hydra-distillation. *arXiv preprint arXiv:2406.06978*, 2024.
- [58] Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *IEEE transactions on pattern analysis and machine intelligence*, 45(11):12878–12895, 2022.

- [59] Chonghao Sima, Kashyap Chitta, Zhiding Yu, Shiyi Lan, Ping Luo, Andreas Geiger, Hongyang Li, and Jose M Alvarez. Centaur: Robust end-to-end autonomous driving with test-time training. *arXiv preprint arXiv:2503.11650*, 2025.
- [60] Zewei Zhou, Hao Xiang, Zhaoliang Zheng, Seth Z Zhao, Mingyue Lei, Yun Zhang, Tianhui Cai, Xinyi Liu, Johnson Liu, Maheswari Bajji, et al. V2XPnP: Vehicle-to-everything spatio-temporal fusion for multi-agent perception and prediction. *arXiv preprint arXiv:2412.01812*, 2024.
- [61] Wenchao Sun, Xuewu Lin, Yining Shi, Chuang Zhang, Haoran Wu, and Sifa Zheng. Sparsedrive: End-to-end autonomous driving via sparse scene representation. *arXiv preprint arXiv:2405.19620*, 2024.
- [62] Yiheng Li, Seth Z. Zhao, Chenfeng Xu, Chen Tang, Chenran Li, Mingyu Ding, Masayoshi Tomizuka, and Wei Zhan. Pre-training on synthetic driving data for trajectory prediction. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5910–5917, 2024. doi: 10.1109/IROS58592.2024.10802492.
- [63] Wenzhao Zheng, Junjie Wu, Yao Zheng, Sicheng Zuo, Zixun Xie, Longchao Yang, Yong Pan, Zhihui Hao, Peng Jia, Xianpeng Lang, et al. Gaussianad: Gaussian-centric end-to-end autonomous driving. *arXiv preprint arXiv:2412.10371*, 2024.
- [64] Yuping Wang, Shuo Xing, Cui Can, Renjie Li, Hongyuan Hua, Kexin Tian, Zhaobin Mo, Xiangbo Gao, Keshu Wu, Sulong Zhou, Hengxu You, Juntong Peng, Junge Zhang, Zehao Wang, Rui Song, Mingxuan Yan, Walter Zimmer, Xingcheng Zhou, Peiran Li, Zhaohan Lu, Chia-Ju Chen, Yue Huang, Ryan A. Rossi, Lichao Sun, Hongkai Yu, Zhiwen Fan, Frank Hao Yang, Yuhao Kang, Ross Greer, Chenxi Liu, Eun Hak Lee, Xuan Di, Xinyue Ye, Liu Ren, Alois Knoll, Xiaopeng Li, Shuiwang Ji, Masayoshi Tomizuka, Marco Pavone, Tianbao Yang, Jing Du, Ming-Hsuan Yang, Hua Wei, Ziran Wang, Yang Zhou, Jiachen Li, and Zhengzhong Tu. Generative ai for autonomous driving: Frontiers and opportunities, 2025. URL <https://arxiv.org/abs/2505.08854>.
- [65] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 17853–17862, 2023.
- [66] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8340–8350, 2023.
- [67] Xinshuo Weng, Boris Ivanovic, Yan Wang, Yue Wang, and Marco Pavone. Para-drive: Parallelized architecture for real-time autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15449–15458, 2024.
- [68] Wenzhao Zheng, Ruiqi Song, Xianda Guo, Chenming Zhang, and Long Chen. Genad: Generative end-to-end autonomous driving. In *European Conference on Computer Vision*, pages 87–104. Springer, 2024.
- [69] 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. *arXiv preprint arXiv:2411.15139*, 2024.
- [70] Zhiqi Li, Zhiding Yu, Shiyi Lan, Jiahao Li, Jan Kautz, Tong Lu, and Jose M Alvarez. 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*, pages 14864–14873, 2024.
- [71] Bu Jin, Xinyu Liu, Yupeng Zheng, Pengfei Li, Hao Zhao, Tong Zhang, Yuhang Zheng, Guyue Zhou, and Jingjing Liu. Adapt: Action-aware driving caption transformer. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7554–7561. IEEE, 2023.

- [72] SungYeon Park, MinJae Lee, JiHyuk Kang, Hahyeon Choi, Yoonah Park, Juhwan Cho, Adam Lee, and DongKyu Kim. Vlaad: Vision and language assistant for autonomous driving. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 980–987, 2024.
- [73] Kairui Ding, Boyuan Chen, Yuchen Su, Huan-ang Gao, Bu Jin, Chonghao Sima, Wuqiang Zhang, Xiaohui Li, Paul Barsch, Hongyang Li, et al. Hint-ad: Holistically aligned interpretability in end-to-end autonomous driving. *arXiv preprint arXiv:2409.06702*, 2024.
- [74] Ana-Maria Marcu, Long Chen, Jan Hünemann, Alice Karnsund, Benoit Hanotte, Prajwal Chidananda, Saurabh Nair, Vijay Badrinarayanan, Alex Kendall, Jamie Shotton, et al. Lingoqa: Visual question answering for autonomous driving. In *European Conference on Computer Vision*, pages 252–269. Springer, 2024.
- [75] Yiheng Li, Cunxin Fan, Chongjian Ge, Zhihao Zhao, Chenran Li, Chenfeng Xu, Huaxiu Yao, Masayoshi Tomizuka, Bolei Zhou, Chen Tang, Mingyu Ding, and Wei Zhan. Womd-reasoning: A large-scale dataset for interaction reasoning in driving, 2025. URL <https://arxiv.org/abs/2407.04281>.
- [76] Yi Xu, Yuxin Hu, Zaiwei Zhang, Gregory P Meyer, Siva Karthik Mustikovela, Siddhartha Srinivasa, Eric M Wolff, and Xin Huang. Vlm-ad: End-to-end autonomous driving through vision-language model supervision. *arXiv preprint arXiv:2412.14446*, 2024.
- [77] Deepti Hegde, Rajeev Yasarla, Hong Cai, Shizhong Han, Apratim Bhattacharyya, Shweta Mahajan, Litian Liu, Risheek Garrepalli, Vishal M Patel, and Fatih Porikli. Distilling multimodal large language models for autonomous driving. *arXiv preprint arXiv:2501.09757*, 2025.
- [78] Pei Liu, Haipeng Liu, Haichao Liu, Xin Liu, Jinxin Ni, and Jun Ma. Vlm-e2e: Enhancing end-to-end autonomous driving with multimodal driver attention fusion. *arXiv preprint arXiv:2502.18042*, 2025.
- [79] Haicheng Liao, Hanlin Kong, Bonan Wang, Chengyue Wang, Wang Ye, Zhengbing He, Chengzhong Xu, and Zhenning Li. Cot-drive: Efficient motion forecasting for autonomous driving with llms and chain-of-thought prompting. *arXiv preprint arXiv:2503.07234*, 2025.
- [80] Jiageng Mao, Yuxi Qian, Junjie Ye, Hang Zhao, and Yue Wang. Gpt-driver: Learning to drive with gpt. *arXiv preprint arXiv:2310.01415*, 2023.
- [81] Songyan Zhang, Wenhui Huang, Zihui Gao, Hao Chen, and Chen Lv. Wisead: Knowledge augmented end-to-end autonomous driving with vision-language model. *arXiv preprint arXiv:2412.09951*, 2024.
- [82] Ran Tian, Boyi Li, Xinshuo Weng, Yuxiao Chen, Edward Schmerling, Yue Wang, Boris Ivanovic, and Marco Pavone. Tokenize the world into object-level knowledge to address long-tail events in autonomous driving. *arXiv preprint arXiv:2407.00959*, 2024.
- [83] Zhijie Qiao, Haowei Li, Zhong Cao, and Henry X. Liu. Lightemma: Lightweight end-to-end multimodal model for autonomous driving, 2025. URL <https://arxiv.org/abs/2505.00284>.
- [84] Wenru Liu, Pei Liu, and Jun Ma. Dsdrive: Distilling large language model for lightweight end-to-end autonomous driving with unified reasoning and planning, 2025. URL <https://arxiv.org/abs/2505.05360>.
- [85] Derun Li, Jianwei Ren, Yue Wang, Xin Wen, Pengxiang Li, Leimeng Xu, Kun Zhan, Zhongpu Xia, Peng Jia, Xianpeng Lang, et al. Finetuning generative trajectory model with reinforcement learning from human feedback. *arXiv preprint arXiv:2503.10434*, 2025.
- [86] Hao Gao, Shaoyu Chen, Bo Jiang, Bencheng Liao, Yiang Shi, Xiaoyang Guo, Yuechuan Pu, Haoran Yin, Xiangyu Li, Xinbang Zhang, et al. Rad: Training an end-to-end driving policy via large-scale 3dgs-based reinforcement learning. *arXiv preprint arXiv:2502.13144*, 2025.

