NEURO-SYMBOLIC DRIVE: PLANNER-GROUNDED FAITHFUL REASONING FOR VISION-LANGUAGE DRIVING

Anonymous authors
Paper under double-blind review

ABSTRACT

Driving VLA models incorporating Chain-of-Thought (CoT) reasoning are attractive because they leverage pretrained VLM representations and expose intermediate decisions in natural language — yet current rationales often lack the step-by-step decision semantics needed to keep the rationale causally connected to the planned motion. We introduce NEURO-SYMBOLIC DRIVE, a neuro-symbolic driving framework that supervises a driving VLA with rule-grounded reasoning traces extracted directly from classical rule-based planners. Our key observation is that rule-based planners are symbolic AI systems that already function as executable reasoning engines: they reason about active safety constraints, search over candidate maneuvers, and select a final trajectory. We instrument these planners in simulation to capture both the executed trajectory and the internal decision trace at each rule-evaluation step. Each trace is serialized into structured rule-grounded reasoning and paired with the trajectory to fine-tune Qwen3.5-4B as a driving VLA. Because these traces are derived directly from the planner states that determine the action, they ensure reasoning is structurally coupled to motion generation by construction, rather than by post-hoc alignment. On our simulator-generated benchmark, detailed rule-grounded reasoning reduces ADE@3s from 0.47 to 0.26 and miss rate from 8.30% to 6.40% under three-camera perception, and from 0.54 to 0.26 and 10.13% to 5.99% under eight-camera perception. NEURO-SYMBOLIC DRIVE thus converts neuro-symbolic planning logic into structured supervision, ensuring generated reasoning is behaviorally grounded rather than a post-hoc narrative. All source code will be made publicly available upon acceptance.

1 INTRODUCTION

Classical rule-based planners and end-to-end models represent two complementary paradigms in autonomous driving: the former, which depends on deep perception modules to interpret raw sensor data, offers inspectable decision procedures and explicit safety constraints, but is brittle and limited by perception capabilities; the latter learns flexible representations directly from data but operates as a black box, lacking interpretability and explicit safety constraints. Before the rise of VLAs, interpretability was largely engineered into the planning stack through structured intermediate representations fed by deep perception — cost volumes, semantic maps, candidate trajectories, and other planning-oriented abstractions (Zeng et al., 2019; Casas et al., 2021; Hu et al., 2023; Jiang et al., 2023; Wang et al., 2021; Biswas et al., 2024). VLA systems offer a different path: by exposing reasoning in natural language, they enable driving agents to articulate scene understanding, interaction logic, and planned behaviors in a form that humans can inspect and audit (Chen et al., 2023; Sha et al., 2023; Sima et al., 2024; Xu et al., 2023; Shao et al., 2024; Hwang et al., 2024; Xing et al., 2025; Jiang et al., 2024; Tian et al., 2025; Wang et al., 2024; Nie et al., 2024; Yang et al., 2023; Cui et al., 2024; Ma et al., 2024; Huang et al., 2024).

The central difficulty is that readable reasoning is not necessarily action-bearing. Many driving VLAs generate language explanations while predicting trajectories either through a separate regression head or by discretizing continuous actions into tokens appended to the vocabulary. Such designs can produce accurate motions, but offer no guarantee that the stated rationale controls the final trajectory. CoT studies show that plausible explanations can be unfaithful (Turpin et al., 2023; Lanham et al.,

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

2023), and recent driving work similarly questions whether language reasoning is closely coupled to planning (Renz et al., 2025; Fu et al., 2025; Li et al., 2025; Song et al., 2025; Tang et al., 2026). The root cause is a supervision mismatch: current reasoning labels are typically generated post-hoc — by humans, by other VLMs, or as high-level causal annotations — rather than derived from the mechanism that actually determines the motion. DeepSeek-R1 suggests a principled fix: strong reasoning emerges when intermediate steps are grounded in verifiable, rule-based feedback rather than free-form imitation alone (DeepSeek-AI et al., 2025). We carry this principle into driving: the verifiable feedback is the execution trace of the rule-based planner that determined the trajectory.

Rule-based planners are a natural fit for this role: their internal execution unfolds as a structured decision procedure — enforcing safety rules to prune the action space, scoring competing candidate trajectories, and committing to the best feasible motion. Crucially, their internal execution traces are causally responsible for the selected motion — not post-hoc reconstructions. Deployed L4 systems — already operating driverless fleets across multiple cities worldwide — run precisely such planner stacks: multiple rule-based modules specialized per scenario family, whose decisions are explainable, inspectable, and by construction the causal source of every trajectory they select. Close in spirit, Alpamayo-R1 takes an important step toward decision-grounded supervision by constructing structured Chain-of-Causation (CoC) labels, integrating action generation, and applying post-training alignment to bridge reasoning and trajectory generation for long-tail driving (Wang et al., 2025). We are inspired by its focus on reasoning-action alignment, but target a different supervision source: rather than relying on human- or VLM-generated causal labels, we extract reasoning directly from the executable rule-based planner.

