STEER: Route-Aware Adaptive Reasoning for Autonomous Driving

Yangang Zou[1,2,3] **Pei Liu**[4,5] **Nan Song**[1,3] **Bozhou Zhang**[1,3] **Mingyu Guo**[6,†]
Jun Ma[4,5] **Jiankang Deng**[7] **Xiatian Zhu**[8,*] **Li Zhang**[1,2,3,‡]

[1]School of Data Science, Fudan University [2]Shanghai Innovation Institute [3]Logos Intelligence Lab
[4]HKUST(GZ) [5]HKUST [6]Li Auto Inc. [7]Imperial College London [8]University of Surrey

[†]Project leader. [*]Joint last authorship. [‡]Corresponding author.

Abstract

Vision-language action (VLA) models excel for end-to-end autonomous driving, yet their tendency to over-invoke chain-of-thought (CoT) reasoning mirrors a human cognitive pitfall: *overthinking*. Just as deliberate reasoning can slow or mislead human judgment in routine tasks, excessive CoT in VLA incurs unnecessary compute overhead and can paradoxically degrade planning performance. Recent research tackles this via adaptive reasoning, which dynamically modulates inference complexity by selectively triggering CoT for complex scenes while responding directly to routine ones. While prior adaptive methods rely on implicit adaptation, recent work shows that explicitly optimizing the route between CoT and direct responses yields superior performance, assuming scene-difficulty annotations which are labor-intensive, poorly scalable, and prone to error due to the subjectivity of judging driving complexity. We explore, for the first time, whether adaptive reasoning with explicit route optimization can be achieved without any scene-difficulty labeling. We introduce **STEER**, a generic over-reasoning mitigation strategy featuring two key components: (i) *routing uncertainty-aware rollout*, which calibrates sampling to align routing diversity with model uncertainty; and (ii) *cross-route advantage credit assignment*, which introduces a differential metric to reinforce optimal routing decisions based on environmental rewards rather than subjective labels. Extensive experiments on NAVSIM v1/v2 demonstrate that STEER is competitive with state-of-the-art methods in both planning quality and the Pareto front of inference efficiency.

1 Introduction

End-to-end autonomous driving has been significantly advanced by vision-language action (VLA) models, which directly map multimodal observations to driving behaviors [7, 10, 8, 34]. With large-scale multimodal pretraining, these models introduce stronger semantic grounding and language-based scene understanding into trajectory generation, making it natural to incorporate explicit chain-of-thought (CoT) reasoning into the driving-planning process [8, 31, 33, 32]. Such reasoning is particularly valuable in scenarios that require multi-step inference, such as negotiating unprotected intersections, handling dense merges, or reacting to partially occluded hazards.

However, applying CoT reasoning uniformly to every driving scene is far from ideal, a pattern reminiscent of a human cognitive pitfall: *overthinking*. Just as an experienced driver does not consciously deliberate at every familiar junction, forcing explicit reasoning in routine situations can slow down responses, incur unnecessary compute overhead, and paradoxically degrade planning performance [18, 30]. These costs are especially concerning in safety-critical autonomous driving,

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

where both decision quality and response efficiency are essential. The key challenge, therefore, is not merely how to reason, but *when* explicit reasoning should be invoked.

Recent research has begun to address this challenge through adaptive reasoning, which dynamically modulates inference complexity. Existing approaches, however, differ substantially in how route selection is handled. AutoVLA [34] is the first work to integrate CoT into autonomous driving, but it does not explicitly optimize the route between reasoning and direct response. AdaThinkDrive [18] further shows that explicitly optimizing this route yields superior performance, suggesting that route selection itself is a critical ingredient for adaptive CoT. However, current explicit-routing approaches typically rely on scene-difficulty annotations to supervise route selection. Such supervision is labor-intensive, poorly scalable, and prone to error due to the subjectivity of judging driving complexity; more importantly, perceived scene difficulty is not always aligned with the actual utility of reasoning. A scene that appears visually crowded may still admit a direct response, while a seemingly simple scene may require nontrivial inference before a safe action can be made.

Motivated by this insight, we explore, for the first time, whether adaptive reasoning with explicit route optimization can be achieved entirely without scene-difficulty labeling. We propose **STEER**, a generic over-reasoning mitigation approach without scene-complexity labels, built within the VLA paradigm. STEER optimizes the routing policy through two key reinforcement learning (RL) components: (i) *routing uncertainty-aware rollout*, which calibrates sampling to align routing diversity with model uncertainty; and (ii) *cross-route advantage credit assignment*, which introduces a differential metric to reinforce optimal routing decisions based on environmental rewards rather than subjective labels. By contrasting the outcomes across different routes under the same prompt, STEER encourages the model to select the route that yields the highest reward. To prevent the model from prematurely collapsing into the shorter direct response, we introduce route-balanced supervised fine-tuning to preserve a balanced dual-route prior.

Our contributions are: **(i)** We identify *overthinking* as a key challenge in VLA-based autonomous driving and, for the first time, explore whether explicit routing preferences can be learned from comparative task outcomes, without any scene-difficulty annotations. **(ii)** We propose STEER, a generic over-reasoning mitigation strategy featuring *route-balanced supervised fine-tuning* to preserve dual-route capabilities, *routing uncertainty-aware rollout* to align sampling diversity with model uncertainty, and *cross-route advantage credit assignment* to reinforce optimal routing decisions directly from environmental rewards, eliminating the reliance on scene-difficulty labels. **(iii)** We achieve results competitive with state-of-the-art methods on NAVSIM v1/v2, consistently improving planning quality, routing quality, and pushing the Pareto front of inference efficiency over both fixed-routing and previous adaptive reasoning alternatives.

2 Related work

Vision-language action models for autonomous driving. End-to-end autonomous driving has evolved significantly from modular pipelines [7, 10] to unified architectures. Recently, Vision-language Models (VLMs) have been introduced to bridge the gap between semantic reasoning and physical actions [23, 1]. Current VLM-based driving approaches broadly fall into two paradigms. The first paradigm utilizes VLMs for scene understanding or high-level reasoning to produce meta-actions that guide downstream planners [27, 17, 12]. Although this facilitates semantic integration, it prevents full end-to-end optimization. The second paradigm, VLA models, directly integrates VLMs with action generation to predict final driving trajectories autoregressively [8, 34, 13]. Our work builds upon this end-to-end VLA paradigm. However, rather than solely focusing on *how* to generate trajectories, we address a critical internal decision process: optimizing *when* the model should invoke explicit reasoning before committing to an action.

Adaptive reasoning in autonomous driving. With the rise of long CoT reasoning in large language models, such as DeepSeek-R1 [6] and OpenAI o1 [9], explicit reasoning has become an effective tool for complex decision-making. In autonomous driving, methods such as EMMA [8] show that domain-specific CoT can improve trajectory prediction. However, reasoning is not universally beneficial: in routine driving scenes, forcing CoT can increase latency, introduce unnecessary verbosity, and even degrade planning quality [18, 30]. This has motivated adaptive reasoning, where the core challenge is to decide when explicit deliberation is needed. AutoVLA [34] addresses this through implicit adaptation: it mainly optimizes trajectory quality with sequence-level reinforcement learning, where

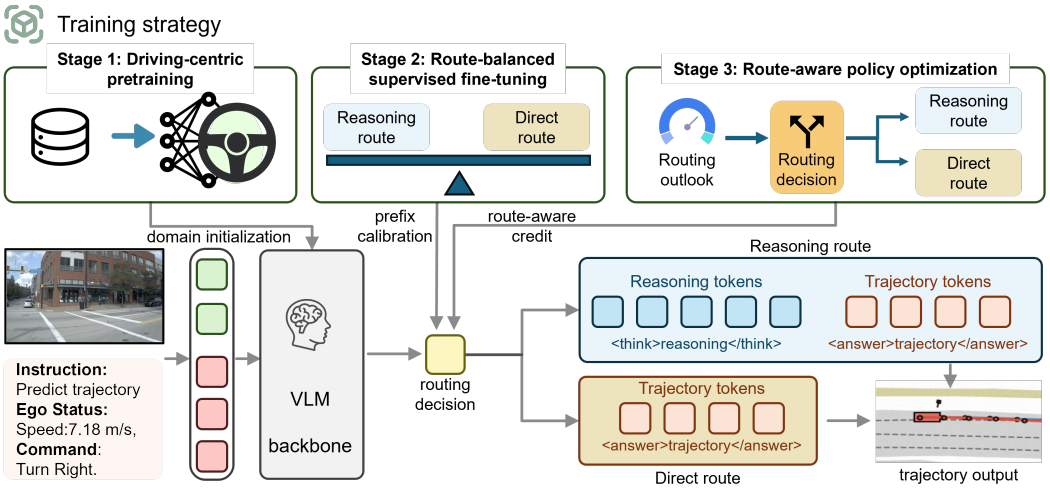

Figure 1: **Overview of STEER.** STEER is built on an autoregressive vision-language model that takes a front-view driving image together with structured driving context as input, and generates a textual future ego trajectory as output. It formulates adaptive reasoning in autonomous driving as an explicit routing problem, where an initial decision selects either a reasoning route or a direct route before trajectory generation. The framework is trained in three stages: (1) driving-centric pretraining for domain adaptation, (2) route-balanced supervised fine-tuning to preserve support for both valid routes, and (3) route-aware policy optimization to learn the routing preference directly from task rewards, without any scene-difficulty labels.

supervision is applied to the overall response rather than to route selection itself. AdaThinkDrive [18] moves one step further by explicitly supervising routing: it defines scene complexity for route selection and adds a dedicated routing reward to directly optimize the choice between CoT reasoning and direct response. It further shows that explicit route optimization outperforms implicit adaptation. The limitation, however, is that such supervision depends on scene-complexity annotation, which is labor-intensive, difficult to scale, and inherently subjective. Our work addresses this limitation by studying explicit adaptive reasoning without scene-complexity labels for autonomous driving.

3 Method

Model formulation. To address overthinking in VLA-based autonomous driving, we propose **STEER**, an adaptive reasoning framework that learns explicit routing between CoT reasoning and direct response from comparative task outcomes without scene-complexity labels. We instantiate STEER on top of an autoregressive vision-language model for end-to-end driving, e.g., Qwen2.5-VL-8B [1]. Given a multimodal driving prompt $\mathbf{x}$, the model generates a textual response $\mathbf{y}$ that specifies the future ego trajectory. To explicitly model whether the model should deliberate or respond immediately, we formulate the first step as a discrete routing decision $\mathcal{R} \in \{\texttt{reason}, \texttt{direct}\}$, corresponding to two valid response formats:

- *Reasoning*: `<think>` reasoning `</think>` `<answer>` trajectory `</answer>`
- *Direct responding*: `<answer>` trajectory `</answer>`

The response distribution is then factorized accordingly as

$$p_\theta(\mathbf{y} \mid \mathbf{x}) = p_\theta(\mathcal{R} \mid \mathbf{x}) \cdot p_\theta(\mathbf{y} \mid \mathbf{x}, \mathcal{R}). \tag{1}$$

The first term determines which route to take, serving as an explicit routing policy, while the second factor generates the driving trajectories conditioned on the selected route.