- [87] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741, 2023.
- [88] Zhejun Zhang, Peter Karkus, Maximilian Igl, Wenhao Ding, Yuxiao Chen, Boris Ivanovic, and Marco Pavone. Closed-loop supervised fine-tuning of tokenized traffic models. *arXiv preprint arXiv:2412.05334*, 2024.
- [89] Jonah Philion, Xue Bin Peng, and Sanja Fidler. Trajenglish: Traffic modeling as next-token prediction. *arXiv preprint arXiv:2312.04535*, 2023.
- [90] Wei Wu, Xiaoxin Feng, Ziyang Gao, and Yuheng Kan. Smart: scalable multi-agent real-time motion generation via next-token prediction. *Advances in Neural Information Processing Systems*, 37:114048–114071, 2024.
- [91] Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.
- [92] Sung-Yeon Park, Can Cui, Yunsheng Ma, Ahmadreza Moradipari, Rohit Gupta, Kyungtae Han, and Ziran Wang. Nuplanqa: A large-scale dataset and benchmark for multi-view driving scene understanding in multi-modal large language models. *arXiv preprint arXiv:2503.12772*, 2025.
- [93] Dongming Wu, Wencheng Han, Yingfei Liu, Tiancai Wang, Cheng-zhong Xu, Xiangyu Zhang, and Jianbing Shen. Language prompt for autonomous driving. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 8359–8367, 2025.
- [94] Tianwen Qian, Jingjing Chen, Linhai Zhuo, Yang Jiao, and Yu-Gang Jiang. Nuscenes-qa: A multi-modal visual question answering benchmark for autonomous driving scenario. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 4542–4550, 2024.
- [95] Ming Nie, Renyuan Peng, Chunwei Wang, Xinyue Cai, Jianhua Han, Hang Xu, and Li Zhang. Reason2drive: Towards interpretable and chain-based reasoning for autonomous driving. In *European Conference on Computer Vision*, pages 292–308. Springer, 2024.
- [96] 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 *European Conference on Computer Vision*, pages 256–274. Springer, 2024.
- [97] OpenScene Contributors. Openscene: The largest up-to-date 3d occupancy prediction benchmark in autonomous driving. <https://github.com/OpenDriveLab/OpenScene>, 2023.
- [98] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.
- [99] Jiang-Tian Zhai, Ze Feng, Jinhao Du, Yongqiang Mao, Jiang-Jiang Liu, Zichang Tan, Yifu Zhang, Xiaoqing Ye, and Jingdong Wang. Rethinking the open-loop evaluation of end-to-end autonomous driving in nuscenes. *arXiv preprint arXiv:2305.10430*, 2023.
- [100] 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. *Advances in Neural Information Processing Systems*, 35:6119–6132, 2022.
- [101] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7953–7963, 2023.

- [102] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.
- [103] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9710–9719, 2021.

NeurIPS Paper Checklist

1. Claims

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

Answer: [\[Yes\]](#)

Justification: We do describe our contributions in the abstract and introduction of the paper: 1) a novel end-to-end autonomous driving framework leveraging a trained VLM backbone augmented with integrated physical action tokens, enabling direct policy learning and semantic reasoning from raw visual inputs. 2) an RL-based fine-tuning method using GRPO to enable adaptive decision-making and further enhance AutoVLA's performance on end-to-end driving tasks.

Guidelines:

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

2. Limitations

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

Answer: [\[Yes\]](#)

Justification: We discuss the real-world implementation limitation of AutoVLA and introduce our future work on model quantization, as Section 5 shown.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

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

Answer: [NA]

Justification: This paper focuses on the experimental exploration of model architectures, and have no theoretical results in the paper.

Guidelines:

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

4. Experimental result reproducibility

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

Answer: [Yes]

Justification: All the necessary information for reproducibility is provided in Section 4 and the Supplementary Material. The model details also are discussed in the methodology of the paper.

Guidelines:

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

- (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

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

Answer: [Yes]

Justification: We will release our code and dataset to facilitate community research.

Guidelines:

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

6. Experimental setting/details

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

Answer: [Yes]

Justification: We provide all the implementation details and dataset settings of AutoVLA in both the SFT and RFT stages in the paper Section 4 and supplementary.

Guidelines:

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

7. Experiment statistical significance

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

Answer: [No]

Justification: The paper reports results from a single run per experiment, consistent with prior work on this task.

Guidelines:

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

8. Experiments compute resources

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

Answer: [Yes]

Justification: The paper provides all necessary details on computational resources, including GPU type (NVIDIA L40S), memory usage, and number of epochs, in Section 4.

Guidelines:

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

9. Code of ethics

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

Answer: [Yes]

Justification: The research presented in this paper adheres to the NeurIPS Code of Ethics, ensuring ethical standards in data collection and analysis.

Guidelines:

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

10. Broader impacts

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

Answer: [Yes]

Justification: The paper includes a discussion of potential societal impacts in the supplementary material.

Guidelines:

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

11. Safeguards

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

Answer: [NA]

Justification: The paper poses no foreseeable risks.

Guidelines:

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

12. Licenses for existing assets

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

Answer: [Yes]

Justification: The original creators and asset owners are appropriately credited, and licenses and terms of use are explicitly stated and duly respected.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.

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

13. **New assets**

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

Answer: [Yes]

Justification: We will release all the code, model weights, and datasets to facilitate research in the domain.

Guidelines:

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

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

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

Answer: [NA]

Justification: The paper does not involve crowdsourcing or human-subject research.

Guidelines:

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

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

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

Answer: [NA]

Justification: The paper does not involve crowdsourcing or human-subject research.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

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

Answer: [Yes]

Justification: We declare the usage of LLM/VLM in the paper, and we integrate the physical action tokens into the pretrained VLM backbone to develop a novel end-to-end autonomous driving framework.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.