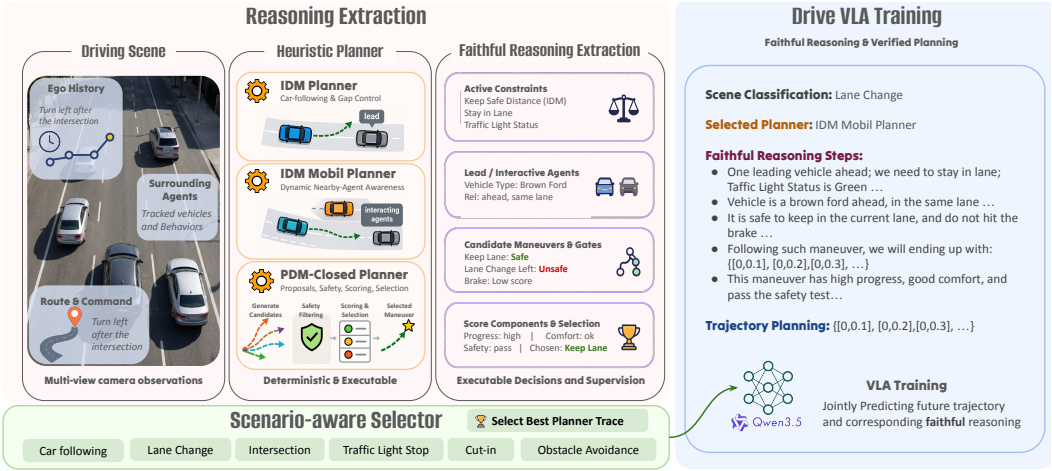

Figure 1: **Overview of NEURO-SYMBOLIC DRIVE.** The left side extracts structured decision traces from rule-based planners. For each simulated driving scene, planner traces expose active constraints, relevant agents, candidate maneuvers, safety gates, score components, and final selections. A scenario-aware selector chooses the trace best matched to the scene family. The right side uses these traces to supervise a driving VLA, so the model learns to jointly produce faithful reasoning and the corresponding future trajectory.

We propose NEURO-SYMBOLIC DRIVE, a neuro-symbolic framework that converts rule-based planner execution traces into structured supervision for a driving VLA. Given a simulated scene, we run complementary planners, record their internal decision traces, select the planner best matched to the scenario family, and convert the chosen trace into rule-grounded reasoning paired with the resulting trajectory. Because the reasoning trace and the trajectory arise from the same planner execution, the rationale is action-bearing by construction — not by post-hoc alignment. Figure 1 summarizes this pipeline. Our main contributions are as follows.

- We introduce NEURO-SYMBOLIC DRIVE, a neuro-symbolic framework that repurposes rule-based planner execution traces as structured reasoning supervision for vision-language driving agents.
- We design a lightweight abstraction that maps heterogeneous raw planner states from multiple teachers to a unified four-slot reasoning schema, enabling cross-teacher learning without planner-specific supervision vocabularies.

- We demonstrate that rule-grounded reasoning improves driving performance and produces more behaviorally consistent rationales than trajectory-only supervision.

2 Related Work

Language Reasoning in Driving VLAs. Early language-augmented systems translate object vectors, route context, or traffic descriptions into prompts or planning states (Chen et al., 2023; Sha et al., 2023; Cui et al., 2025), while recent VLA models directly connect multi-view observations to explanations, decisions, controls, or trajectories (Sima et al., 2024; Xu et al., 2023; Shao et al., 2024; Hwang et al., 2024; Xing et al., 2025; Jiang et al., 2024; Tian et al., 2025; Nie et al., 2024; Wang et al., 2024; Yuan et al., 2024). Agentic approaches study instruction following, rule reasoning, experience use, and interaction in LLM/VLM driving agents (Yang et al., 2023; Cui et al., 2024; Ma et al., 2024; Huang et al., 2024), while recent VLA systems explore adaptive reasoning, RL, instructed action generation, and decoupled high-level reasoning (Zhou et al., 2025; Li et al., 2025; Ye et al., 2026; Fu et al., 2025; Tao et al., 2026). These works improve interpretability but often rely on free-form rationales or post-hoc alignment; Neuro-Symbolic Drive instead derives reasoning directly from the executable planner state that produced the trajectory.