Model training. As illustrated in Figure 1, STEER follows a three-stage training pipeline: **(1)** driving-centric pretraining to adapt the base model to driving tasks (see Appendix C for details), **(2)** route-balanced supervised fine-tuning (SFT) to preserve the model's prior to execute both responding routes (Section 3.1), and **(3)** route-aware policy optimization to optimize the routing policy (Section 3.2).

3.1 Stage 2: Route-balanced supervised fine-tuning

Before RL can discover *when* explicit reasoning is beneficial, the model must first retain the capacity to execute *both* valid response routes. However, a vanilla SFT [20] objective is insufficient for this purpose. Because the direct route is substantially shorter than the reasoning route, standard teacher forcing introduces a strong optimization bias toward the shorter sequence. This length imbalance causes the model to prematurely collapse into the direct response, suppressing its reasoning ability before policy optimization even begins.

To prevent this early collapse, we introduce *route-balanced supervised fine-tuning*. For each prompt $\mathbf{x}$, we provide both a reasoning-route target $\mathbf{y}^{\texttt{reason}}$ and a direct-route target $\mathbf{y}^{\texttt{direct}}$, and optimize the model using a unified loss function:

$$\mathcal{L}_{\texttt{sft}}(\mathbf{x}) = \underbrace{-\frac{\lambda}{2}\sum_{\mathcal{R}} \log p_\theta(g_{\mathcal{R}} \mid \mathbf{x})}_{\text{Routing Calibration}} + \sum_{\mathcal{R}} \frac{\alpha_{\mathcal{R}}}{|\mathbf{y}^{\mathcal{R}}|} \sum_{t=1}^{|\mathbf{y}^{\mathcal{R}}|} -\log p_\theta(y_t^{\mathcal{R}} \mid \mathbf{x}, g_{\mathcal{R}}, \mathbf{y}_{<t}^{\mathcal{R}}). \tag{2}$$

Here, $g_{\mathcal{R}}$ denotes the route-indicating initial token corresponding to route $\mathcal{R}$, and $\alpha_{\mathcal{R}}$ balances the two routes ($\alpha_{\texttt{reason}} = \alpha$, $\alpha_{\texttt{direct}} = 1 - \alpha$). The sequence $\mathbf{y}^{\mathcal{R}}$ denotes the continuation after $g_{\mathcal{R}}$, which remains in the autoregressive prefix for every continuation token.

This objective addresses the pitfalls of vanilla SFT through two key designs: *(1) Routing calibration:* The first term explicitly regularizes the initial decision step toward a uniform distribution, ensuring the model maintains non-trivial support for both routes. *(2) Length normalization:* The second term divides the next-token loss by the route-specific length $|\mathbf{y}^{\mathcal{R}}|$. This prevents the optimization gradients from being skewed by the massive length disparity, ensuring the shorter direct route is not treated as an optimization shortcut. Ultimately, this balanced initialization serves a preparatory role. It preserves the dual-route capability, leaving the actual decision of *when* to reason entirely to the subsequent reward-driven RL stage.

3.2 Stage 3: Route-aware policy optimization

Starting from the balanced SFT model, we optimize the routing policy using sequence-level reinforcement learning (e.g., GRPO [25]) by introducing two key designs: *routing uncertainty-aware rollout* and *cross-route advantage credit assignment* (Figure 2).

Routing uncertainty-aware rollout. In vanilla GRPO, a group of responses is generated for each prompt via standard on-policy sampling. However, learning *when* to reason requires counterfactual comparison—observing outcomes from both the reasoning and direct routes under the same prompt. Standard on-policy sampling quickly collapses into the slightly preferred route, halting the exploration needed to discover true route utility.

Given prompt $\mathbf{x}$, the first generated token serves as the routing token. Specifically, if this token is the reasoning special token (i.e., "`<t>`"), the model takes the reasoning route; if it is the direct special token (i.e., "`<`"), the model takes the direct route. We compute the initial routing probabilities $p_{\texttt{reason}}$ and $p_{\texttt{direct}}$ by normalizing the model probabilities over these two valid route tokens, and quantify the routing uncertainty using binary entropy:

$$\mathcal{H}(\mathbf{x}) = -p_{\texttt{reason}} \log p_{\texttt{reason}} - p_{\texttt{direct}} \log p_{\texttt{direct}},$$

and define the minority-route probability as $p_{\min}(\mathbf{x}) = \min(p_{\texttt{reason}}, p_{\texttt{direct}})$. For a budget of n samples, we dynamically allocate the rollout regime:

$$(\text{Mode}, n_{\texttt{reason}}, n_{\texttt{direct}}, n_{\texttt{on}}) = \begin{cases} (\mathbf{A}, \lfloor n/2 \rfloor, \lfloor n/2 \rfloor, 0), & p_{\min} \geq \epsilon,\ \mathcal{H}(\mathbf{x}) > \tau_{\text{high}}, \\ (\mathbf{B}, \lfloor n/4 \rfloor, \lfloor n/4 \rfloor, \lfloor n/2 \rfloor), & p_{\min} \geq \epsilon,\ \tau_{\text{low}} \leq \mathcal{H}(\mathbf{x}) \leq \tau_{\text{high}}, \\ (\mathbf{C}, 0, 0, n), & \text{otherwise}, \end{cases} \tag{3}$$

where $n_{\texttt{reason}}$, $n_{\texttt{direct}}$, and $n_{\texttt{on}}$ denote the numbers of forced-reasoning, forced-direct, and standard on-policy rollouts.

Modes A and B force route-contrastive exploration (manually setting the initial token g and sampling autoregressively) when the routing decision is uncertain ($\mathcal{H}(\mathbf{x})$ is high). Mode C reverts to standard on-policy rollout when the policy is confident, avoiding wasteful exploration.

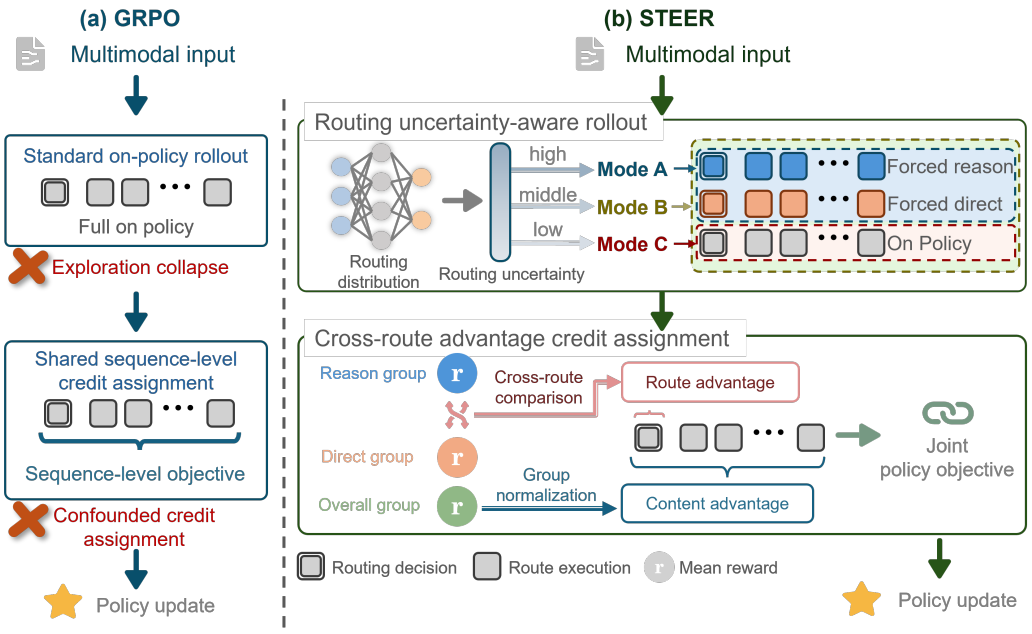

Figure 2: **Route-aware policy optimization in STEER.** For each prompt, STEER first inspects the initial routing distribution to estimate routing uncertainty. When the routing decision is uncertain, the model performs route-contrastive rollout by explicitly sampling both the reasoning and direct routes under the same prompt; otherwise, it falls back to standard on-policy rollout. During optimization, STEER applies a standard content advantage to all response tokens and adds a cross-route advantage at the initial routing position, jointly optimizing response quality and route preference.

Cross-route advantage credit assignment. Vanilla GRPO evaluates the generated group by computing a shared sequence-level advantage for all tokens in a response based on its relative reward. Without scene-difficulty labels, the model must infer route preference entirely from these task rewards. However, applying a shared advantage confounds the initial discrete routing choice with the much longer trajectory execution, making it impossible to distinguish whether a low reward stems from an unnecessary route or a poorly executed trajectory.

To resolve this confounding without relying on scene-difficulty labels, we derive the routing preference directly from comparative environmental rewards. By adding a dedicated cross-route learning signal to the initial decision, we explicitly optimize route preference alongside response quality.

For prompt $\mathbf{x}_k$, let $\mathcal{G}_k^{\texttt{reason}}$ and $\mathcal{G}_k^{\texttt{direct}}$ be the subsets of reasoning and direct rollouts, with route-wise mean rewards $\bar{r}_k^{\texttt{reason}}$ and $\bar{r}_k^{\texttt{direct}}$, and group mean $\bar{r}_k$. For the initial routing decision g, we define a *cross-route advantage* that explicitly compares the realized route against its counterfactual alternative:

$$A_i^{\texttt{route}} = \begin{cases} r_i - \bar{r}_k^{\texttt{direct}}, & i \in \mathcal{G}_k^{\texttt{reason}},\ |\mathcal{G}_k^{\texttt{direct}}| > 0, \\ r_i - \bar{r}_k^{\texttt{reason}}, & i \in \mathcal{G}_k^{\texttt{direct}},\ |\mathcal{G}_k^{\texttt{reason}}| > 0, \\ r_i - \bar{r}_k, & \text{otherwise.} \end{cases} \tag{4}$$

We analyze the expected routing update induced by Eq. 4 to characterize its effect on route preference. For a fixed prompt, let $\mu_{\mathrm{r}}, \mu_{\mathrm{d}}$ be the expected rewards of reasoning and direct responses, $\Delta = \mu_{\mathrm{r}} - \mu_{\mathrm{d}}$ their difference, and $p = \pi_\theta(\mathcal{R} = \texttt{reason} \mid \mathbf{x}) = \sigma(z)$ the reasoning probability. The sampled route proportions $q_{\mathrm{r}}, q_{\mathrm{d}}$ can differ from the model probabilities because rollout allocation can force either route.

With both routes represented, a simplified population analysis gives:

$$\mathbb{E}[\partial_z \log \pi_\theta(\mathcal{R} \mid \mathbf{x}) A^{\texttt{route}}] = [q_{\mathrm{r}}(1-p) + q_{\mathrm{d}} p]\,\Delta. \tag{5}$$

For $0 < p < 1$, the coefficient multiplying Δ is positive, so this additional update increases the probability of the route with higher expected reward. The derivation and assumptions are provided in Appendix G.2.

Optimization objective. We combine this route-specific signal with standard response optimization. All response tokens, including the routing token, receive the group-relative normalized advantage $A_i^{\texttt{content}} = (r_i - \bar{r}_k)/(\sigma_k + \delta)$, while $A_i^{\texttt{route}}$ provides an additional update at the routing position.