A Action Tokenization

A.1 Action Codebook

To enable trajectory-level planning within a language model, we introduce **physical action tokens**, each representing a short-term feasible vehicle maneuver. These tokens are derived from clustering vehicle motion patterns in the Waymo Open Motion Dataset (WOMD) [103], which contains extensive real-world vehicle trajectories. The resulting codebook, visualized in Fig. S1(a), is used across experiments involving multiple real-world datasets, including nuPlan, nuScenes, and Waymo. Due to differences in vehicle dynamics compared to real-world data for simulation testing, we construct a separate action codebook using the same clustering procedure on the CARLA-Garage dataset [55]. This produces another set of 2048 action tokens for simulation testing, as shown in Fig. S1(b).

We begin by sampling short motion segments from the dataset to construct a **discrete action codebook**. Each segment represents 0.5 seconds of vehicle motion, characterized by its final-frame bounding-box contour, computed based on the vehicle's position, dimensions, and heading with respect to the current-frame coordinate. Then, we apply K-Disk clustering on these sampled segments, which iteratively selects a diverse set of representative segments $\{m_1, \dots, m_K\}$, such that no two segments are within a distance threshold $\delta = 0.05$ m, measured using average contour distance. For each selected segment m_k , we extract its spatial displacement and heading change, denoted as $(\Delta x, \Delta y, \Delta \theta)$, and define it as the action token a_k . The resulting action codebook is $\mathcal{A} = \{a_1, \dots, a_K\}$, with $K = 2048$, where each token encodes a distinct and physically feasible short-term vehicle behavior.

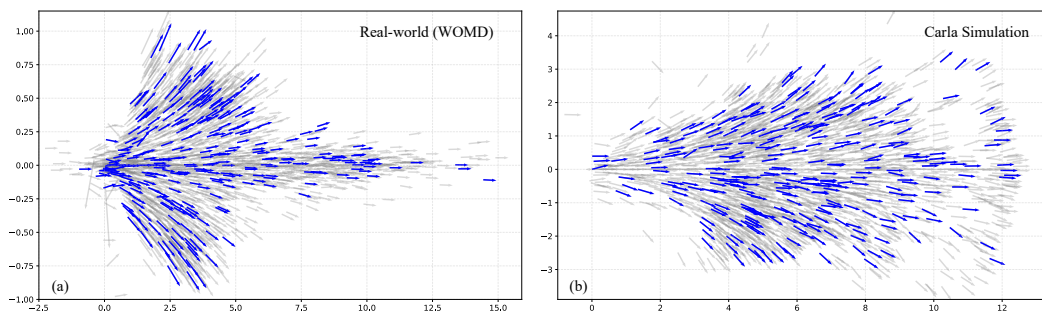


Figure S1: Visualization of action codebooks from (a) real-world dataset (WOMD) and (b) simulation dataset (CARLA). Grey arrows represent all 2048 tokens (position and heading), with 300 tokens randomly highlighted in blue for clarity.

A.2 Action Tokenizer

We implement an action tokenizer based on the constructed action codebook. During training, continuous trajectories are discretized by mapping each 0.5-second segment to its nearest action token in the codebook $\mathcal{A}$, resulting in a sequence of discrete tokens $[a_1, a_2, \dots, a_T]$. During inference, the language model autoregressively generates a sequence of action tokens, each representing a 0.5-second motion segment. These tokens are then converted back to their corresponding motion segments in the codebook and applied sequentially from the initial ego pose. Therefore, the model reconstructs a continuous trajectory in the ego-centric coordinate frame by composing these local displacements and rotations.

B Reasoning Data Collection

A large-scale, high-quality reasoning dataset with chain-of-thought (CoT) annotations is essential for enabling robust reasoning capabilities in vision-language-action (VLA) models. In this paper, we introduce an automated reasoning annotation pipeline using the state-of-the-art Qwen2.5-VL-72B vision-language model [21], as illustrated in Fig. S2. The pipeline significantly reduces reliance on

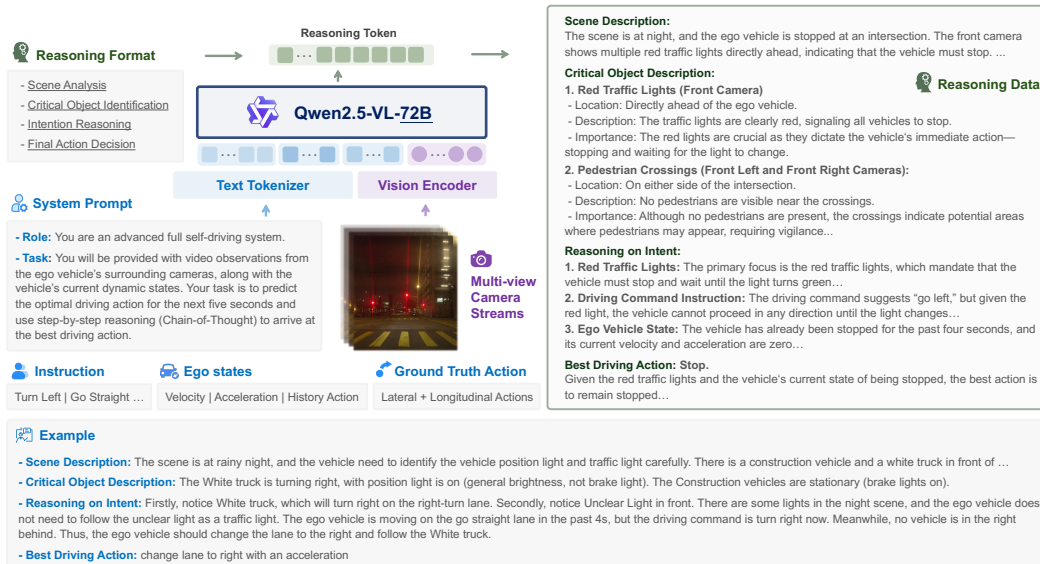


Figure S2: Automated reasoning annotation pipeline for autonomous driving, illustrated with a reasoning annotation example from the Waymo end-to-end driving dataset.

human annotations and facilitates effective knowledge distillation from a more powerful, large-scale model to a more efficient, compact model.

B.1 Reasoning Annotation Pipeline

System Prompt. The system prompt specifies the model's role, task, expected CoT reasoning format, and examples of CoT reasoning. The definition of role and CoT reasoning format aligns with the required outputs of the AutoVLA model, which focuses on structured CoT reasoning. We carefully design several representative reasoning examples to guide the model. Moreover, the reasoning process includes four main steps: 1) scene description and analysis, 2) critical object identification and description, 3) intention reasoning of the surrounding objects, and 4) decision-making and meta-action.

User Message. The user message includes the driving instructions, ego vehicle states, and multi-view camera streams. Notably, we introduce the ground-truth driving meta-action derived from the data as explicit hints in reasoning, guiding the model to produce causal explanations that directly associate decisions with the driving context. This structured prompting significantly reduces nonsensical outputs and minimizes manual revisions.

Reasoning Data Generation. We employ the most capable model in the Qwen-VL series as the reasoning data annotation engine, leveraging its strong reasoning capacity and extensive world knowledge. The maximum length for newly generated reasoning textual tokens is set to 700. Additionally, we extract answers to relevant questions from the DriveLM VQA dataset [96], which focuses on perception, prediction, and planning in the nuScenes and CARLA datasets, and then we reformat them into our standardized reasoning format. These reformatted samples are then combined with the generated data to construct the final reasoning dataset.

Human Quality Check. We evaluate the quality of generated reasoning based on the accuracy and completeness of critical object identification, causal reasoning, and action decisions. The evaluation uses a binary scoring scheme: any error in these aspects results in a score of 0; otherwise, the sample receives a score of 1. Human annotators assessed 3,000 randomly selected samples, yielding an overall accuracy of 88.8%, which demonstrates the high reliability of our proposed annotation pipeline. Erroneous samples were either corrected or discarded.