Chain-of-Thought Reasoning, Faithfulness, and the Supervision Mismatch. CoT (Wei et al., 2022) and process reward models (Lightman et al., 2024) highlight the value of step-level supervision, while OpenAI o1 (OpenAI, 2024) and DeepSeek-R1 (DeepSeek-AI et al., 2025) emphasize rule-based verifiable rewards over free-form imitation. EMMA (Hwang et al., 2024) tokenizes continuous controls with the pretrained VLM tokenizer, placing actions and language reasoning in a shared embedding space. DriveVLM (Tian et al., 2025) uses language as an intermediate reasoning layer between perception and trajectory generation; DriveLM (Sima et al., 2024) structures reasoning as a graph of visual question-answering steps; DriveCoT (Wang et al., 2024) and Reason2Drive (Nie et al., 2024) generate supervised thinking-process traces from expert demonstrations. Alpamayo-R1 (Wang et al., 2025) constructs structured CoC labels and applies post-training alignment to bridge decision-grounded reasoning and trajectory generation. CoT studies warn that plausible explanations can omit or distort the true factors behind a model output (Turpin et al., 2023; Lanham et al., 2023), and imitation policies may exploit correlated but non-causal signals (de Haan et al., 2019). In autonomous driving, these concerns surface concretely as language-action mismatch, weak reasoning-planning coupling, and confounding in end-to-end planning (Renz et al., 2025; Fu et al., 2025; Li et al., 2025; Song et al., 2025; Tang et al., 2026). The root cause is a supervision mismatch: existing approaches generate reasoning labels post-hoc — as text narrations, causal annotations, or latent surrogates — rather than from the mechanism that determined the motion. Neuro-Symbolic Drive addresses this at the source: reasoning traces are abstracted directly from the planner computation that selected the supervision trajectory, not constructed after the fact.

Rule-Based Planners and Decision-Grounded Supervision. Interpretable planning predates driving VLAs. Neural planners and map-perceive-predict-plan systems expose detections, semantic maps, and cost volumes as structured intermediate variables (Zeng et al., 2019; Casas et al., 2021); end-to-end planning-oriented systems retain task decomposition (Hu et al., 2023; Jiang et al., 2023); and IVMP and QuAD emphasize interpretable factors and candidate evaluation (Wang et al., 2021; Biswas et al., 2024). Closed-loop benchmarks show that rule-based planners remain competitive and complementary to learned models (Caesar et al., 2021; Dauner et al., 2023; Ghosh et al., 2026). Rule-based planners expose an internal execution state—active constraints, candidate proposals, safety checks, and score components—causally synchronized with the selected trajectory. Decision-grounded supervision takes several forms: DriveMLM aligns language output with planning states (Cui et al., 2025), DriveCoT generates thinking-process labels from rule-based experts (Wang et al., 2024), Hydra-MDP distills from human and rule-based teachers (Li et al., 2024), and DiMA distills LLM knowledge into efficient driving models (Hegde et al., 2025). Program-grounded learning further suggests that executable structure can be serialized and distilled into neural models (Shlapentokh-Rothman et al., 2025), while NaviDriveVLM and RAD-LAD explore decoupled or hybrid language-planner systems (Tao et al., 2026; Ghosh et al., 2026). These approaches use planners as baselines, teachers, safety modules, or inference-time collaborators. Neuro-Symbolic Drive differs by treating planner execution traces as the primary supervision source: the reasoning and

the trajectory arise from the same transparent computation, making the rationale action-bearing by construction rather than by post-hoc alignment.

3 METHOD

We treat the faithfulness gap — the disconnect between a model's stated rationale and what actually determined its motion — as a supervision sourcing problem: reasoning labels should be derived from the mechanism that determined the motion, not constructed after the fact.

3.1 PROBLEM FORMULATION

At timestep t, the driving input is $x_t := (I_t, h_t, g_t)$, where I_t denotes synchronized multi-view images, h_t is the recent ego-state history, and g_t is the route or mission context. A deterministic planner $p \in \mathcal{P}$ (the set of available rule-based planners) receives x_t and returns both a future trajectory y_t and an internal execution trace z_t: $(y_t, z_t) = p(x_t)$.

The trace z_t is the raw program state — active constraints, route context, candidate proposals, safety checks, score components, and final trajectory choice — not free-form text. An abstraction operator $\mathcal{A}$ converts it into a reasoning sequence r_t: $r_t := \mathcal{A}(z_t)$.

The training target for the VLA is the concatenated sequence

$$u_t := [\texttt{<REASONING>}\ r_t\ \texttt{</REASONING>}\ \texttt{<PLANNING>}\ y_t\ \texttt{</PLANNING>}]. \quad (1)$$

The key design choice is that r_t is derived from the same planner state that produced y_t, making the reasoning action-bearing by construction rather than merely correlated with the trajectory.

3.2 PLANNER TRACE EXTRACTION

We use three complementary rule-based planners in simulation: Intelligent Driver Model Planner (IDM), IDM MOBIL Planner, which augments IDM with the MOBIL lane-change criterion (Minimizing Overall Braking Induced by Lane Changes) (Treiber & Kesting, 2009), and Predictive Driver Model Closed Planner (PDM-Closed) (Dauner et al., 2023). IDM provides stable centerline car-following traces through lead-agent, headway, closing-speed, safety-gap, and acceleration terms, while IDM MOBIL Planner augments IDM with interaction-aware lane-change and gap-selection logic for cut-in, merge, queueing, and close-following cases. PDM-Closed follows a proposal-based closed-loop procedure: it generates candidate trajectories, applies safety and progress checks, scores surviving proposals, and selects the final maneuver (Hallgarten et al., 2024; Dauner et al., 2023; Karnchanachari et al., 2024). These three teachers therefore expose complementary reasoning patterns: gap control, interaction-aware maneuver selection, and propose-evaluate-select planning.