Let $\rho_i^{(t)} = p_\theta(y_{i,t} \mid \mathbf{x}_i, \mathbf{y}_{i,<t}) / p_{\theta_{\text{old}}}(y_{i,t} \mid \mathbf{x}_i, \mathbf{y}_{i,<t})$ be the token probability ratio, where $y_{i,0} = g_i$ and θ_{old} denotes the frozen pre-update parameters. We optimize all response tokens using a standard clipped surrogate objective:

$$\mathcal{L}^{\texttt{content}} = -\frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \sum_{t=0}^{T_i} \frac{m_i^{\texttt{resp}}(t)}{Z_i} \min\Big(\rho_i^{(t)} A_i^{\texttt{content}}, \text{clip}(\rho_i^{(t)}, 1-\epsilon_c, 1+\epsilon_c) A_i^{\texttt{content}}\Big), \quad (6)$$

where $m_i^{\texttt{resp}}(t)$ equals one for response tokens, including the initial routing token ($m_i^{\texttt{resp}}(0) = 1$), and zero otherwise; $Z_i = \sum_t m_i^{\texttt{resp}}(t)$ counts the response tokens.

For the routing decision, we additionally apply a clipped surrogate at the first response position using $A_i^{\texttt{route}}$:

$$\mathcal{L}^{\texttt{route}} = -\frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \min\Big(\rho_i^{(0)} A_i^{\texttt{route}}, \text{clip}(\rho_i^{(0)}, 1-\epsilon_c, 1+\epsilon_c) A_i^{\texttt{route}}\Big). \quad (7)$$

This objective is the core driver of our adaptation without scene-complexity labels. By updating the initial routing probability based on the comparative advantage, it directly optimizes the route selection policy $p_\theta(\mathcal{R} \mid \mathbf{x})$.

Combined with a KL regularization term $\mathcal{L}^{\text{KL}}$, the final joint objective is

$$\mathcal{L} = \mathcal{L}^{\texttt{content}} + \lambda_r \mathcal{L}^{\texttt{route}} + \beta_{\text{KL}} \mathcal{L}^{\text{KL}}. \quad (8)$$

4 Experiments

Datasets and evaluation metrics. We evaluate STEER on NAVSIM v1 and NAVSIM v2 [4] using the official simulator. Both route-balanced SFT and RL are trained on data constructed from the official navtrain split. For driving-centric pretraining, we additionally use public driving-reasoning datasets, including ReCogDrive [14], LingoQA [19], NuScenes-QA [22], DriveLM [26], and OmniDrive [28].

Following the official protocol, we use the Predictive Driver Model Score (PDMS) on NAVSIM v1 and Extended Predictive Driver Model Score (EPDMS) on NAVSIM v2 as the primary planning metrics. These composite scores evaluate trajectory quality by aggregating multiple sub-metrics, such as No Collision (NC), Drivable Area Compliance (DAC), and Ego Progress (EP). To evaluate routing quality without scene-complexity labels, we report RouteAcc, AUC, and AvgRegret. These metrics are computed against the *oracle route* (i.e., the route that yields the higher planning score for a given prompt), measuring route-selection correctness, route-ranking quality, and the performance gap to the oracle, respectively. Additionally, to analyze the route balance preserved during the SFT stage, we report the average routing probabilities and routing uncertainty. Full metric definitions for both driving and routing are provided in Appendix E.1.

Implementation details. STEER is initialized from Qwen2.5-VL-8B [1] and directly auto-regresses text waypoint trajectories without an additional planner. All training stages use full-parameter fine-tuning and are implemented with DeepSpeed ZeRO-3 [24] on 64 NVIDIA H20 GPUs. Detailed training hyperparameters are provided in Appendix E.2. For repeated evaluations, the main tables report means; standard deviations and evaluation details are provided in Appendix G.1.

4.1 Main results

Results on NAVSIM v1. As shown in Table 1, STEER achieves **91.2** PDMS on NAVSIM v1, competitive with state-of-the-art driving methods. Crucially, while prior explicit adaptive methods like AdaThinkDrive [18] rely on human-annotated scene-difficulty labels to supervise routing, STEER achieves superior planning performance (91.2 vs. 90.3) without scene-complexity labels. Furthermore, despite using a pure autoregressive VLM without an external planner, STEER numerically slightly surpasses the planner-assisted baseline, SGDrive (91.1). Under Best-of-N evaluation, STEER† further reaches **93.4** PDMS, demonstrating exceptional candidate quality.

Table 1: **Performance on NAVSIM v1** navtest using closed-loop metrics. * denotes fine-tuned on the NAVSIM trajectory dataset. † denotes Best-of-N evaluation with six generated candidates.

	Sensors	Trajectory	NC↑	DAC↑	EP↑	TTC↑	C↑	PDMS↑
Human GT								
Human [4]	-	-	100.0	100.0	87.5	100.0	99.9	94.8
End-to-End Methods								
Ego-MLP [16]	6x C	Continuous	93.1	78.3	63.2	84.0	**100.0**	66.4
UniAD [7]	6x C	Continuous	97.8	91.9	78.8	92.9	**100.0**	83.4
PARA-Drive [29]	6x C	Continuous	97.9	92.4	79.3	93.0	99.8	84.0
Hydra-MDP [15]	3x C + L	Continuous	98.3	96.0	78.7	94.6	**100.0**	86.5
DiffusionDrive [17]	3x C + L	Continuous	98.2	96.2	82.2	94.7	**100.0**	88.1
DiffusionDriveV2 [36]	3x C + L	Continuous	98.3	**97.9**	87.5	94.8	99.9	91.2
VLA (Planner-assisted VLM Methods)								
ImagiDrive-S [11]	1x C	Continuous	**98.6**	96.2	80.5	94.5	**100.0**	87.4
ReCogDrive [14]	3x C	Continuous	97.9	97.3	87.3	94.9	**100.0**	90.8
SGDrive [12]	1x C	Continuous	**98.6**	97.8	85.8	96.2	**100.0**	91.1
VLA (Autoregressive VLM Methods)								
Qwen2.5-VL* [1]	1x C	Text Waypoint	97.8	92.1	78.3	92.8	**100.0**	83.3
InternVL3* [35]	1x C	Text Waypoint	97.0	92.4	78.9	91.8	**100.0**	83.3
AutoVLA [34]	3x C	Discrete Action	98.4	95.6	81.9	**98.0**	99.9	89.1
AdaThinkDrive [18]	1x C	Text Waypoint	98.4	97.8	84.4	95.2	**100.0**	90.3
Curious-VLA [2]	1x C	Text Waypoint	98.4	96.9	88.5	97.9	98.1	90.3
STEER	1x C	Text Waypoint	97.4	97.3	90.2	92.3	**100.0**	91.2
STEER†	1x C	Text Waypoint	98.0	97.7	**91.6**	94.4	**100.0**	**93.4**

Table 2: **Performance on NAVSIM v2** navtest with extended metrics.

Method	NC ↑	DAC ↑	DDC ↑	TLC ↑	EP ↑	TTC ↑	LK ↑	HC ↑	EC ↑	EPDMS ↑
VADv2 [3]	97.3	91.7	98.2	99.9	77.6	92.7	66.0	**100**	97.4	76.6
TransFuser [21]	97.7	92.8	98.3	99.9	79.2	92.8	67.6	**100**	95.3	77.8
HydraMDP [15]	97.5	96.3	98.3	99.9	80.1	93.0	65.5	**100**	97.4	79.8
HydraMDP++ [15]	97.9	96.5	98.9	**100**	79.2	93.4	67.2	**100**	97.7	80.6
ARTEMIS [5]	98.3	95.1	98.6	99.8	81.5	97.4	96.5	**100**	**98.3**	83.1
ReCogDrive [14]	98.3	95.2	**99.5**	99.8	87.1	97.5	96.6	98.3	86.5	83.6
DiffusionDrive [17]	98.2	95.9	99.4	99.8	87.5	97.3	96.8	98.3	87.7	84.5
Curious-VLA [2]	98.4	96.9	99.2	99.8	88.5	**97.9**	**96.9**	98.1	81.5	85.3
SGDrive [12]	**98.6**	94.3	**99.5**	99.9	86.0	**97.9**	96.1	98.3	85.9	86.2
STEER	97.4	**97.6**	98.1	99.2	**95.6**	97.1	95.6	97.8	84.5	**87.2**

Table 3: **Adaptive routing versus fixed routing on NAVSIM v1.** Adaptive routing is compared against forced direct and forced reasoning policies after route-balanced SFT and after RL.

Training Stage	Route Policy	NC↑	DAC↑	EP↑	TTC↑	C↑	PDMS↑	Avg. Resp. Tokens↓
SFT	Forced direct	97.7	94.5	80.2	93.4	99.9	85.3	155
	Forced reasoning	97.7	94.4	80.1	93.7	99.9	86.4	417
	Adaptive	97.9	94.4	80.3	94.2	100.0	**86.5**	274
RL	Forced direct	97.6	96.5	89.0	92.1	99.9	89.9	156
	Forced reasoning	97.5	97.0	90.3	91.9	100.0	90.8	415
	Adaptive	97.4	97.3	90.2	92.3	100.0	**91.2**	315

Results on NAVSIM v2. As shown in Table 2, STEER achieves the best **87.2** EPDMS on NAVSIM v2. It improves over SGDrive (86.2) by **1.0** points and also obtains the best DAC (**97.6**) and EP (**95.6**), indicating strong decision quality under the extended evaluation protocol.

Overall, STEER consistently improves planning quality across both benchmarks. As shown in the following routing analyses, these gains are accompanied by improved route selection quality, indicating that the model benefits from learning *when* to invoke explicit reasoning rather than uniformly increasing reasoning length.

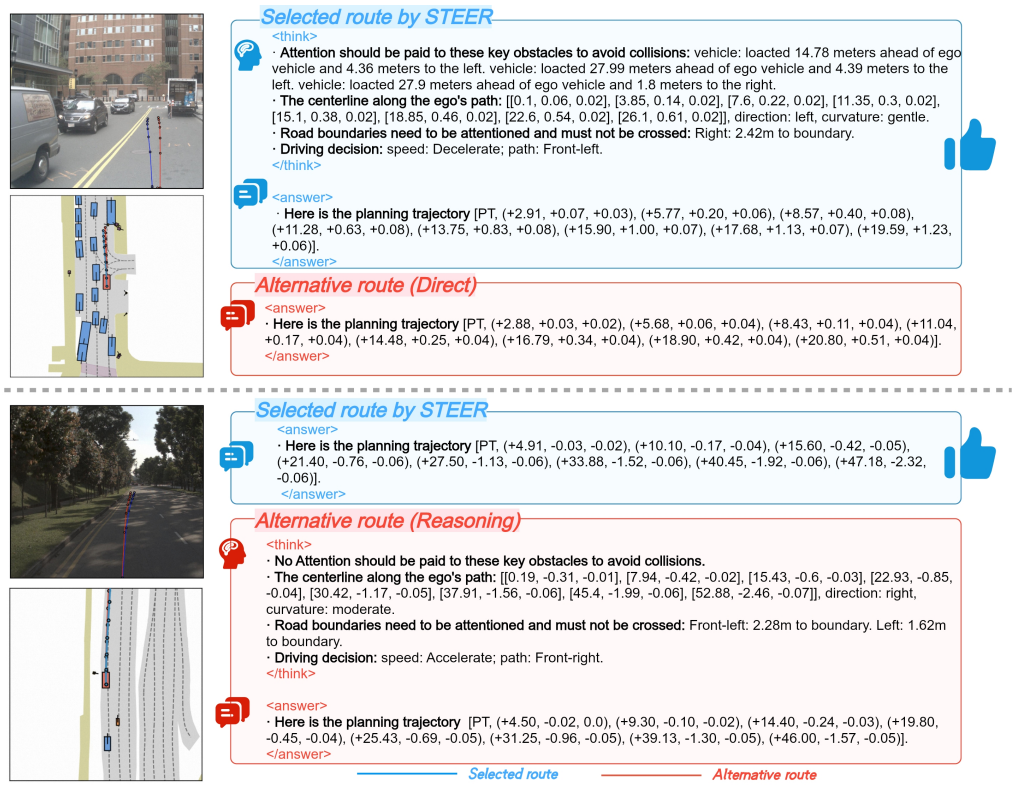

Figure 3: **Prompt-level route comparison.** For each prompt, we compare the trajectory produced by the route selected by STEER (blue) with the counterfactual alternative route (red). In the top case, explicit reasoning yields safer obstacle clearance. In the bottom case, STEER correctly avoids over-reasoning and selects the direct route, which better follows the road geometry compared to the perturbed reasoning alternative.