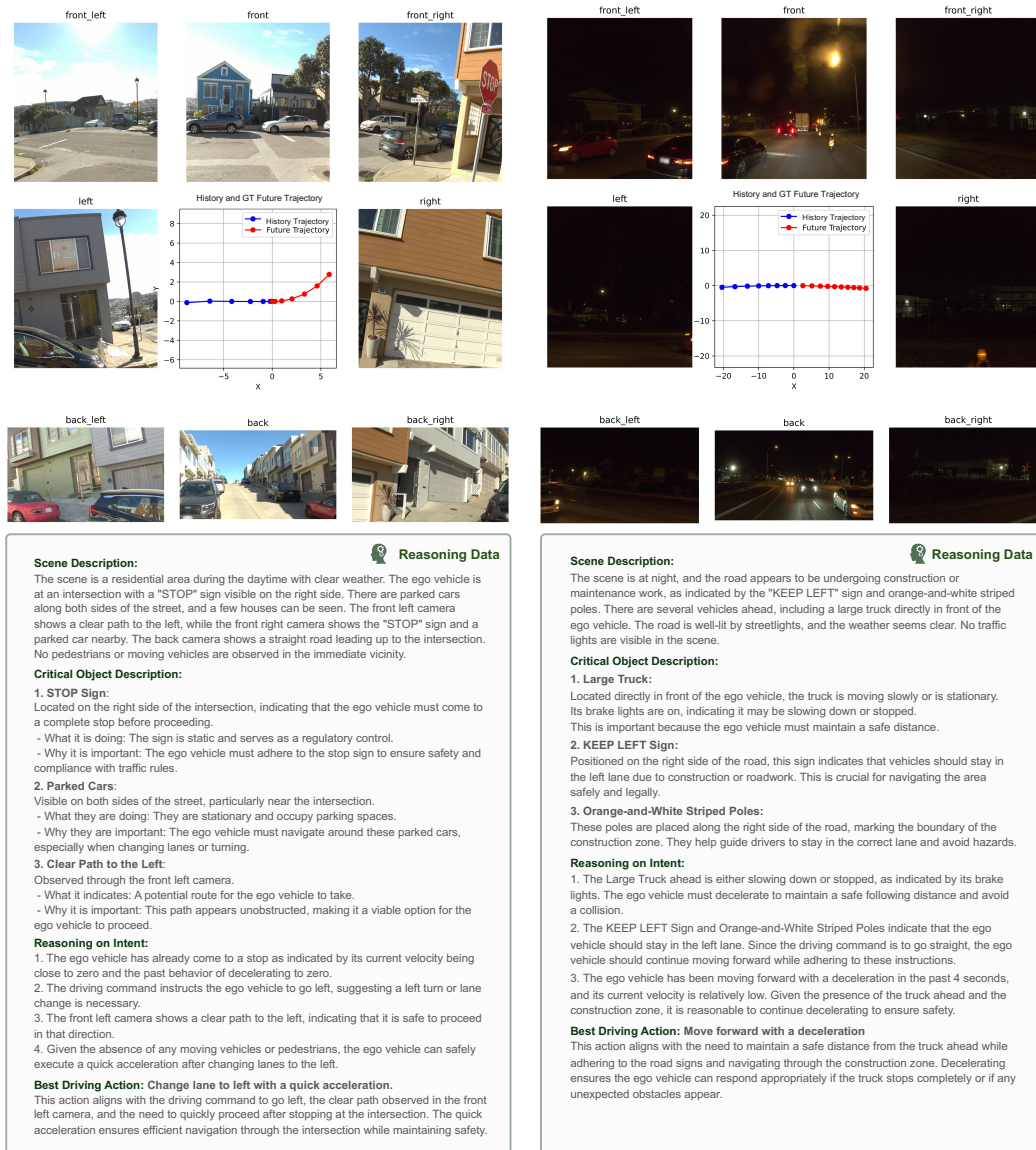


Figure S3: Visualization of the reasoning data annotation on the Waymo end-to-end driving dataset.

B.2 Reasoning Annotation Visualization

Some CoT reasoning annotation examples from the Waymo end-to-end driving dataset are shown in Fig. S3, showcasing that our annotation pipeline can deal with situations where a vehicle already stopped at a stop sign can proceed, rather than remaining stopped indefinitely. The pipeline also accurately interprets construction-related road control scenarios, enabling high-quality reasoning annotations. Examples on the nuPlan dataset are provided in Fig. S4, demonstrating the pipeline's capability to distinguish the functional relevance of stop signs and traffic lights across different lanes, rather than stopping at every detected stop sign.

C Details of Supervised Fine-tuning

We perform supervised fine-tuning (SFT) with the Fully Sharded Data Parallel (FSDP) strategy for efficient multi-GPU training. We enable mixed-precision training using BFloat16 for parameters,

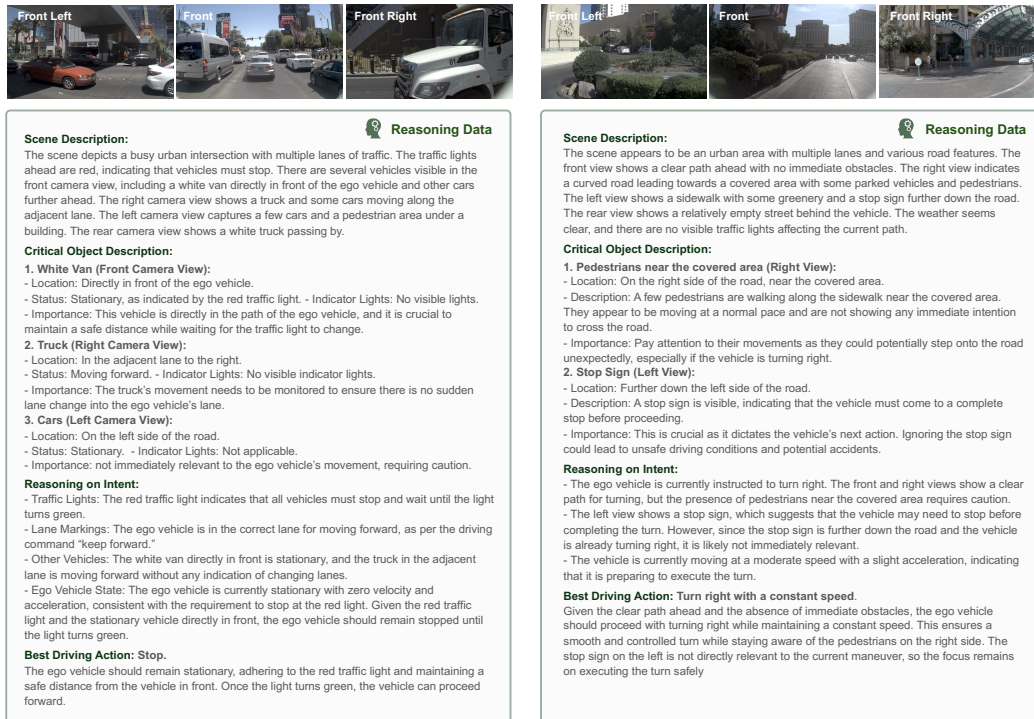


Figure S4: Visualization of the reasoning data annotation on the nuPlan dataset.