This multi-planner design mirrors the architecture of production L4 autonomous driving stacks, which operate as a hierarchy of specialized planners — one for longitudinal car-following and ACC-style control, another for lane-change and merge decisions, another for intersection and complex-maneuver planning — rather than delegating all scenes to a single universal module. We adopt this principle directly in data construction: for each scenario family, closed-loop evaluation identifies which teacher produces the most reliable behavior, and that planner's trace and trajectory become the default supervision. This teacher-selection step is the data-construction analog of L4 dispatch — the right planner for the right scene. When multiple planners are complementary, we also retain multiple successful rollouts for the same scene, increasing supervision diversity without forcing a single teacher to cover scenarios outside its competence.

Figure 2 summarizes the merged scenario coverage used for planner trace extraction. The distribution spans 59 scenario types and roughly 330 hours of simulated duration in the dataset, with the largest individual scenario families contributing only a small fraction of the full corpus. This broad but relatively balanced coverage is important for our setting: the model sees planner traces from routine lane following, longitudinal car-following, traffic-light traversal, turning, interaction, and long-tail maneuver scenarios, rather than learning reasoning from a narrow planner-specific subset.

Instrumentation is read-only: we record the planner program state without changing the trajectory computation itself. The extracted fields include route context, relevant agents, safety constraints, candidate outcomes, score components, selected proposal identity, and fallback status, when applicable.

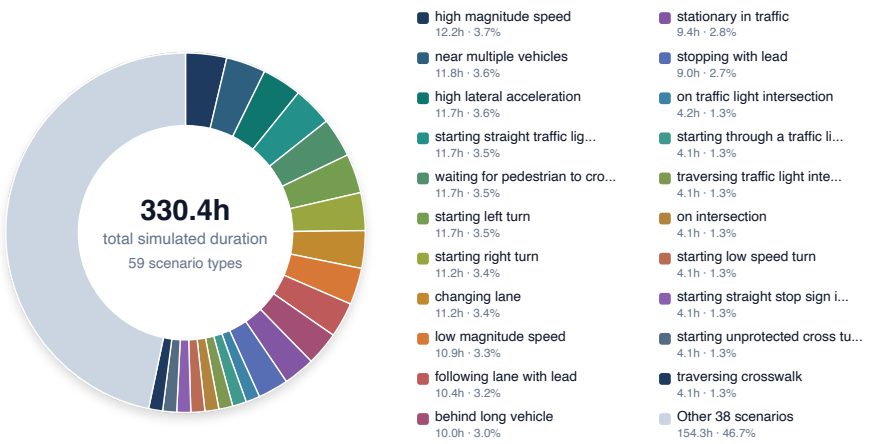

Figure 2: **Merged dataset duration by scenario type.** We compute each scenario's simulated duration as the number of examples multiplied by the average scenario duration. The chart shows the top 21 scenario types and groups the remaining 38 types as *Other*, illustrating the diverse scenario coverage used to extract rule-grounded reasoning traces.

Because the trace is captured from the same execution that produces the trajectory, every reasoning target remains synchronized with the exact motion used for supervision.

3.3 Reasoning Abstraction and Serialization

Raw program traces are not suitable supervision targets by themselves. First, the three planners expose different internal vocabularies: IDM is organized around lead vehicles, headway, desired gaps, and acceleration updates; IDM MOBIL Planner adds lateral incentives and lane-change safety checks; PDM-Closed exposes proposal banks, safety filters, and score components. A single VLA cannot learn unified decision semantics from these heterogeneous variable names alone. Second, raw traces mix action-relevant evidence with implementation artifacts such as loop counters, temporary buffers, and intermediate accumulators. Directly imitating them would encourage memorization of planner-specific execution details rather than learning reusable driving logic. Figure 3 provides a concrete example of the resulting serialized reasoning trace together with the corresponding top-down predicted trajectory. We therefore serialize each trace into a compact schema with four slots: $r_t := \left[s_t^{\text{scene}}, s_t^{\text{constraint}}, s_t^{\text{candidate}}, s_t^{\text{decision}}\right]$. The *scene* slot summarizes the route and ego context. The *constraint* slot specifies the currently active safety or rule signal. The *candidate* slot describes either the gap update sequence or the proposal-screening outcomes. The *decision* slot states the chosen maneuver and its immediate consequence for future motion.

We instantiate two text realizations. The *concise* variant keeps the schema short and quantitative — exposing *what* the planner decided at each step — yielding a stable supervision target with limited token overhead. The *detailed* variant expands the same slots into a step-by-step narrative — exposing *how* each decision was reached through candidate filtering and rejection rationale. Both variants share the same four-slot semantics; they differ only in the depth of the decision trace exposed. This lets us test whether performance gains come from the decision structure or the depth of the explanation.

The target sequence is serialized as rule-grounded reasoning first, followed by the future trajectory (Eq. 1). Explicit delimiters are used in the implementation, but serialized targets exclude planner-specific method names and simulator-specific debug fields. The reasoning sequence only preserves decision semantics that remain stable between teachers and scenarios.