4.2 Adaptive routing analysis

Table 3 compares STEER's learned adaptive routing with forced-direct and forced-reasoning policies. The results directly validate our core motivation regarding the pitfall of overthinking. After route-balanced SFT, the adaptive policy yields only a marginal gain over the fixed policies, indicating that this stage primarily preserves dual-route support rather than learning a strong routing preference.

After RL, however, the forced-reasoning policy generates excessively long responses (415 tokens) but achieves a sub-optimal PDMS of 90.8. This confirms that blindly applying CoT to all scenes induces overthinking, which not only incurs unnecessary compute overhead but can also degrade planning quality. Importantly, the gain does not come from simply suppressing reasoning altogether: adaptive routing also outperforms the forced-direct policy (91.2 vs. 89.9), showing that STEER learns to preserve explicit reasoning precisely for scenes where it is beneficial. As a result, STEER achieves the highest PDMS of **91.2** while reducing the average response length to 315 tokens, demonstrating a superior effectiveness-efficiency trade-off.

To further understand the learned routing behavior, Figure 3 shows prompt-level comparisons between the route selected by STEER and the counterfactual alternative for the same input. The examples suggest that the model does not uniformly favor either response mode. Instead, it invokes explicit reasoning when doing so improves safety or constraint handling, and correctly avoids over-reasoning by choosing direct generation when it is sufficient to produce a cleaner trajectory. Additional qualitative comparisons are provided in Appendix I.

Table 4: **Ablation of route-aware policy optimization**. Our proposed mechanisms effectively overcome exploration collapse and confounded credit assignment, achieving the best routing quality and planning performance.

ID	Routing Uncert.-Aware Rollout	Cross-Route Advantage	RouteAcc (%)↑	AUC (%)↑	AvgRegret↓	Avg. Resp. Tokens↓	PDMS↑
1	×	×	69.3	73.9	0.034	363	89.5
2	✓	×	70.7	74.4	0.023	362	90.9
3	×	✓	77.5	76.5	0.024	350	89.9
4	✓	✓	**81.1**	**79.1**	**0.018**	**315**	**91.2**

Table 5: **Ablation of route-balanced SFT.** Length normalization and routing calibration alleviate the direct-route bias.

ID	Length Norm.	Route Calib.	Reasoning Prob. (%)	Direct Prob. (%)	PDMS↑
1	×	×	26.6	73.4	86.4
2	✓	×	43.2	56.8	86.1
3	✓	✓	**49.1**	**50.9**	**86.5**

Table 6: **Ablation of the training pipeline.** Planning performance with and without driving-centric pretraining and RL.

ID	Pretraining	RL	NC↑	DAC↑	EP↑	TTC↑	C↑	PDMS↑
1	×	×	97.8	94.1	79.9	94.0	100.0	85.9
2	✓	×	**97.9**	94.4	80.3	**94.2**	100.0	86.5
3	✓	✓	97.4	97.3	90.2	92.3	100.0	91.2
4	×	✓	97.5	97.3	90.2	92.5	100.0	91.2

Table 7: **Rollout-mode dynamics during route-aware policy optimization.** Mode columns give percentages of prompts.

Point	Mode A	Mode B	Mode C	Reasoning ratio
Start	85%	13%	2%	50%
Middle	44%	35%	21%	57%
End	17%	22%	61%	64%

4.3 Ablation studies

Effect of route-aware policy optimization. Table 4 isolates the contributions of our RL mechanisms. Vanilla GRPO (ID 1) improves over the route-balanced SFT initialization, but its routing regret remains relatively high, suggesting that sequence-level reward optimization alone does not reliably recover the latent route preference. Removing *cross-route advantage credit assignment* (ID 2) leads to credit confounding between route selection and route-conditioned execution, making the model less effective at penalizing unnecessary reasoning, as reflected by the inflated response length (362 tokens) and lower PDMS (90.9). Removing *routing uncertainty-aware rollout* (ID 3) deprives the model of counterfactual route comparison in uncertain scenes, causing exploration collapse and reducing PDMS to 89.9. Combining both components (ID 4) yields the best routing quality, the lowest regret, and the strongest planning performance, allowing STEER to achieve a competitive PDMS of 91.2 entirely without scene-difficulty labels.

Effect of the route-balanced SFT. In Table 5, length normalization raises the reasoning probability from 26.6% to 43.2%, alleviating the direct-route bias. Routing calibration further balances the prior (49.1%/50.9%). With comparable PDMS (86.1–86.5), this prior supports subsequent reward-driven route learning.

Effect of the training pipeline. Table 6 shows the contributions of the training stages. The SFT-only baseline (ID 1) achieves a PDMS of 85.9. Adding driving-centric pretraining (ID 2)

Table 8: **Efficiency analysis.** End-to-end inference latency (s/sample, left) and RL training cost (GPU-hours, right).

(a) Inference

Route policy	PDMS↑	Time (s/sample)↓
Forced direct	89.9	**2.4**
Forced reasoning	90.8	6.7
STEER	**91.2**	5.1

(b) RL training

Training configuration	GPU-hours↓
Vanilla GRPO	2,560
STEER w/o pretraining	2,387
STEER	**2,304**

provides a stronger driving-aware initialization, improving PDMS to 86.5. The largest gain then comes from route-aware policy optimization (ID 3), which further boosts PDMS to 91.18. Without pretraining, SFT followed by the same route-aware RL obtains 91.20 (ID 4). With comparable final PDMS, pretraining reduces RL-stage training cost from 2,387 to 2,304 GPU-hours, a 3.5% reduction (Table 8(b)).

Rollout-mode dynamics. The uncertainty-aware scheduler forces both routes for uncertain prompts (Modes A/B) to compare their task rewards, and uses more on-policy rollouts (Mode C) as routing confidence grows. Table 7 shows Mode A decreasing from 85% to 17% and Mode C increasing from 2% to 61% during training, while the budget stays at sixteen rollouts per prompt. At convergence, Modes A/B still cover 39% of prompts, averaging 4.48 forced rollouts per prompt; the reasoning/direct ratio remains 64%/36%. Thus uncertain prompts continue to receive cross-route comparisons, while confident prompts use more of the budget to optimize the current policy and both routes remain represented overall.

4.4 Efficiency analysis

For inference, we measure end-to-end latency from model input to complete trajectory output over the full NAVSIM v1 navtest set on a single H20 GPU with batch size one and identical input resolution and decoding settings. Latencies are averaged over ten evaluation runs. Table 8(a) uses the same checkpoint and route policies as Table 3. Relative to forced reasoning, STEER reduces latency from 6.7 to 5.1 s/sample (23.9%) while improving PDMS from 90.8 to 91.2.

Table 8(b) reports GPU-hours, calculated as $64\times$ the elapsed RL training time in hours, including rollout generation and forced-route allocation. STEER uses 10.0% fewer GPU-hours than vanilla GRPO; pretraining reduces STEER's RL-stage cost by 3.5%.

5 Conclusion

In this work, we identified *overthinking* as a critical cognitive pitfall in VLA-based autonomous driving. This led us to explore a central question: can adaptive reasoning with explicit route optimization be achieved entirely without subjective scene-difficulty labels? Our results support this feasibility claim. We introduced **STEER**, a generic over-reasoning mitigation strategy without scene-complexity labels that learns to balance CoT and direct responding directly from environmental rewards. By integrating *route-balanced supervised fine-tuning* to preserve a dual-route prior, alongside *routing uncertainty-aware rollout* and *cross-route advantage credit assignment*, STEER successfully reinforces scene-dependent routing decisions based on comparative task outcomes rather than manual difficulty annotations. Extensive experiments on NAVSIM v1 and v2 demonstrate that STEER achieves planning performance competitive with state-of-the-art methods while pushing the Pareto front of inference efficiency.

Acknowledgments

This work was supported in part by New Generation Artificial Intelligence- National Science and Technology Major Project (2025ZD0123004), National Natural Science Foundation of China (Grant No.62376060) and Ningbo grant (2025Z038).

References

[1] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025. URL `https://arxiv.org/abs/2502.13923`.

[2] Canyu Chen, Yuguang Yang, Zhewen Tan, Yizhi Wang, Ruiyi Zhan, Haiyan Liu, Xuanyao Mao, Jason Bao, Xinyue Tang, Linlin Yang, et al. Devil is in narrow policy: Unleashing exploration in driving vla models. *arXiv preprint arXiv:2603.06049*, 2026.

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

[4] Daniel Dauner, Marcel Hallgarten, Tianyu Li, Xinshuo Weng, Zhiyu Huang, Zetong Yang, Hongyang Li, Igor Gilitschenski, Boris Ivanovic, Marco Pavone, et al. Navsim: Data-driven non-reactive autonomous vehicle simulation and benchmarking. *Advances in Neural Information Processing Systems*, 37:28706–28719, 2024.

[5] Renju Feng, Ning Xi, Duanfeng Chu, Rukang Wang, Zejian Deng, Anzheng Wang, Liping Lu, Jinxiang Wang, and Yanjun Huang. Artemis: Autoregressive end-to-end trajectory planning with mixture of experts for autonomous driving. *IEEE Robotics and Automation Letters*, 11(1): 226–233, 2025.

[6] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] Jingyu Li, Bozhou Zhang, Xin Jin, Jiankang Deng, Xiatian Zhu, and Li Zhang. Imagidrive: A unified imagination-and-planning framework for autonomous driving. *arXiv preprint arXiv:2508.11428*, 2025.

[12] Jingyu Li, Junjie Wu, Dongnan Hu, Xiangkai Huang, Bin Sun, Zhihui Hao, Xianpeng Lang, Xiatian Zhu, and Li Zhang. Sgdrive: Scene-to-goal hierarchical world cognition for autonomous driving, 2026. URL `https://arxiv.org/abs/2601.05640`.

[13] Xidong Li, Mingyu Guo, Chenchao Xu, Bailin Li, Wenjing Zhu, Yangang Zou, Rui Chen, and Zehuan Wang. Mtdrive: Multi-turn interactive reinforcement learning for autonomous driving. *arXiv preprint arXiv:2601.22930*, 2026.