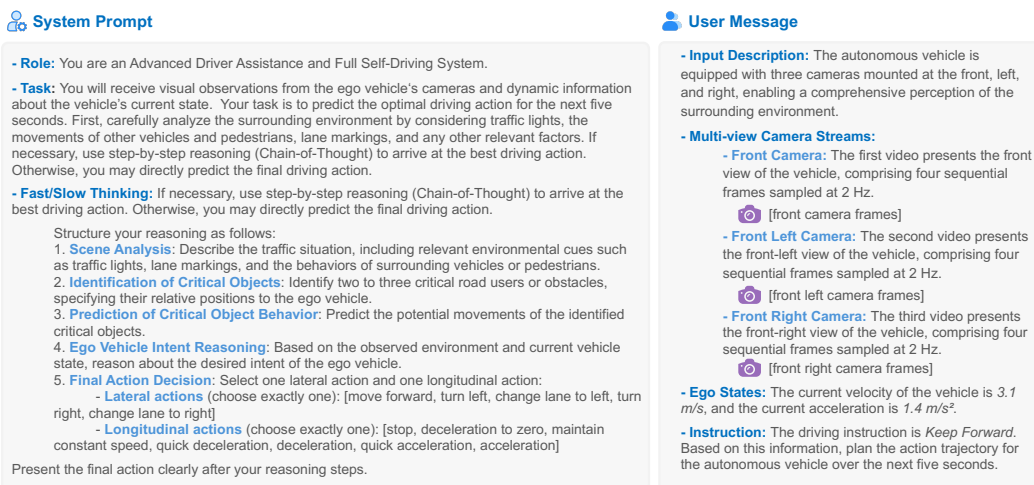


Figure S5: System Prompt and User Message of AutoVLA.

gradients, and buffers to reduce memory usage and accelerate computation. Gradient checkpointing is enabled to reduce GPU memory consumption. The learning rate warm-up is 500 steps and decays by 2% every 2,000 steps. The model is trained for 5 epochs, with gradient clipping applied at a maximum value of 1.0 to ensure training stability.

The model takes both system and user prompts as context inputs. As illustrated in Fig. S5, the system prompt defines the model's role, task, and the expected CoT reasoning format. The user message describes the multi-view camera observations, the ego vehicle's current state (i.e., speed and acceleration) and optionally historical states, and the high-level driving instruction. We use three camera views (front, front-left, and front-right), each providing four consecutive frames. Images in the video stream are resized to maintain original aspect ratios but reduced to $28 \times 28 \times 128$ pixels.

Algorithm 1 RFT for AutoVLA with GRPO

Input: Supervised fine-tuned policy π_{SFT} , Action Codebook $\mathcal{A}$, Group size G , Training step K , Dataset D , Driving reward function r_{Driving} , Reasoning reward function r_{CoT} , Balance weight λ_r , KL regularization weight β

Output: Reinforcement fine-tuned policy π_{RFT}

```
1: Initialize reference policy  $\pi_{\text{ref}} \leftarrow \pi_{\text{SFT}}$  and current policy  $\pi_\theta \leftarrow \pi_{\text{SFT}}$ 
2: for training step 1 to  $K$  do
3:   Sample scenario  $U$  from  $D$ 
4:   for sample  $i$  from 1 to  $G$  do
5:     Sample input query  $q$ , final output  $o_i$ , and per-token probability  $\pi_\theta(o_i|q)$ 
6:      $\pi_{\theta_{\text{old}}}(o_i|q) \leftarrow \pi_\theta(o_i|q)$ 
7:     Calculate per-token probability  $\pi_{\text{ref}}(o_i|q)$  of  $\pi_{\text{ref}}$  under  $o_i$ 
8:     Decode trajectory  $\tau \leftarrow \mathcal{A}(o_i)$ 
9:     Calculate reward  $r_i \leftarrow r_{\text{Driving}}(\tau, U) - \lambda_r r_{\text{CoT}}(o_i)$ 
10:   end for
11:   Group average reward  $\bar{r}$ ,  $\sigma_r \leftarrow \text{mean}(r_1, \dots, r_G)$ ,  $\text{std}(r_1, \dots, r_G)$ 
12:   Group relative advantage  $A_i \leftarrow (r_i - \bar{r}) / (\sigma_r)$ 
13:    $\mathcal{L}_{\text{RFT}} \leftarrow -\sum_i^G \left( -\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{\text{old}}}(o_i|q)} A_i + \beta \mathbb{D}_{\text{KL}}(\pi_\theta(o_i|q) \parallel \pi_{\text{ref}}(o_i|q)) \right)$ 
14:   Optimize  $\pi_\theta$  according to  $\mathcal{L}_{\text{RFT}}$ 
15: end for
16: return Converged policy  $\pi_{\text{RFT}}$ 
```

D Details of Reinforcement Fine-tuning

Group Relative Policy Optimization (GRPO) employs group-based sampling to compute the advantage function, replacing conventional state-value estimators or critic models. This design accelerates training while aligning naturally with the inherent multimodality of planning, which requires evaluating and selecting from a set of candidate trajectories. The overall reinforcement fine-tuning (RFT) procedure is illustrated in Algorithm 1.

D.1 Kullback-Leibler (KL) Divergence

The KL divergence term $\mathbb{D}_{\text{KL}}$ is incorporated in Eq. (3) to regularize the current policy with respect to the reference policy:

$$\mathbb{D}_{\text{KL}}(\pi_\theta \parallel \pi_{\text{ref}}) = \frac{\pi_{\text{ref}}(o_i|q)}{\pi_\theta(o_i|q)} - \log \left(\frac{\pi_{\text{ref}}(o_i|q)}{\pi_\theta(o_i|q)} \right) - 1, \quad (\text{S1})$$

where θ denotes the parameters of the current policy π_θ , q is the scenario input query, o_i is the output of i th sample in the group, and π_{ref} is the reference policy from the SFT stage. This regularization term penalizes large deviations from the reference policy, ensuring that policy updates remain within a stable area. As a result, the model retains useful knowledge acquired during SFT while improving driving behavior through reinforcement learning guided by validated reward functions.

D.2 Reward Function

The reward function provides the primary RFT training signal, guiding the optimization of policy updates. To further enhance comprehensive driving performance beyond imitating expert driving, we adopt a hybrid reward design that integrates the primary driving reward r_{Driving} and a CoT penalty r_{CoT} to discourage lengthy reasoning. For the NAVSIM Benchmark, we employ Predictive Driver Model Score (PDMS) [51] as the primary reward. For the Waymo end-to-end dataset, due to the limited annotation of RFS, we employ the (normalized) Average Displacement Error (ADE) as the driving reward. The final reward function, balanced by a weighting coefficient λ_r , is defined as:

$$r = r_{\text{Driving}} - \lambda_r r_{\text{CoT}}. \quad (\text{S2})$$

In the nuPlan dataset, we adopt the PDMS score as the driving reward signal r_{Driving} , which serves as a comprehensive measure of driving quality. It is defined as follows:

$$r_{\text{Driving}} = \text{PDMS} = \text{NC} \times \text{DAC} \times \left(\frac{5\text{TTC} + 2\text{C} + 5\text{EP}}{12} \right), \quad (\text{S3})$$

where the components include No at-fault Collision (NC), Drivable Area Compliance (DAC), Ego Progress (EP), Comfort (C), and Time-to-Collision (TTC). Detailed descriptions of each component can be found in [51]. Each sub-score is expressed as a percentage, and their weighted combination yields a normalized score within the range $[0, 1]$, reflecting aspects of safety, comfort, and progress. For any scenario in which the planner fails (due to output errors), we assign a score of 0.