4 Experiments

The goal of our experiments is to test whether rule-grounded reasoning improves autonomous-driving planning, rather than to rank rule-based planners as standalone systems. Because our supervision is extracted from simulator rollouts, all training, inference, and testing are conducted in simulation. This

gives us synchronized observations, planner traces, and target trajectories, and therefore a controlled setting for isolating the effect of reasoning depth and teacher selection.

4.1 Experimental Setup

We evaluate on the NSD-Sim dataset, a simulator-generated corpus built from 520 nuPlan scenarios and 77,558 timestep-level training examples. Each example contains synchronized multi-view images, ego history, mission context, a planner-generated trajectory, and the corresponding planner trace. We use a scenario-level split to avoid timestep leakage, with 70% of scenarios for training, 15% for validation, and 15% for testing. We evaluate two camera settings: a 3-camera setting with front, front-left, and front-right views, and an 8-camera setting with the full surrounding camera suite. Following common planning evaluation protocols in autonomous driving (Caesar et al., 2020; 2021; Hu et al., 2023; Jiang et al., 2023; Shao et al., 2024; Hwang et al., 2024; Xing et al., 2025), we evaluate the predicted future trajectory against the planner-generated target trajectory (which serves as ground truth in our simulation setting) using Average ADE, ADE/FDE at 1s/2s/3s, AHE/FHE at 3s, and Miss Rate at 3s. ADE/FDE measure displacement error, AHE (Average Heading Error) / FHE (Final Heading Error) measure heading error, and Miss Rate measures the fraction of generations that exceed the predefined trajectory error threshold.

We instantiate the driving backbone with Qwen3.5-4B (Team, 2026) and fine-tune all parameters with standard supervised learning. Unless otherwise noted, all variants use the same optimizer, tokenizer, image preprocessing, maximum sequence length, and number of epochs. The controlled variables are the camera setting, the reasoning level, and the planner source used to construct the training data. We compare three levels of reasoning supervision that reflect increasing exposure to the planner's decision process. *Without reasoning* trains the model on trajectories alone, with no intermediate language. *Concise reasoning* exposes the active safety constraint and the selected maneuver at each planning step — the *what* of the decision, without per-step rationale. *Detailed reasoning* additionally exposes candidate filtering, rejection causes, and selection rationale at each step — the *how* of the trajectory, showing why each rule evaluation led to the chosen motion. This comparison tests whether exposing more of the planner's decision process improves trajectory generation, and whether gains come from decision structure or explanation depth. Since all conditions use the same planner-generated trajectories as supervision targets, the comparison is internally controlled — improvements reflect differences in the supervisory signal rather than differences in the reference trajectory.

4.2 Main Results: Camera Setting and Reasoning Depth

Table 1: **Main planning results.** All rows use the full multi-planner data construction pipeline. The comparison isolates whether rule-grounded reasoning depth improves planning performance under 3-camera and 8-camera perception settings. Lower is better for all metrics.

Reasoning level	Average ADE↓	ADE@1s↓	ADE@2s↓	ADE@3s↓	FDE@1s↓	FDE@2s↓	FDE@3s↓	AHE@3s↓	FHE@3s↓	MR@3s↓
Three Cameras										
Without reasoning	0.32	0.22	0.33	0.47	0.21	0.47	0.68	1.99	1.85	8.30%
Concise reasoning	0.20	0.13	0.19	0.28	**0.14**	0.29	0.52	1.04	1.05	6.90%
Detailed reasoning	**0.19**	**0.12**	**0.18**	**0.26**	**0.14**	**0.27**	**0.49**	**1.02**	**1.01**	**6.40%**
Eight Cameras										
Without reasoning	0.36	0.24	0.31	0.54	0.28	0.44	0.88	1.59	1.48	10.13%
Concise reasoning	0.20	**0.13**	0.18	0.27	**0.13**	0.28	0.49	1.03	0.97	6.50%
Detailed reasoning	**0.19**	**0.13**	**0.17**	**0.26**	**0.13**	**0.27**	**0.46**	**0.99**	**0.92**	**5.99%**

Main results. Table 1 shows a clear and consistent trend: adding rule-grounded reasoning improves planning quality under both perception settings, and richer reasoning consistently yields the best or tied results. In the three-camera setting, concise reasoning already produces a large gain over the no-reasoning baseline, reducing Average ADE from 0.32 to 0.20, ADE@3s from 0.47 to 0.28, and MR@3s from 8.30% to 6.90%. Detailed reasoning improves the same model further, lowering Average ADE to 0.19, ADE@3s to 0.26, FDE@3s to 0.49, and MR@3s to 6.40%. This pattern suggests that rule-grounded reasoning helps not only on long-horizon failure cases, but also on the overall trajectory quality summarized by Average ADE.

The eight-camera setting makes the same pattern even clearer. The no-reasoning baseline is substantially worse here, with 0.36 Average ADE and 10.13% MR@3s, indicating that simply exposing the model to more views does not by itself produce stable planning. Once rule-grounded reasoning is added, performance improves sharply: concise reasoning reduces Average ADE to 0.20 and MR@3s to 6.50%, while detailed reasoning reaches the best overall result with 0.19 Average ADE, 0.26 ADE@3s, 0.46 FDE@3s, 0.99 AHE@3s, 0.92 FHE@3s, and 5.99% MR@3s. This result matters for our main claim. The gain does not come from a richer sensor suite alone; it comes from giving the model intermediate decision semantics that organize how perceptual evidence translate into behavior.