[14] Yongkang Li, Kaixin Xiong, Xiangyu Guo, Fang Li, Sixu Yan, Gangwei Xu, Lijun Zhou, Long Chen, Haiyang Sun, Bing Wang, et al. Recogdrive: A reinforced cognitive framework for end-to-end autonomous driving. *arXiv preprint arXiv:2506.08052*, 2025.

[15] 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.

[16] Zhiqi Li, Zhiding Yu, Shiyi Lan, Jiahan 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.

[17] Bencheng Liao, Shaoyu Chen, Haoran Yin, Bo Jiang, Cheng Wang, Sixu Yan, Xinbang Zhang, Xiangyu Li, Ying Zhang, Qian Zhang, et al. Diffusiondrive: Truncated diffusion model for end-to-end autonomous driving. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 12037–12047, 2025.

[18] Yuechen Luo, Fang Li, Shaoqing Xu, Zhiyi Lai, Lei Yang, Qimao Chen, Ziang Luo, Zixun Xie, Shengyin Jiang, Jiaxin Liu, et al. Adathinkdrive: Adaptive thinking via reinforcement learning for autonomous driving. *arXiv preprint arXiv:2509.13769*, 2025.

[19] Ana-Maria Marcu, Long Chen, Jan Hünermann, 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.

[20] 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.

[21] Aditya Prakash, Kashyap Chitta, and Andreas Geiger. Multi-modal fusion transformer for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7077–7087, 2021.

[22] 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.

[23] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[24] Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In *SC20: international conference for high performance computing, networking, storage and analysis*, pages 1–16. IEEE, 2020.

[25] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[26] 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.

[27] 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.

[28] Shihao Wang, Zhiding Yu, Xiaohui Jiang, Shiyi Lan, Min Shi, Nadine Chang, Jan Kautz, Ying Li, and Jose M Alvarez. Omnidrive: A holistic vision-language dataset for autonomous driving with counterfactual reasoning. In *Proceedings of the computer vision and pattern recognition conference*, pages 22442–22452, 2025.

[29] 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.

[30] Qi Yang, Bolin Ni, Shiming Xiang, Han Hu, Houwen Peng, and Jie Jiang. R-4b: Incentivizing general-purpose auto-thinking capability in mllms via bi-mode annealing and reinforce learning. *arXiv preprint arXiv:2508.21113*, 2025.

[31] Zhenlong Yuan, Chengxuan Qian, Jing Tang, Rui Chen, Zijian Song, Lei Sun, Xiangxiang Chu, Yujun Cai, Dapeng Zhang, and Shuo Li. Autodrive-r^2: Incentivizing reasoning and self-reflection capacity for vla model in autonomous driving. *arXiv preprint arXiv:2509.01944*, 2025.

[32] Shuang Zeng, Xinyuan Chang, Mengwei Xie, Xinran Liu, Yifan Bai, Zheng Pan, Mu Xu, Xing Wei, and Ning Guo. Futuresightdrive: Thinking visually with spatio-temporal cot for autonomous driving. *arXiv preprint arXiv:2505.17685*, 2025.

[33] Lingjun Zhang, Yujian Yuan, Changjie Wu, Xinyuan Chang, Xin Cai, Shuang Zeng, Linzhe Shi, Sijin Wang, Hang Zhang, and Mu Xu. Minddriver: Introducing progressive multimodal reasoning for autonomous driving, 2026. URL `https://arxiv.org/abs/2602.21952`.

[34] Zewei Zhou, Tianhui Cai, Seth Z Zhao, Yun Zhang, Zhiyu Huang, Bolei Zhou, and Jiaqi Ma. Autovla: A vision-language-action model for end-to-end autonomous driving with adaptive reasoning and reinforcement fine-tuning. *arXiv preprint arXiv:2506.13757*, 2025.

[35] Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, et al. Internvl3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*, 2025.

[36] Jialv Zou, Shaoyu Chen, Bencheng Liao, Zhiyu Zheng, Yuehao Song, Lefei Zhang, Qian Zhang, Wenyu Liu, and Xinggang Wang. Diffusiondrivev2: Reinforcement learning-constrained truncated diffusion modeling in end-to-end autonomous driving. *arXiv preprint arXiv:2512.07745*, 2025.

A Appendix overview

This appendix provides additional details that complement the main paper. We describe the data preparation pipeline, reward design, and full implementation details, and then report metric definitions, hyper-parameter settings, the route-surrogate analysis and controlled experiments, sensitivity analyses, and additional qualitative visualizations.

B Data preparation

This appendix summarizes the data preparation pipeline used in STEER. We construct two types of data: (1) **driving-centric pretraining data**, which adapts a general-purpose VLM to driving semantics and planning-relevant reasoning patterns, and (2) **dual-format SFT data**, which provides both reasoning-route and direct-route supervision under the same prompt distribution.

B.1 Driving-centric pretraining data

The pretraining stage bridges general vision–language capability and autonomous-driving decision making before end-to-end trajectory generation. To this end, we construct three categories of driving-centric supervision.

Scene structure and key-obstacle understanding. We build instruction–answer pairs for lane topology recognition, drivable-area understanding, road-boundary identification, ego-lane localization, and key-obstacle identification under the current driving context. These tasks encourage the model to form structured scene representations and focus on decision-relevant agents.

Trajectory feasibility and safety judgment. Inspired by the trajectory-judgment formulation in MTDrive [13], we construct samples in which the model is given candidate future trajectories and asked to judge whether each trajectory is safe and feasible. This trains the model to connect scene geometry and agent interactions with future safety outcomes.

Public driving VLM datasets. We further incorporate public driving QA and reasoning datasets, including ReCogDrive [14], LingoQA [19], NuScenes-QA [22], DriveLM [26], and OmniDrive [28], to enrich semantic coverage and improve generalization.

B.2 Dual-format SFT data

To support explicit adaptive routing, each SFT sample is paired with two valid targets under the same prompt: a **direct route** target and a **reasoning route** target. Each sample contains a front-view image, structured driving context, and the target future ego trajectory.

For the reasoning route, we use a structured CoT format aligned with planning-relevant driving cues. Each reasoning trace contains four components: (1) identifying the key obstacle under the current scene context, (2) describing the road centerline or nominal drivable path, (3) analyzing road layout and boundary constraints, and (4) summarizing a high-level driving decision, including lateral intent and longitudinal speed trend. This structure encourages reasoning traces that remain closely aligned with downstream planning.

B.3 Automatic annotation pipeline

We construct the above supervision with an automatic annotation pipeline that combines rule-based grounding and VLM-based distillation. Objective scene elements such as lane geometry, road boundaries, ego-lane position, dynamic object states, trajectory targets, and navigation signals are first extracted from structured annotations and simulator metadata. These grounded signals are then organized into a prompt template together with the front-view image and fed to a strong teacher VLM (Qwen2.5-VL-72B [1]) to synthesize the reasoning trace. Figure 4 illustrates the prompt template used for reasoning-data generation.

The prompt template is designed to expose the teacher model to the core factors relevant to planning. On the input side, it provides static-road information (e.g., road boundaries and drivable-area

geometry), dynamic-object attributes (e.g., position, heading, speed, and object class), and ego-state information (e.g., ego pose, motion history, navigation instruction, and front-view image). On the output side, the teacher model is asked to generate planning-oriented reasoning rather than free-form scene descriptions. In particular, the generated CoT is required to cover the same four components used in our reasoning-route supervision: key-obstacle identification, road centerline or nominal path, road layout and boundary analysis, and high-level driving decision.

Within this pipeline, rule-based processing is used to provide reliable scene grounding, while the teacher VLM is responsible for converting these grounded signals into coherent natural-language reasoning. For example, road-boundary distances and drivable-area geometry can be computed directly from structured map annotations, while key-obstacle selection and high-level decision description are distilled from the teacher model under grounded prompts. This hybrid design combines the factual precision of structured annotations with the flexibility of language generation.

These data components provide driving-aware initialization and dual-route execution supervision, but they do not provide route-preference labels or scene-difficulty annotations. The final routing policy is learned later from environmental rewards during route-aware policy optimization. The 72B teacher constructs driving data and CoT execution targets offline, without assessing scene complexity or providing routing labels.

B.4 Training data template

Figure 5 shows an example of the training data template used in our experiments, including the task description, sensory inputs, and target trajectory format.

C Driving-centric pretraining details

The driving-centric pretraining stage is used to adapt the base vision-language model to autonomous-driving tasks before route-balanced SFT and route-aware policy optimization. We use a driving-specific pretraining corpus constructed from multiple data sources and automatic annotation pipelines. Detailed data composition and construction are provided in Appendix B.1.

D Reward design

This appendix details the scalar reward used in the RL stage of STEER. The reward is designed to supervise response validity, trajectory quality, and reasoning-action consistency. Importantly, the reward itself does not impose any scene-complexity or route-choice label. The actual learning signal for route selection is provided by the cross-route advantage defined in Section 3.2, which compares the realized outcomes of the reasoning and direct routes under the same prompt.

Format reward. A binary reward $r^{\texttt{fmt}} \in \{0, 1\}$ checks whether the output follows one of the two valid response formats, namely the direct route or the reasoning route. For the reasoning route, we additionally require that the `<think>` block does not contain nested `<answer>` tags. If the format check fails, the rollout receives zero effective task reward.

Planning reward. We use PDMS [4] as the main trajectory-quality signal:

$$r^{\texttt{pdms}} = \mathrm{PDMS}(\hat{\boldsymbol{\tau}}), \tag{9}$$

where $\hat{\boldsymbol{\tau}}$ is the predicted trajectory parsed from the `<answer>` block. PDMS jointly evaluates safety, drivable-area compliance, and route progress, and serves as the dominant task objective during RL.

Reasoning-action consistency reward. For the reasoning route, we introduce a binary consistency reward $r^{\texttt{cons}} \in \{0, 1\}$ that measures whether the high-level decision expressed in `<think>` is consistent with the generated trajectory. Concretely, we compare two aspects: speed trend and path intent. The reward is defined as

$$r^{\texttt{cons}} = \mathbb{1}\left[s_{\texttt{reason}} = s_{\texttt{traj}} \ \wedge \ p_{\texttt{reason}} = p_{\texttt{traj}}\right], \tag{10}$$

where s and p denote the speed and path categories, respectively. This term acts as a lightweight regularizer that encourages the generated reasoning to remain aligned with the final planning output, rather than drifting into irrelevant or contradictory deliberation.

Data annotation template

System prompt:

You are an autonomous-driving reasoning module.

Task:Generate a concise reasoning summary strictly from the provided inputs. Do not invent facts. Use only the given information.

Output only: <think>...</think>. Inside <think>, include exactly four parts in this order:

1. **Key obstacles**
- Select up to 3 obstacles with the highest collision risk.
- Prefer obstacles that are closer to the ego vehicle, closer to the planned path, in front of the ego vehicle, and approaching if velocity is available.
- Ignore obstacles clearly irrelevant to forward driving.
  If no obstacle requires attention, output exactly: No attention should be paid to key obstacles.
  Otherwise output: Attention should be paid to these key obstacles to avoid collisions: followed by up to 3 lines in the format:{object type}: located {X} meters ahead of ego vehicle and {Y} meters to the left/right.

2. **Ego path summary**
- Output exactly: The centerline along the ego's path: {centerline}, direction: {direction}, curvature: {curvature}.
- Keep direction consistent with the path decision.