For the Waymo dataset, we define the driving reward r_{Driving} based on the ADE metric:

$$r_{\text{Driving}} = \frac{\delta - \text{ADE}}{\kappa}, \quad \text{ADE} = \frac{1}{T} \sum_{t=1}^T \|\hat{\mathbf{y}}_t - \mathbf{y}_t\|_2, \quad (\text{S4})$$

where δ denotes the maximum displacement error, and κ is a scaling factor to normalize the reward. The planning trajectory $\hat{\mathbf{y}}$ is evaluated against the ground truth trajectory $\mathbf{y}$, and ADE is computed as the average L2 distances over T time steps.

To penalize excessively long reasoning chains, we introduce a CoT penalty term r_{CoT} , defined via a sigmoid function that normalizes the length of reasoning:

$$r_{\text{CoT}} = \frac{1}{1 + e^{-(L - L_{\text{tol}})\gamma}}, \quad (\text{S5})$$

where L denotes the length of the CoT reasoning, L_{tol} is the tolerance threshold, and γ is a scaling coefficient that controls the steepness of the penalty curve.

D.3 Implementation Details

We use the navtrain split of the nuPlan dataset for RFT for the NAVSIM benchmark and the validation split of the Waymo end-to-end driving dataset for the Waymo benchmark. The vision encoder of pretrained AutoVLA is frozen, and the model is fine-tuned using Low-Rank Adaptation (LoRA) to reduce training costs and memory consumption. Specifically, both the LoRA rank and alpha are set to 8, with a dropout rate of 0.1. The pretrained SFT model is used as the reference policy during optimization. The hyperparameters γ , L_{tol} , and learning rate are set to 2×10^{-3} , 400, and 3×10^{-5} , respectively. To ensure that the driving reward signal is dominant, the regularization weight is set to a relatively small value, $\lambda_r = 0.3$. We also set $\delta = 2$, $\kappa = 10$ in the RFT of the Waymo dataset. Moreover, we configure the generation parameters with a sampling temperature of 1.0, top-p of 1.0, and top-k of 0.0 to encourage diverse and exploratory generation during GRPO sampling, which effectively covers a wider range of possible actions.

E Experiment Details

E.1 Data Preprocessing

To enable mixed training across multiple driving datasets, we develop a unified data preprocessing pipeline that standardizes the format across all datasets. For each sample, we extract and standardize: 1) Ground truth trajectory coordinates and headings in the ego vehicle's coordinate frame at 2 Hz, 2) Image paths for multi-view camera image sequences consisting of 4 consecutive frames captured at 2 Hz (providing 2 seconds of history), 3) CoT reasoning annotations, 4) Vehicle states including its current velocity and acceleration, 5) High-level driving instructions. The preprocessing pipeline handles dataset-specific differences in data format, sampling rates, and coordinate systems to create a consistent format. The size and distribution of the final formatted dataset are shown in Table S1.

nuPlan (NAVSIM). We randomly sample 45.6k scenarios from the nuPlan trainval split and generate reasoning data using our proposed automated annotation pipeline. The resulting reasoning samples, together with the remaining training data with only trajectory annotation, constitute our full training set of nuPlan. Following the NAVSIM benchmark, the navtest split is used as the test set.

Table S1: Training and Testing Data Size for Different Datasets

Dataset	Train Samples	Reasoning Samples	Test Samples
nuPlan (NAVSIM) [51]	166.3k	45.6k	12.1k
nuScenes [53]	19.0k	2.9k	5.6k
Waymo [52]	23.8k	7.2k	1.5k
CARLA [96]	274.5k	53.2k	–

nuScenes. We preprocess all samples from the training set. For samples included in the DriveLM dataset, we reformat the question-answer (QA) pairs to generate structured reasoning annotations following our four-step reasoning format. Samples not covered by DriveLM are also used for training but only with trajectory supervision. The validation set is used for testing.

Waymo. The Waymo end-to-end driving dataset provides 2037 training and 479 validation segments, each containing a 20-second video with driving logs for the entire duration. We sample reasoning data using a 4-second sliding window and extract trajectory-only data using a 2-second sliding window offset by 1 second from the reasoning samples. Due to noise in the position data, the estimated vehicle heading can exhibit abrupt fluctuations when the vehicle is stationary. To address this issue, we apply a motion threshold to detect stationary periods and smooth the heading accordingly. The test set comprises 1505 samples.

CARLA. The CARLA-Garage dataset is used to train the model for closed-loop evaluation. Since only front camera images are available, we use single-view inputs (with four consecutive frames) for CARLA training and testing. We sample data using a sliding window with an offset of 0.5 seconds, and downsample the trajectories from 4 Hz to 2 Hz. For reasoning annotations, we leverage the DriveLM-CARLA dataset, which provides QA pairs similar to DriveLM-nuScenes, and we reformat the QA pairs to generate samples with reasoning annotations.

E.2 Evaluation Metrics

nuPlan (NAVSIM). PDMS is the official benchmark metric in NAVSIM on the nuPlan dataset, with its formulation provided in Eq. (S3).

nuScenes. Following previous works, we adopt the **L2 Distance** to measure the average displacement error between predicted and ground-truth trajectories, and use **Collision Rate** to assess the frequency of predicted trajectories overlapping with surrounding objects. We follow the UniAD protocol, reporting both metrics at 1, 2, and 3 seconds in the future, instead of averaging over the horizons.

Waymo. Following the Waymo end-to-end driving benchmark, **Rater Feedback Score (RFS)** is used to evaluate the driving performance. The Waymo dataset provides three human-annotated trajectories, each associated with a scalar quality score in $[0, 10]$. A planning trajectory is matched to its closest reference trajectory, and a trust region is defined around it using lateral and longitudinal thresholds at fixed timesteps (3 s and 5 s). These thresholds are scaled by a piecewise linear function of the rater’s trajectory’s initial speed. Predictions within the trust region inherit the matched rater’s score, and those outside are penalized exponentially based on normalized deviation.^{tr}

CARLA. We evaluate driving performance in closed-loop testing using four key metrics: **Driving Score**, **Success Rate**, **Efficiency**, and **Comfortness** following the Bench2Drive benchmark [51]. Driving Score reflects the overall route completion penalized by infraction severity. Success Rate measures the percentage of routes completed without major infractions within the time limit. Efficiency quantifies how well the ego vehicle maintains a reasonable speed relative to surrounding traffic, computed as the average speed ratio over multiple checkpoints. Comfortness assesses the smoothness of the trajectory based on physical signals such as acceleration, jerk, and yaw rate, using expert-derived thresholds.

E.3 Model Training

Our primary base model is trained on the mixed training set of the nuPlan and nuScenes datasets. The ego state contains the vehicle’s current velocity and acceleration. For the test on the Waymo end-to-end driving dataset, we further fine-tune this base model using the Waymo end-to-end driving



Figure S6: Planning and reasoning results of AutoVLA on the nuScenes dataset.