Comparing concise and detailed reasoning is also informative. Concise reasoning — which exposes *what* the planner decided at each step — already captures most of the improvement, indicating that active constraints and selected maneuvers carry the highest-value supervisory signal. Detailed reasoning — which additionally exposes *how* each decision was reached through candidate filtering and rejection rationale — still brings a further, repeatable gain on long-horizon displacement, heading, and miss-rate metrics, implying that step-by-step decision semantics provide additional information beyond the final maneuver label. Taken together, these results support the central thesis: reasoning helps not because it makes the model more verbose, but because rule-grounded reasoning makes the planning target easier to align with and learn from.

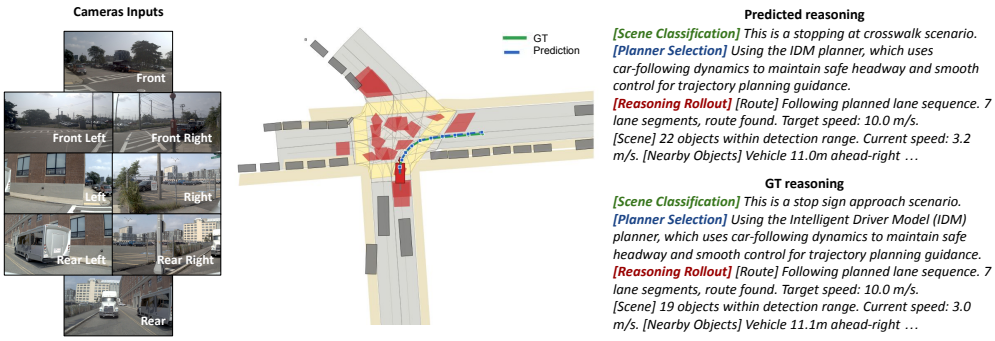

Figure 3: **Generated reasoning and trajectory visualization.** We show a qualitative prediction from the trained model with the generated rule-grounded reasoning trace and the corresponding top-down trajectory comparison. The example illustrates that the model does not merely output waypoints: it first identifies the active driving context and decision rationale, then predicts a trajectory that is visually aligned with the ground-truth motion.

Figure 3 gives a qualitative view of the same trend. The generated reasoning trace exposes the scene-level constraint and maneuver choice before the model emits future waypoints, and the top-down overlay makes it possible to inspect whether the planned motion is consistent with that reasoning.

4.3 Ablation: Planner Sources and Data Organization

Table 2: **Planner-source ablation.** All rows use detailed rule-grounded reasoning. The comparison varies only which rule-based teacher supplies reasoning traces and trajectories, and how multiple teachers are selected or organized. Lower is better for all metrics.

Teacher source	Selection policy	Average ADE↓	ADE@1s↓	ADE@2s↓	ADE@3s↓	FDE@1s↓	FDE@2s↓	FDE@3s↓	AHE@3s↓	FHE@3s↓	MR@3s↓
Three Cameras											
IDM	single teacher	0.31	0.21	0.25	0.41	0.27	0.49	0.86	1.70	1.72	10.21%
PDM-Closed	single teacher	0.28	0.15	0.26	0.40	0.22	0.39	0.57	1.30	1.39	8.47%
IDM MOBIL Planner	single teacher	3.17	3.04	3.45	3.60	2.62	3.61	6.02	6.65	4.81	30.16%
All Planners	random teacher per scene	0.22	0.15	0.20	0.28	0.15	0.30	0.54	**1.02**	1.07	**6.27%**
All Planners	scenario-aware selection	**0.19**	**0.12**	**0.18**	**0.26**	**0.14**	**0.27**	**0.49**	**1.02**	**1.01**	6.40%
Eight Cameras											
IDM	single teacher	0.29	0.21	0.27	0.40	0.24	0.47	0.78	1.67	1.64	10.32%
PDM-Closed	single teacher	0.24	0.16	0.22	0.34	0.18	0.35	0.59	1.26	1.19	7.39%
IDM MOBIL Planner	single teacher	3.13	2.62	3.08	3.70	2.76	3.84	5.35	6.12	5.25	28.91%
All Planners	random teacher per scene	0.20	**0.13**	0.19	0.27	0.14	0.28	0.51	1.06	0.97	6.22%
All Planners	scenario-aware selection	**0.19**	**0.13**	**0.17**	**0.26**	**0.13**	**0.27**	**0.46**	**0.99**	**0.92**	**5.99%**

Planner-source ablation. The single-teacher rows show that no rule-based planner is a universally reliable source of supervision. PDM-Closed is stronger than IDM on most metrics in both camera settings, while IDM MOBIL Planner alone performs poorly when used as the only teacher. This result

is expected rather than a data bug: IDM MOBIL Planner is primarily designed for car-following with lane-change incentives and nearby-agent interaction, but it has limited ability to handle route-level turning, intersection traversal, and many traffic-control cases. When forced to supervise the entire benchmark, it produces very large errors on those out-of-scope scenes. Its value appears in a narrower set of lateral maneuver and interaction-heavy scenarios, which is why we use it as a specialized teacher inside the multi-planner mixture rather than as a universal teacher.