3. **Road boundaries**
- Check Left, Right, Front, Front-left, and Front-right.
- If any distance is below 3.0 m, output: Road boundaries need attention and must not be crossed: followed by one line per direction: {Direction}: {distance}m to boundary.

4. **Driving decision**
- Output exactly: Driving decision: speed: {Stop/Accelerate/Decelerate}; path: {Turn left/Turn right/Lane Keep/Front-left/Front-right}.

Use concise, formal, and grammatically correct English.

User prompt:

Generate the reasoning summary for the following driving scene.

Inputs:

1. **Front-view image**
   <image>

2. **Detected obstacles**
- Each obstacle is given as: id, type, bbox [x1, y1, x2, y2], x_forward (m), y_left (m), optional vx_forward, optional vy_left
- Obstacles:
  {id: {id_1}, type: {type_1}, bbox: [{x1_1}, {y1_1}, {x2_1}, {y2_1}], x_forward: {xf_1}, y_left: {yl_1}, vx_forward: {vxf_1}, vy_left: {vyl_1}}
  {id: {id_2}, type: {type_2}, bbox: [{x1_2}, {y1_2}, {x2_2}, {y2_2}], x_forward: {xf_2}, y_left: {yl_2}, vx_forward: {vxf_2}, vy_left: {vyl_2}}
  ...

3. **Ego path information**
- Centerline: {centerline}; Direction hint: {direction_hint}; Curvature hint: {curvature_hint}

4. **Road-boundary distances (m)**
- Left: {dist_left}; Right: {dist_right}; Front: {dist_front}; Front-left: {dist_front_left}; Front-right: {dist_front_right}

5. **Ground-truth driving decision**
- Path decision: {Turn left / Turn right / Lane Keep / Front-left / Front-right}
- Speed decision: {Stop / Accelerate / Decelerate}

Figure 4: Prompt template for reasoning-data generation.

Training data template

System prompt:

You are a vehicle trajectory prediction model for autonomous driving.

Task: Predict the ego vehicle's 4-second trajectory based on the following inputs: an image from the front-view camera, ego vehicle states (position), and discrete navigation commands. The input provides a 2-second history, and your output should ensure a safe trajectory for the next 4 seconds.

Your predictions must adhere to the **following metrics**:
1. **No at-fault Collisions (NC)**: Avoid collisions with other objects/vehicles.
2. **Drivable Area Compliance (DAC)**: Stay within the drivable area.
3. **Time to Collision (TTC)**: Maintain a safe distance from other vehicles.
4. **Ego Progress (EP)**: Ensure the ego vehicle moves forward without being stuck.
5. **Comfort (C)**: Avoid sharp turns and sudden decelerations.
6. **Driving Direction Compliance (DDC)**: Align with the intended driving direction.
For evaluation, use the **PDM Score**, which combines these metrics:
PDM Score = NC * DAC * (5*TTC + 5*EP + 2*C + 0*DDC) / 12

Your predictions will be evaluated through a non-reactive 4-second simulation with an LQR controller and background actors following their recorded trajectories.

The better your predictions, the higher your score.

User prompt:

<image>
As an autonomous driving system, predict the vehicle's trajectory based on:
1. Visual perception from front camera view.
2. **Historical motion context** (last 4 timesteps): - t-3: (-0.05, 0.0, 0.0) - t-2: (-0.05, 0.0, 0.0) - t-1: (-0.04, 0.0, 0.0) - t-0: (0.0, 0.0, 0.0)
3. Active navigation command: [**GO STRAIGHT**]

Output requirements:
- Predict 8 future trajectory points
- Each point format: (x:float, y:float, heading:float)
- Use [PT, ...] to encapsulate the trajectory\n- Maintain numerical precision to 2 decimal places.

Front-view image:



Answer of reasoning route:

<think>Attention should be paid to these key obstacles to avoid collisions:
vehicle: loacted 13.77 meters ahead of ego vehicle and 0.14 meters to the left.
vehicle: loacted 14.19 meters ahead of ego vehicle and 3.39 meters to the left.
vehicle: loacted 19.38 meters ahead of ego vehicle and 6.76 meters to the left.
The centerline along the ego's path: [[0.18, 0.32, 0.0], [3.18, 0.31, -0.0], [5.93, 0.31, -0.0], [8.61, 0.29, 0.0], [11.36, 0.28, 0.0], [14.11, 0.3, 0.01], [16.86, 0.34, 0.02], [19.61, 0.37, 0.01]], direction: straight, curvature: gentle.
Road boundaries need to be attentioned and must not be crossed: Front-right: 1.9m to boundary Right: 1.34m to boundary.
Driving decision: speed: Accelerate; path: Lane keep. </think>
<answer>Here is the planning trajectory [PT, (+0.21, 0.0, 0.0), (+0.64, 0.0, 0.0), (+1.34, 0.0, 0.0), (+2.32, 0.0, 0.0), (+3.63, +0.02, 0.0), (+5.25, +0.05, 0.0), (+7.19, +0.06, 0.0), (+9.48, +0.08, 0.0)].</answer>

Answer of direct-response route:

<answer>Here is the planning trajectory [PT, (+0.21, 0.0, 0.0), (+0.64, 0.0, 0.0), (+1.34, 0.0, 0.0), (+2.32, 0.0, 0.0), (+3.63, +0.02, 0.0), (+5.25, +0.05, 0.0), (+7.19, +0.06, 0.0), (+9.48, +0.08, 0.0)].</answer>

Figure 5: Training data template used for driving-centric pretraining and SFT.

Final scalar reward. We use a route-dependent scalar reward:

$$r = \begin{cases} \alpha_f\, r^{\texttt{fmt}} + \alpha_p\, r^{\texttt{pdms}} + \alpha_c\, r^{\texttt{cons}}, & \text{if the reasoning route is used,} \\ \alpha_f\, r^{\texttt{fmt}} + (1-\alpha_f)\, r^{\texttt{pdms}}, & \text{if the direct route is used.} \end{cases} \tag{11}$$

Here, $r^{\texttt{pdms}}$ remains the dominant signal in both routes. The consistency term is only applied to the reasoning route, where explicit deliberation is present, and serves as a mild auxiliary regularizer rather than a primary optimization target. With the weights in Table 9, the direct reward is $0.9r^{\texttt{pdms}} + 0.1r^{\texttt{fmt}}$, whereas the reasoning reward is $0.8r^{\texttt{pdms}} + 0.1r^{\texttt{fmt}} + 0.1r^{\texttt{cons}}$. Both weight sums equal one, normalizing the maximum reward scale across routes. The consistency term contributes to the scalar return used for route learning.

Overall, this reward formulation provides task-level supervision for valid response generation and driving performance, with comparative policy optimization learning which route to prefer from these total returns, rather than from routing annotations.

E Experimental details

E.1 Detailed metric definitions

In the main paper, we report driving planning metrics from the NAVSIM benchmark [4] as well as custom routing metrics (**RouteAcc**, **AUC**, and **AvgRegret**) to evaluate routing quality. Here we provide their full definitions.

Driving planning metrics. Following the NAVSIM benchmark, we evaluate trajectory quality using the Predictive Driver Model Score (PDMS) for v1 and Extended Predictive Driver Model Score (EPDMS) for v2.

NAVSIM v1 Metrics: PDMS is a composite metric that heavily weights the driving progress while strictly penalizing safety and comfort violations. It is computed based on several sub-metrics:

- **NC (No Collision):** A severity-aware at-fault collision score taking values in $\{0, 0.5, 1\}$; its mean is not a binary collision-event rate.
- **DAC (Drivable Area Compliance):** Evaluates whether the trajectory stays within the designated drivable area. This is a binary per-scenario compliance score.
- **EP (Ego Progress):** Measures the ego vehicle's normalized longitudinal progress along the reference route.
- **TTC (Time to Collision):** A binary per-scenario score checking prospective collision risk; reported TTC scores are compliance percentages, rather than times in seconds.
- **C (Comfort):** Penalizes excessive acceleration, deceleration, or jerk to ensure a smooth ride.

NAVSIM v2 Metrics: EPDMS extends PDMS by incorporating additional fine-grained compliance checks:

- **DDC (Driving Direction Compliance):** Penalizes driving against the permitted traffic direction.
- **TLC (Traffic Light Compliance):** Checks adherence to traffic light signals.
- **LK (Lane Keeping):** Evaluates lateral stability and alignment with the lane.
- **HC (History Comfort):** Evaluates comfort using the historical and predicted ego motion.
- **EC (Extended Comfort):** Evaluates the consistency of predicted accelerations and jerks across consecutive frames.

Routing quality metrics. For each prompt $\mathbf{x}$, we force the model to follow the reasoning route and the direct-response route, obtaining PDMS scores $r^{\texttt{reason}}(\mathbf{x})$ and $r^{\texttt{direct}}(\mathbf{x})$, respectively. Let $\hat{a}(\mathbf{x}) \in \{\texttt{reason}, \texttt{direct}\}$ denote the route selected under free decoding.

RouteAcc measures whether the selected route matches the higher-reward route:

$$\text{RouteAcc} = \frac{1}{N} \sum_{\mathbf{x}} \mathbb{1}\left[\hat{a}(\mathbf{x}) = \arg \max_{a \in \{\texttt{reason},\texttt{direct}\}} r^{a}(\mathbf{x})\right]. \tag{12}$$

AUC measures how well the predicted reasoning-route probability ranks prompts according to whether the reasoning route is preferable. For each prompt, we define a binary label

$$y(\mathbf{x}) = \mathbb{1}\left[r^{\texttt{reason}}(\mathbf{x}) > r^{\texttt{direct}}(\mathbf{x})\right], \tag{13}$$

and use the predicted reasoning-route probability $P_{\texttt{reason}}(\mathbf{x})$ as the ranking score. Higher AUC indicates that the model assigns larger reasoning probabilities to prompts for which the reasoning route is indeed more beneficial.

AvgRegret measures the gap between the selected route and the oracle route:

$$\text{AvgRegret} = \frac{1}{N}\sum_{\mathbf{x}}\left(\max\{r^{\texttt{reason}}(\mathbf{x}),\, r^{\texttt{direct}}(\mathbf{x})\} - r^{\hat{a}(\mathbf{x})}(\mathbf{x})\right). \tag{14}$$

For the route-balanced SFT stage analysis, we additionally report routing-distribution statistics based on the initial decision probabilities. Let $P_{\texttt{reason}}(\mathbf{x})$ and $P_{\texttt{direct}}(\mathbf{x})$ denote the predicted probabilities of taking the reasoning and direct-response routes, respectively. We report their dataset averages,

$$\bar{P}_{\texttt{reason}} = \frac{1}{N}\sum_{\mathbf{x}} P_{\texttt{reason}}(\mathbf{x}), \qquad \bar{P}_{\texttt{direct}} = \frac{1}{N}\sum_{\mathbf{x}} P_{\texttt{direct}}(\mathbf{x}), \tag{15}$$

as well as the average routing uncertainty (entropy):

$$\text{RoutingUncertainty} = -\frac{1}{N}\sum_{\mathbf{x}}\sum_{a\in\{\texttt{reason},\texttt{direct}\}} P_a(\mathbf{x})\log P_a(\mathbf{x}). \tag{16}$$

E.2 Detailed implementation

Model. STEER is initialized from Qwen2.5-VL-8B and directly auto-regresses text waypoint trajectories without an additional planner. Given the current front-view image and the ego vehicle's historical motion, the model predicts a 4-second future trajectory as 8 waypoint tokens in natural language format. The historical trajectory is represented as numerical coordinates and concatenated into the prompt.