Table S2: Testing Results of AutoVLA on the nuScenes Planning Benchmark

Method	ST-P3 metrics								UniAD metrics							
	L2 (m) ↓				Collision (%) ↓				L2 (m) ↓				Collision (%) ↓			
	1s	2s	3s	Avg.	1s	2s	3s	Avg.	1s	2s	3s	Avg.	1s	2s	3s	Avg.
ST-P3 [14]	1.33	2.11	2.90	2.11	0.23	0.62	1.27	0.71	-	-	-	-	-	-	-	-
VAD [66]	0.17	0.34	0.60	0.37	0.07	0.10	0.24	0.14	-	-	-	-	-	-	-	-
UniAD [65]	0.44	0.67	0.96	0.69	0.04	0.08	0.23	0.12	0.48	0.96	1.65	1.03	0.05	0.17	0.71	0.31
EMMA [37]	0.14	0.29	0.54	0.32	-	-	-	-	-	-	-	-	-	-	-	-
OpenEMMA [44]	1.45	3.21	3.76	2.81	-	-	-	-	-	-	-	-	-	-	-	-
OpenDriveVLA-3B [34]	0.14	0.30	0.55	0.33	0.02	0.07	0.22	0.10	0.19	0.58	1.24	0.67	0.02	0.18	0.70	0.30
AutoVLA (action only)	0.22	0.39	0.61	0.41	0.10	0.17	0.28	0.18	0.29	0.67	1.17	0.71	0.15	0.34	0.56	0.35
AutoVLA (w/ CoT)	0.21	0.38	0.60	0.40	0.13	0.18	0.28	0.20	0.28	0.66	1.16	0.70	0.14	0.25	0.53	0.31

dataset. For the model trained on Waymo, the ego vehicle's state encompasses current acceleration and a 4-second history of vehicle positions and velocities. For closed-loop simulation testing, we train a separate model using data preprocessed from the CARLA Garage dataset and DriveLM-CARLA annotations. This model is trained with single-view inputs instead of the multi-view setup used in the primary model, and we use a large resolution for the input images (with $28 \times 28 \times 384$ pixels).

E.4 Inference

We employ a stochastic generation strategy using top-p and top-k sampling to generate reasoning and planning outputs. Higher sampling diversity (e.g., temperature=1.0, top-p=0.5, top-k=20) supports slow thinking modes, enabling the model to produce deeper and more elaborate reasoning chains. In contrast, more deterministic settings (e.g., temperature=0.1, top-p=0.01, top-k=1) produce fast thinking, yielding consistent and direct responses.



Figure S7: Planning and reasoning results of AutoVLA on the nuPlan dataset.

F Additional Results

F.1 nuScenes Results

We evaluate AutoVLA on the nuScenes dataset following both ST-P3 [14] and UniAD [65] protocols, in comparison with state-of-the-art end-to-end models. As shown in Table S2, our model demonstrated competitive performance in the nuScenes planning benchmark.

Fig. S6 shows AutoVLA’s planning outputs. The model generates safe trajectories that closely align with ground-truth motions, accompanied by coherent and context-aware reasoning outputs. However, it can be observed that many nuScenes scenarios are relatively straightforward, often not requiring complex reasoning. This may explain the lack of performance gain in quantitative metrics when reasoning is introduced.

F.2 nuPlan Results

We present additional visualization results of our model on the nuPlan dataset in Fig. S7. In relatively simple scenarios such as curved roads and intersections, our model generates high-quality trajectories via fast thinking. In more complex scenes with numerous traffic regulations, it leverages slow thinking

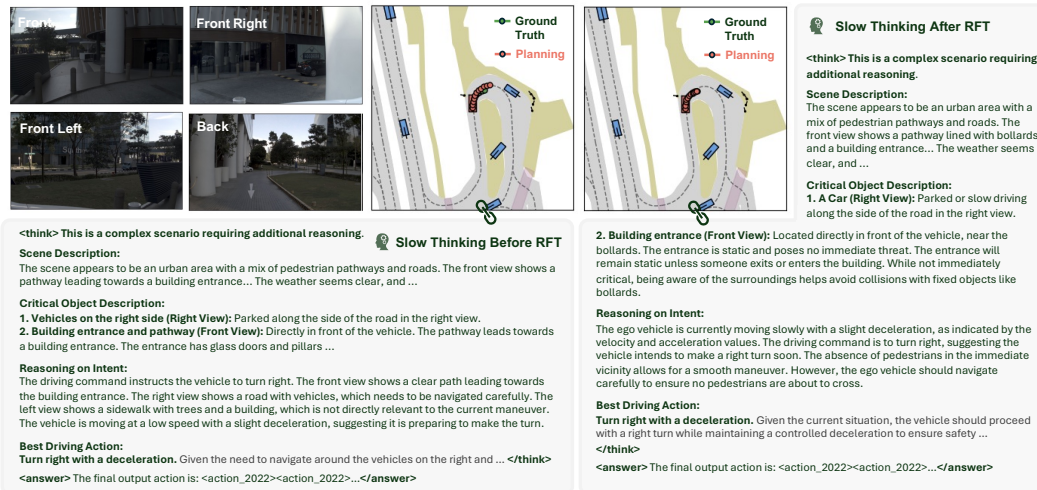


Figure S8: Qualitative comparison of planning and reasoning performance in complex scenarios, before and after RFT. Results indicate that RFT maintains reasoning capabilities in complex scenarios while enhancing planning performance.

Table S3: Waymo Vision-based End-to-End Driving Challenge Leaderboard

Method Name	RFS (Overall) ↑	ADE at 5s (Overall) ↓	ADE at 3s (Overall) ↓	RFS (Spotlight) ↑
Poutine	7.9860	2.7419	1.2055	6.8929
HMVLM	7.7367	3.0715	1.3269	6.7269
UniPlan	7.6925	2.9864	1.3083	6.6544
DiffusionLTF	7.5919	2.9768	1.3605	6.5688
<u>AutoVLA</u>	7.5566	2.9580	1.3507	6.9436
Swin-Trajectory	7.5432	2.8135	1.2082	6.6791
waymo	7.5281	3.0182	1.3200	6.5953
ViT-Adapter-GRU	7.4988	2.7024	1.1968	6.4543
DriveTraj	7.4957	2.9556	1.3038	6.4101
open-llama	7.4288	3.2165	1.3140	6.2510
MTR-VP	7.3433	3.3485	1.4232	6.4023
WayPredict-XL	7.2922	3.2915	1.4385	6.3083
DriveTraj	7.2787	3.4573	1.5346	6.2428
WayPredict	7.0641	3.5779	1.7242	5.8562
LightEMMA	6.5169	3.7395	1.7052	5.7103
FrozenResNet50	6.4719	3.9148	1.9446	5.7977
OpenEMMA	5.1575	12.4755	6.6842	4.7131

to produce better and regulation-compliant planning results with CoT reasoning. As illustrated in Fig. S8, our AutoVLA model preserves reasoning capabilities in complex scenarios despite removing redundant reasoning after RFT. Qualitative comparisons demonstrate that RFT enhances planning performance in complex scenarios.

E.3 Waymo Results

To overcome the limited size of the Waymo end-to-end driving dataset, we augment the training data with samples from the nuPlan and nuScenes datasets. For convenience, we refer to this combined dataset as **nuX**. We use this nuX data to pretrain the model before fine-tuning it on the Waymo dataset. To further enhance planning performance, we apply RFT on the AutoVLA model after SFT. Given the limited number of RFS-labeled samples (only 480 in the validation set), we adopt the average displacement error (ADE) as the primary reward signal.