Multi-teacher data is consistently better than any single-teacher setting: with three cameras, random teacher per scene reduces MR@3s from 8.47% for the best single teacher to 6.27%; with eight cameras, scenario-aware selection reaches the best overall result with 0.19 Average ADE and 5.99% MR@3s. The comparison between the two multi-planner policies is also informative. In the random-teacher setting, each scene still contributes only one retained training example: we first form a pool of valid rollouts from the three planners, then randomly choose one teacher trace and trajectory for that scene. Thus, the dataset size does not grow relative to the single-teacher and scenario-aware settings, and the gain cannot be attributed to a simple increase in supervision volume. Scenario-aware selection is the cleaner policy for our main claim because it assigns one default teacher to each scenario family according to closed-loop reliability, reducing conflicting targets while preserving the same dataset scale. This supports our design choice of using planner capability estimates to decide which reasoning trace and trajectory should supervise each scenario family.

4.4 Quality of the Rule-based Planner Teachers

Figure 5 reports the closed-loop quality of the rule-based teachers used to construct rule-grounded supervision over the 59 scenario families observed in our dataset. We use the Composite Closed-Loop Score Reactive (CCLS-R) to compare planners across scenario families. For each rollout, CCLS-R combines the major closed-loop safety and progress terms:

$$\begin{aligned}\mathrm{CCLS-R} := {} & 0.35C_{\mathrm{collision}} + 0.25C_{\mathrm{drivable}} + 0.15C_{\mathrm{ttc}} \\ & + 0.10C_{\mathrm{direction}} + 0.10C_{\mathrm{comfort}} + 0.05\min(P_{\mathrm{progress}}, 1).\end{aligned} \tag{2}$$

This metric is intentionally broader than a binary success rate: it captures whether a planner produces usable closed-loop behavior even when nuPlan's strict aggregate score is low. For data construction, CCLS-R is used to rank teachers within each scenario family. We then use the best-CCLS-R teacher as the default source of reasoning traces and trajectories, after applying the simulator route-success filter. A rollout is retained for supervision only if it completes the full planned trajectory without collision or off-road violation. We do not apply an additional global CCLS-R threshold; CCLS-R guides teacher selection, while the route-success filter determines whether a rollout is kept for VLA training.

Figures 4 and 5 support a mixture-of-teachers data strategy. PDM-Closed is the strongest generalist on most intersection, turn, and traffic-light scenarios; IDM MOBIL Planner is more competitive in lateral maneuver and static-obstacle cases; and IDM remains useful in conservative crawl-and-yield settings where partial closed-loop credit matters. Thus, the planner used for supervision is not chosen arbitrarily: it is selected from closed-loop evidence about which rule-based teacher is most reliable for each scenario family.

5 Conclusion

We introduced Neuro-Symbolic Drive, a neuro-symbolic framework that grounds vision-language driving models in executable planner semantics rather than free-form post-hoc rationalization. Rule-based planners encode decision-making logic—constraint activation, proposal filtering, safety validation, and outcome ranking—whose execution traces provide faithful reasoning supervision for driving VLAs. We instrument three complementary planners and route each scenario to its most capable teacher, mirroring the L4 hierarchy of specialized planners while keeping reasoning targets synchronized with their trajectories. Empirically, detailed rule-grounded reasoning reduces ADE@3s by 45% and miss rate by 23% under three-camera perception, and brings an unstable eight-camera baseline to parity with the three-camera setting. These results support executable decision semantics as a strong supervision signal.

Beyond autonomous driving, these results suggest a broader principle: rather than imitating human explanations or generic chain-of-thought, faithful reasoning can be obtained by treating any