Training pipeline. Training consists of three stages: driving-centric pretraining, route-balanced SFT, and route-aware policy optimization. In pretraining, we use about 600K samples for 2 epochs with a global batch size of 256 and a learning rate of 4.0×10^{-5}. In SFT, we train on 80K NAVSIM samples for 5 epochs with a global batch size of 256 and a learning rate of 4.0×10^{-5}. In RL, we use 40K NAVSIM samples for 5 epochs with a global batch size of 256, a learning rate of 3.0×10^{-6}, and 16 rollouts per prompt.

Training setup. All stages use full-parameter fine-tuning and are implemented with DeepSpeed ZeRO-3 on 64 NVIDIA H20 GPUs. Additional system and optimization details follow the standard distributed training setup used in our codebase.

F Hyper-parameter settings

We summarize the main hyper-parameter settings used in SFT and RL in Table 9. Unless otherwise specified, all experiments use the same backbone, input resolution, and distributed training setup.

G Supplementary analysis and controlled experiments

G.1 Evaluation statistics

Tables 10–13 report mean $\pm$ standard deviation over three evaluation seeds for each listed fixed checkpoint. The vanilla-GRPO and STEER conditions in Table 12 share the same backbone, initialization, data, reward, split, sixteen-rollout budget, and optimization settings; they differ in uncertainty-aware rollout and cross-route advantage credit assignment.

G.2 Expected-update analysis of the route surrogate

For a fixed prompt, let $\mu_{\mathrm{r}} = \mathbb{E}[r \mid \mathcal{R} = \texttt{reason}]$, $\mu_{\mathrm{d}} = \mathbb{E}[r \mid \mathcal{R} = \texttt{direct}]$, and $\Delta = \mu_{\mathrm{r}} - \mu_{\mathrm{d}}$, where r is the total scalar reward. We analyze the additional route-specific update induced by $\mathcal{L}^{\texttt{route}}$ using the population form of Eq. 4, with both routes represented, holding route-conditioned returns fixed and ignoring clipping and KL regularization. The score update is evaluated at $\theta = \theta_{\mathrm{old}}$, where

Table 9: **Main hyper-parameter settings for route-balanced SFT and route-aware policy optimization.**

SFT		**Route-aware policy optimization**	
Parameter	Value	Parameter	Value
Base model	Qwen2.5-VL-8B	Rollout group size n	16
Max pixels	2073600	Clipping coefficient ϵ_c	0.2
Global batch size	256	Minority-route threshold ϵ	0.05
Epochs	5	Low-entropy threshold τ_{low}	0.25
Learning rate	4.0×10^{-5}	High-entropy threshold τ_{high}	0.50
Weight decay	0.05	Global batch size	256
Warmup ratio	0.1	Epochs	5
bfloat16	Enabled	Learning rate	3.0×10^{-6}
Routing calibration weight λ	0.1	Weight decay	0.01
Route-balance coefficient α	0.5	KL coefficient β_{KL}	0.1
GPUs	64	Max response length	1000
		Route loss weight λ_r	0.05
		Format reward weight α_f	0.1
		PDMS reward weight α_p (reasoning route)	0.8
		Consistency reward weight α_c	0.1
		PDMS reward weight $(1 - \alpha_f)$ (direct route)	0.9
		GPUs	64

Table 10: **NAVSIM v1 evaluation statistics.** Mean $\pm$ standard deviation over three evaluation seeds.

Setting	NC↑	DAC↑	EP↑	TTC↑	C↑	PDMS↑
STEER	97.41 ± 0.36	97.25 ± 0.55	90.22 ± 0.81	92.25 ± 1.10	100.0 ± 0.0	91.18 ± 0.25
STEER (Best-of-6)	98.02 ± 0.19	97.73 ± 0.31	91.61 ± 0.32	94.36 ± 0.51	100.0 ± 0.0	93.44 ± 0.12

Table 11: **NAVSIM v2 evaluation statistics for STEER.** Mean $\pm$ standard deviation over three evaluation seeds.

Metric	Score	Metric	Score
NC↑	97.40 ± 1.21	TTC↑	97.12 ± 0.92
DAC↑	97.63 ± 0.58	LK↑	95.61 ± 1.31
DDC↑	98.05 ± 1.10	HC↑	97.81 ± 0.08
TLC↑	99.15 ± 0.15	EC↑	84.47 ± 1.12
EP↑	95.62 ± 0.22	EPDMS↑	87.15 ± 0.33

Table 12: **Matched comparison with vanilla GRPO.** Mean $\pm$ standard deviation over three evaluation seeds for IDs 1 and 4 of Table 4.

Setting	RouteAcc (%)↑	AUC (%)↑	AvgRegret↓	Avg. tokens↓	PDMS↑
Vanilla GRPO	69.33 ± 0.68	73.91 ± 0.31	0.034 ± 0.005	363 ± 6	89.45 ± 0.31
STEER	81.09 ± 0.34	79.06 ± 0.40	0.018 ± 0.002	315 ± 4	91.18 ± 0.25

Table 13: **Pretraining ablation after route-aware RL.** Mean $\pm$ standard deviation over three evaluation seeds; both settings use route-balanced SFT and the same RL procedure.

Setting	NC↑	DAC↑	EP↑	TTC↑	C↑	PDMS↑
With pretraining	97.41 ± 0.36	97.25 ± 0.55	90.22 ± 0.81	92.25 ± 1.10	100.0 ± 0.0	91.18 ± 0.25
Without pretraining	97.45 ± 0.40	97.31 ± 0.34	90.17 ± 0.63	92.54 ± 0.90	100.0 ± 0.0	91.20 ± 0.28

the token probability ratio equals one. For forced routes, this ratio still uses the old model probability; route allocation determines the sampling proportions below.

Table 14: **Sensitivity analysis of the route-balanced SFT calibration weight λ.** Moderate route calibration improves route balance and yields the best downstream planning performance.

λ	Reasoning Prob. (%)	Direct Prob. (%)	Routing Uncertainty↑	PDMS↑
0	43.2	56.8	0.681	86.1
0.05	48.5	51.5	0.689	86.0
0.10	49.1	50.9	**0.693**	**86.5**
0.15	49.4	50.6	0.692	86.0
0.20	49.6	50.4	0.692	85.9

Writing $p = \pi_\theta(\mathcal{R} = \texttt{reason} \mid \mathbf{x}) = \sigma(z)$ and denoting the sampled route proportions by q_r and q_d, with $q_\mathrm{r} + q_\mathrm{d} = 1$, the population advantages satisfy

$$\mathbb{E}[A^{\texttt{route}} \mid \mathcal{R} = \texttt{reason}] = \Delta, \qquad \mathbb{E}[A^{\texttt{route}} \mid \mathcal{R} = \texttt{direct}] = -\Delta. \tag{17}$$

Since $\partial_z \log p = 1 - p$ and $\partial_z \log(1 - p) = -p$, substituting these derivatives and Eq. 17 gives the route-logit update in Eq. 5. For $0 < p < 1$ with both routes represented, the coefficient is positive: the update favors reasoning when $\Delta > 0$, favors direct responding when $\Delta < 0$, and has zero expected cross-route signal when $\Delta = 0$.

In the purely on-policy population case, $q_\mathrm{r} = p$ and $q_\mathrm{d} = 1 - p$, so Eq. 5 becomes $2p(1 - p)\Delta$. The expected route reward and its logit gradient are

$$J(p) = p\mu_\mathrm{r} + (1 - p)\mu_\mathrm{d}, \qquad \partial_z J = p(1 - p)\Delta. \tag{18}$$

In this on-policy population setting, the two updates have the same direction and stationary conditions ($\Delta = 0$ or limiting boundary policies). Forced-route proportions preserve the interior preference direction. This analysis applies to the simplified population update.

The per-rollout surrogate decomposes as

$$A_i^{\texttt{route}} = \begin{cases} \Delta + (r_i - \mu_\mathrm{r}), & \mathcal{R}_i = \texttt{reason}, \\ -\Delta + (r_i - \mu_\mathrm{d}), & \mathcal{R}_i = \texttt{direct}. \end{cases} \tag{19}$$

The within-route residual has zero conditional expectation, so this form and a pure route-level contrast $\pm\Delta$ give the same expected preference direction under the stated assumptions. The per-rollout form integrates directly into the token-level clipped GRPO/PPO objective. An empirical comparison with pure route-level contrast remains future work.

The reward has no length or latency penalty: equal total expected rewards give zero expected cross-route update. Efficiency gains are empirical; a length penalty could reinforce the direct-route bias observed in vanilla SFT (Table 5).

G.3 Safety and progress analysis

In Table 3, the same RL checkpoint is evaluated under three route policies. For the binary DAC and TTC metrics, the violation rate is $100\% - \text{score}\%$. Forced direct, forced reasoning, and adaptive STEER have DAC-violation rates of 3.5%, 3.0%, and 2.9%, and TTC-violation rates of 7.9%, 8.1%, and 7.7%, respectively. Adaptive STEER has higher TTC, DAC, EP, and PDMS point estimates, while NC is 0.1–0.2 points lower. This controlled comparison does not show a TTC-for-progress trade-off. The cross-system results in Tables 1 and 2 reflect different safety/progress balances.

NC is severity-aware (Appendix E.1). Collision-event counts are unavailable in the retained aggregate outputs, so a separate binary collision rate cannot be reconstructed.

H Sensitivity analysis

We further analyze the sensitivity of two key route-related hyper-parameters: the route calibration weight λ in SFT and the route loss weight λ_r in RL. Tables 14 and 15 report the results on NAVSIM v1.

For the SFT route calibration weight λ, increasing λ makes the initial routing distribution more balanced, thereby preserving a stronger dual-route prior. However, larger values do not further

Table 15: **Sensitivity analysis of the RL route loss weight λ_r.** A moderate route loss weight provides the best routing quality, efficiency, and planning performance.

λ_r	RouteAcc (%)↑	AUC (%)↑	AvgRegret↓	Avg. Resp. Tokens↓	PDMS↑
0	70.7	74.4	0.023	362	90.9
0.02	77.8	77.4	0.020	333	91.1
0.05	**81.1**	**79.1**	**0.018**	**315**	**91.2**
0.10	80.1	76.8	0.029	322	90.4
0.20	77.2	74.7	0.039	324	89.4

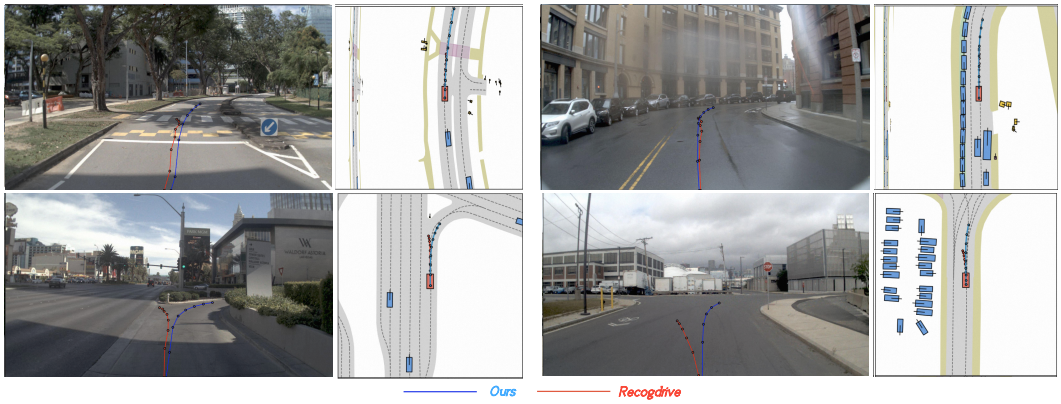

Figure 6: **Qualitative comparison with a representative baseline.** We compare STEER (blue) with ReCogDrive [14] (red) on representative prompts. STEER produces trajectories that better align with the road geometry and intended maneuver.