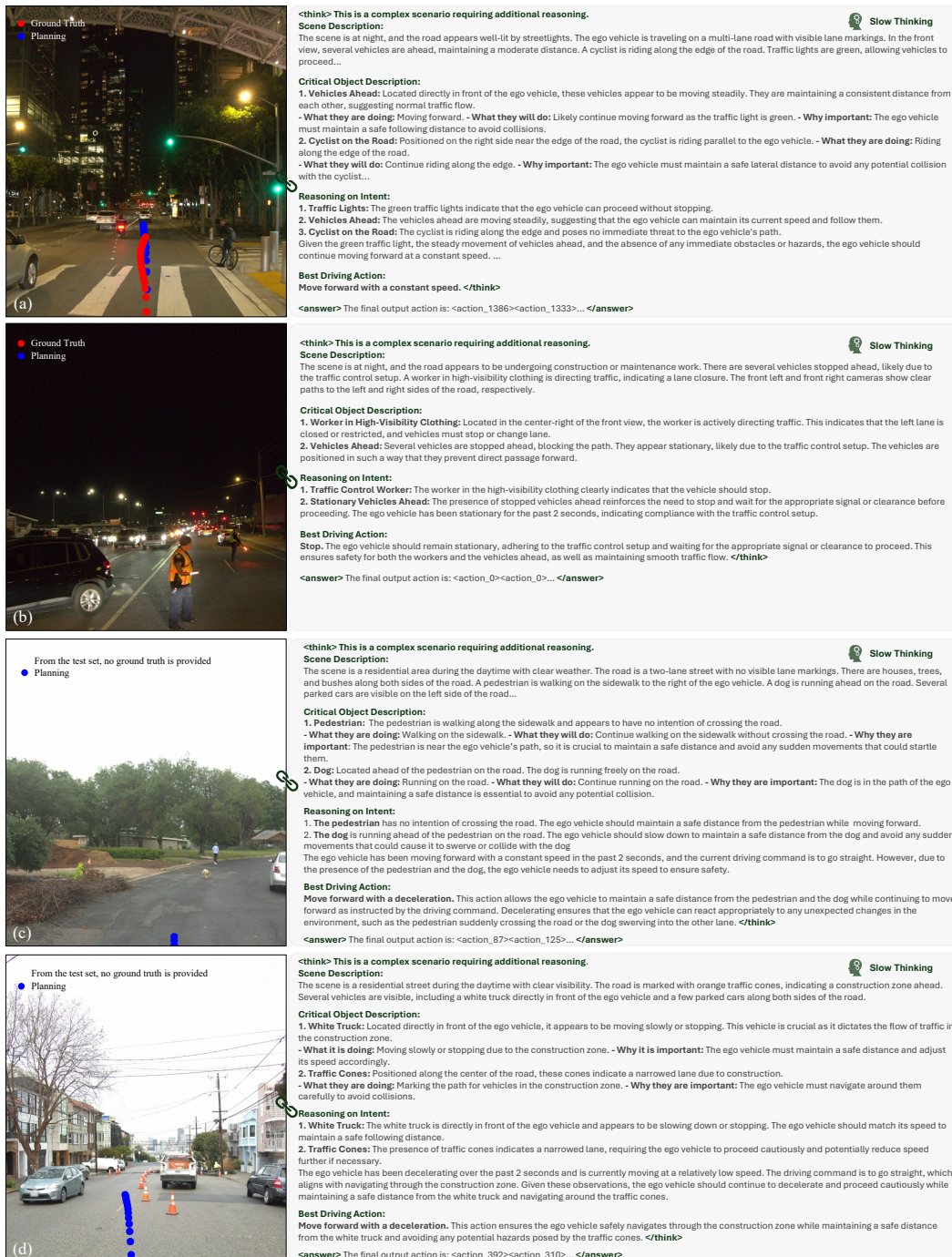


Figure S9: Planning and reasoning results of AutoVLA on the Waymo end-to-end driving dataset.

As of May 22, 2025, AutoVLA achieves competitive performance on the Waymo End-to-End Driving Challenge leaderboard, as shown in Table S3. The model ranks highly in both RFS Overall and ADE metrics and achieves the top score in the RFS Spotlight metric, which focuses on the most challenging scenarios. Qualitative results are shown in Fig. S9, where AutoVLA demonstrates its ability to generate safe, context-aware trajectories in complex environments. The model effectively handles interactions, construction zones, and traffic regulations in diverse scenarios while providing coherent reasoning to justify its decisions.

Table S4: Ablation Study on the Waymo End-to-End Driving Test Set

Camera	Pretraining	Output	RFS (Overall) $\uparrow$	ADE at 5s $\downarrow$
Front	None	Action-only	6.938	3.595
Front	None	CoT-enhanced	7.127	3.188
Multi	None	Action-only	7.239	3.243
Multi	None	CoT-enhanced	7.283	3.182
Multi	nuX	Action-only	7.406	3.116
Multi	nuX	CoT-enhanced	7.447	3.115
Post-RFT			7.557	2.958

Table S4 presents the ablation studies of the AutoVLA model in different training setups. The results indicate that multi-camera input consistently enhances driving performance. When trained solely on the Waymo end-to-end driving dataset, incorporating reasoning significantly improves performance compared to action-only setups. Moreover, pretraining on nuX data provides a substantial performance boost, suggesting that such pretraining enhances the scene understanding of the model with more driving data. RFT can significantly improve planning performance by mitigating error accumulation in generation and aligning with the task-specific rewards.

F.4 CARLA Results

We evaluate the model after SFT in closed-loop testing within the CARLA simulator. The replanning frequency is set to 2 Hz, meaning the AutoVLA model is queried every 0.5 seconds in simulation, and the planned trajectory is used to generate control commands. High-level driving instructions are derived from a predefined route plan, while the vehicle's current state (including speed and acceleration) is obtained from the IMU and speedometer sensors. The model receives four RGB images from the front-camera sensor covering the past two seconds as visual input. AutoVLA predicts a five-second trajectory, which is then used by a PID controller to compute the control actions (throttle, brake, and steering) that are applied to the vehicle.

Closed-loop testing results in the CARLA simulator are shown in Fig. S10. Two representative scenarios are illustrated: (1) the ego vehicle equipped with AutoVLA successfully responds to a cut-in vehicle, and (2) it executes a smooth left turn. Additional closed-loop simulation results are available on the project website.

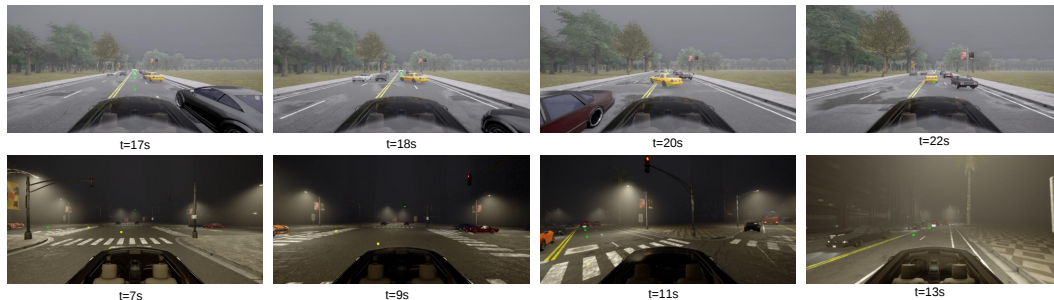


Figure S10: Closed-loop testing in the CARLA simulator. The two scenarios demonstrate AutoVLA's capability to (top) respond safely to a cut-in vehicle and (bottom) execute a smooth left turn.

G Broader Impacts

Autonomous driving is a safety-critical system, further emphasized by the integration of language guidance into the VLA model. This integration necessitates robust safeguards against adversarial attacks and proactive identification and filtering of unsafe human instructions. To mitigate potential hacking threats, it is essential to establish a secure communication channel in the vehicle, complemented by a gated-release strategy for model updates rather than online continual reinforcement fine-tuning on individual vehicles.