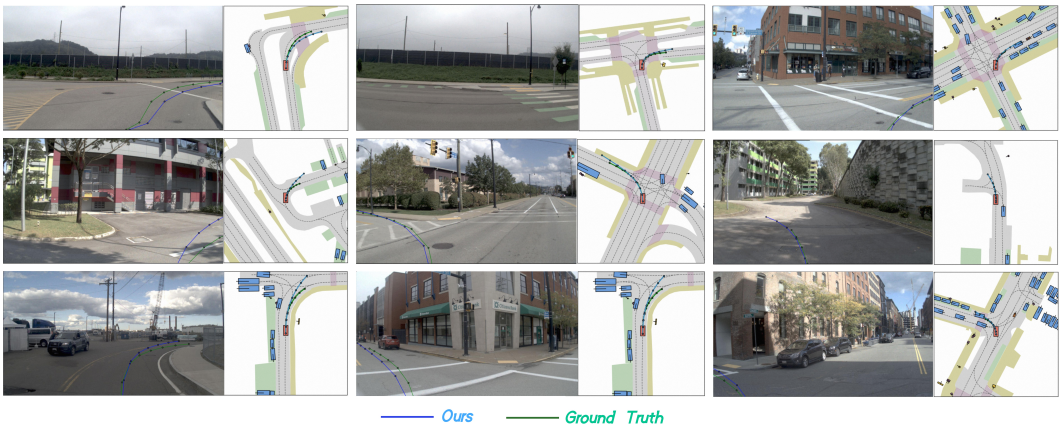

Figure 7: Representative failure cases on the NAVSIM benchmark.

improve downstream planning. The best PDMS is obtained at $\lambda = 0.10$, suggesting that moderate route calibration is sufficient to prevent early route collapse without over-regularizing the supervised objective.

For the RL route loss weight λ_r, introducing route-level supervision clearly improves routing quality over the $\lambda_r = 0$ baseline. As λ_r increases from 0 to 0.05, RouteAcc and AUC improve while AvgRegret decreases, and PDMS correspondingly rises from 90.9 to 91.2. At the same time, the average response length is reduced, indicating that better route learning also avoids unnecessary reasoning. When λ_r becomes too large, both routing quality and planning performance degrade, suggesting that excessive emphasis on route-level optimization can interfere with content generation.

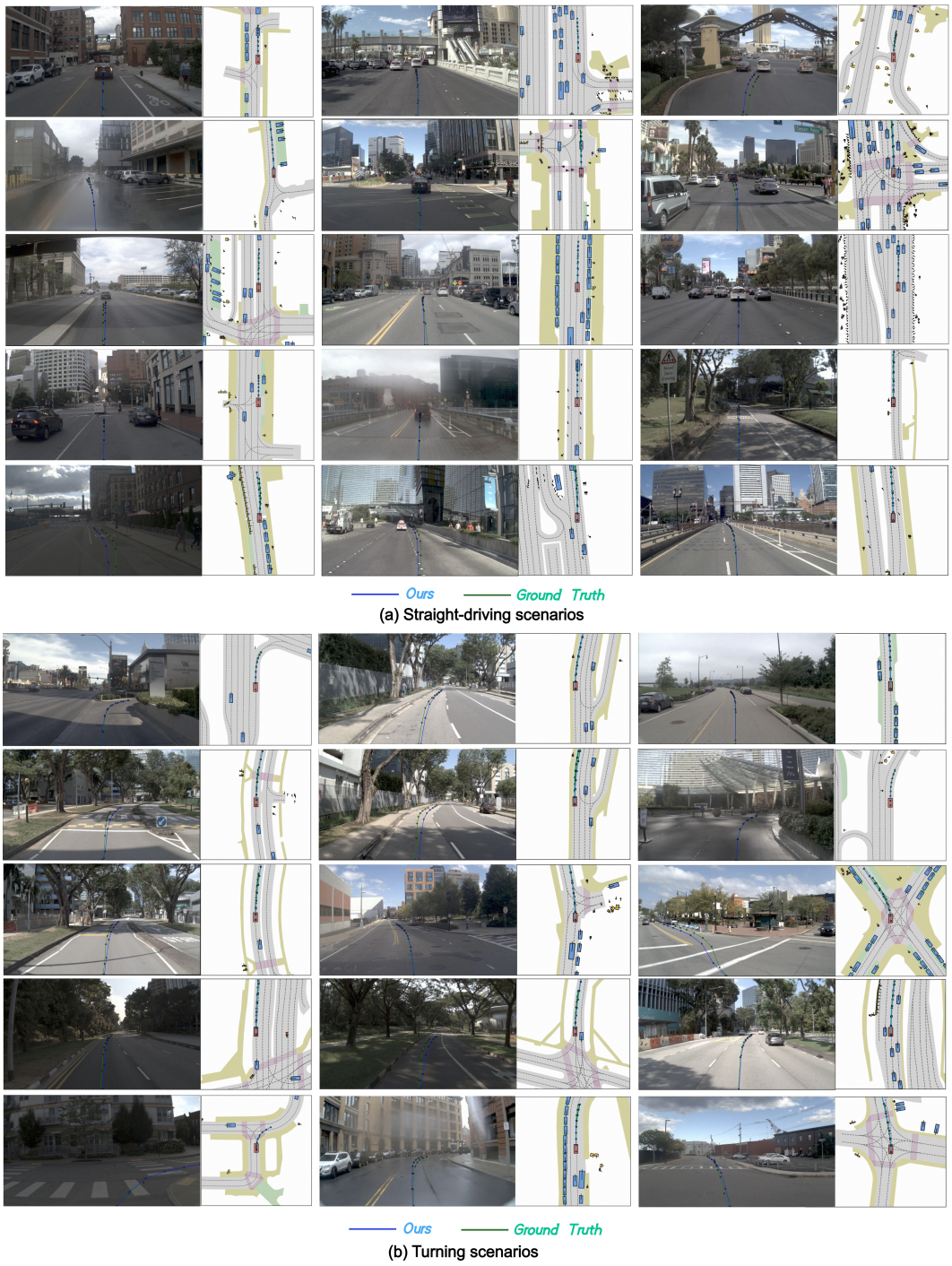

Figure 8: Additional qualitative results on the NAVSIM benchmark.

The thresholds τ_{low}, τ_{high}, and ϵ control allocation of the fixed 16-rollout budget. Table 4 tests this mechanism on/off, and Table 7 reports its training dynamics.

I Additional qualitative results

Figure 6 further compares STEER with ReCogDrive [14]. In both examples, our method produces trajectories that better align with the road geometry and intended maneuver, whereas the baseline tends to under-steer and deviate from the desirable path. These visualizations are consistent with the quantitative gains and illustrate that STEER improves both route selection and trajectory execution.

J Additional visualizations

We provide additional qualitative results in Figure 8. In both straight-driving and turning scenarios, the predicted trajectories closely follow the ground-truth trajectories, illustrating the effectiveness of STEER across diverse driving conditions.

We also show representative failure cases in Figure 7. When the model relies on a single front-view image, prediction errors may still occur in sharp-turn scenarios with limited contextual coverage. In such cases, the restricted field of view makes accurate long-horizon planning more difficult and can occasionally lead to lane-selection errors. These cases suggest that incorporating multi-view observations may further improve robustness in future work.

K Limitations

STEER has several limitations. First, our current experiments focus primarily on a single front-view input setting, which can limit route selection and planning accuracy in heavily occluded or interaction-rich scenarios. Second, the current formulation treats adaptive reasoning as a binary routing problem between a reasoning route and a direct-response route, and does not yet model finer-grained reasoning strategies such as variable reasoning depth. Third, although STEER avoids manually annotated scene-difficulty labels, route learning still depends on simulator-based reward quality. Extending the method to richer sensory inputs, more expressive routing spaces, and broader reward formulations is an important direction for future work. Evaluation is restricted to NAVSIM. Training-seed variance, entropy-threshold sensitivity, and rare high-regret routing errors remain to be studied.

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: The claims made in the abstract and introduction are fully supported by the experimental results in Section 4.

   Guidelines:

   - The answer [N/A] 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 [N/A] 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 explicitly discuss the limitations of our work, such as the reliance on single front-view input and the binary routing formulation, in Appendix K.

   Guidelines:

   - The answer [N/A] 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: [Yes]

Justification: Appendix G.2 states the assumptions and derives the expected route-logit contribution of the additional route objective in a two-route population setting.

Guidelines:

- The answer [N/A] 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: We provide detailed descriptions of our method in Section 3, and comprehensive implementation details, hyperparameters, and data preparation pipelines in Appendices B, E, and F.

Guidelines:

- The answer [N/A] 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 detail the data construction process in Appendix B.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.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://neurips.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) necessary to understand the results?

   Answer: [Yes]

   Justification: We specify the datasets, evaluation metrics, and implementation details in Section 4 and provide full hyperparameter settings in Appendix F.

   Guidelines:

   - The answer [N/A] 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: [Yes]

   Justification: Appendix G.1 reports mean $\pm$ standard deviation over three evaluation seeds for each listed fixed checkpoint; the corresponding entries in the main tables report means. These standard deviations quantify evaluation-seed variability.

   Guidelines:

   - The answer [N/A] 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: We explicitly disclose the compute resources (64 NVIDIA H20 GPUs) and distributed training setup (DeepSpeed ZeRO-3) in Section 4 and Appendix E.2. Section 4.4 additionally reports end-to-end inference latency on one H20 GPU and RL-stage GPU-hours ($64\times$ elapsed training hours), including rollout generation.

   Guidelines:

   - The answer [N/A] 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: Our research focuses on autonomous driving algorithms and strictly conforms to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] 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: We discuss the potential positive impacts on autonomous driving safety and efficiency in the Introduction, as well as the limitations that could lead to failure cases in Appendix K.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release high-risk models or scraped datasets that require specific safeguards beyond standard autonomous driving safety considerations.

Guidelines:

- The answer [N/A] 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: We properly cite the original papers for the public datasets and base models (e.g., NAVSIM, Qwen2.5-VL) used in our experiments in Section 4 and Appendix B.

Guidelines:

- The answer [N/A] 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 introduce a new data preparation pipeline and will provide the documentation and scripts alongside our code release.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: The research does not involve crowdsourcing or human subjects.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: The research does not involve human subjects.

Guidelines:

- The answer [N/A] 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 rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: We explicitly state the use of Qwen2.5-VL-8B as our base vision-language action model and Qwen2.5-VL-72B for data annotation in Section 3 and Appendix B.

    Guidelines:

    - The answer [N/A] 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 in the NeurIPS handbook for what should or should not be described.