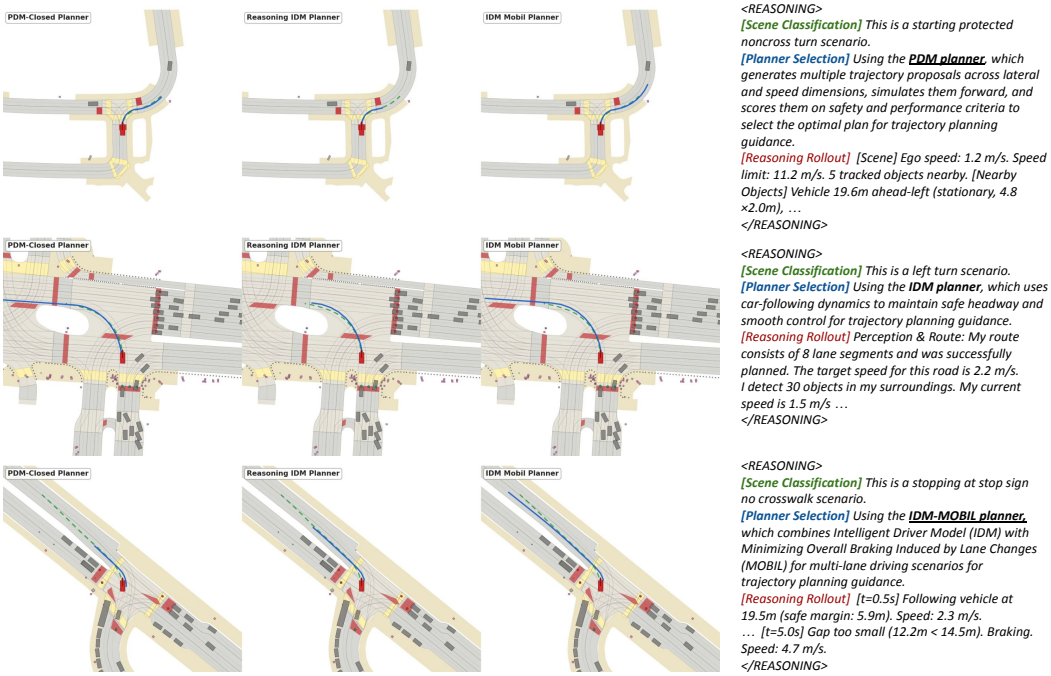

Figure 4: **Same-scene planner comparison.** We visualize the three heuristic teachers on matched scenarios and timestamps. The examples show that different rule-based planners can make distinct closed-loop choices under the same observation, which motivates selecting the teacher trace according to scenario-level closed-loop quality rather than treating any single planner as a universal expert.

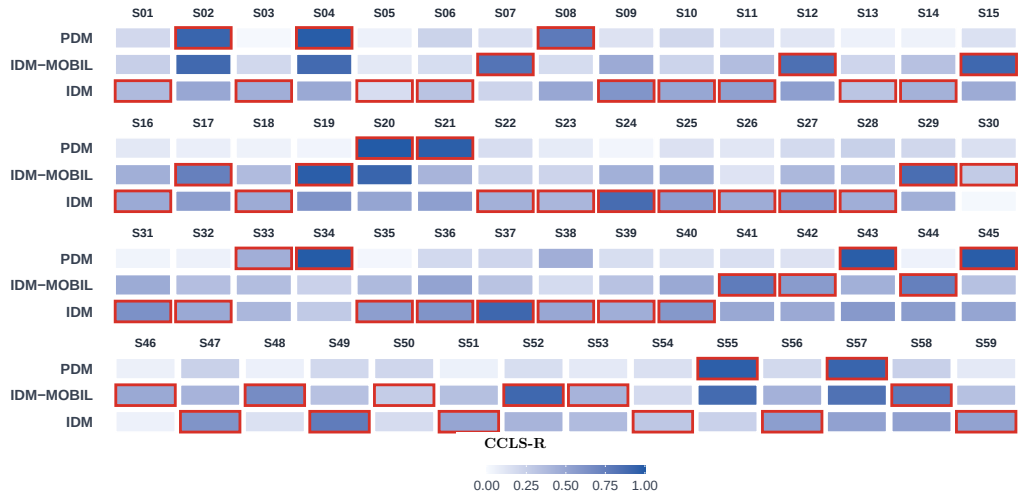

Figure 5: **Closed-loop teacher quality.** Each indexed column is a shuffled scenario family in the dataset, and each row is one teacher planner. Color indicates CCLS-R (higher is better). Red boxes mark the highest-CCLS-R teacher for each scenario, which guides scenario-aware selection of the planner trace and trajectory used for VLA supervision. Scenario-index mappings are listed in Appendix A.2.

deterministic domain-specific system — planner, constraint solver, rule engine — as an executable oracle, instrumenting its decision trace, and using it to supervise a generalist model. This also opens a path toward joint training with general reasoning tasks: planner traces provide verifiable embodied decisions, while general reasoning data may improve abstraction, language consistency, and compositional problem solving.

AI USE STATEMENT

Generative AI tools assisted with manuscript preparation and LaTeX formatting. The pretrained vision-language model used in our experiments is specified in Section 4.1. The authors take responsibility for the final content of the paper, including its claims and reported results.

REFERENCES

Sourav Biswas, Sergio Casas, Quinlan Sykora, Ben Agro, Abbas Sadat, and Raquel Urtasun. Quad: Query-based interpretable neural motion planning for autonomous driving. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 14236–14243. IEEE, 2024.

Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020.

Holger Caesar, Juraj Kabzan, Kok Seang Tan, Whye Kit Fong, Eric Wolff, Alex Lang, Luke Fletcher, Oscar Beijbom, and Sammy Omari. nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles. *arXiv preprint arXiv:2106.11810*, 2021.

Sergio Casas, Abbas Sadat, and Raquel Urtasun. Mp3: A unified model to map, perceive, predict and plan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14403–14412, 2021.

Long Chen, Oleg Sinavski, Jan Hünermann, Alice Karnsund, Andrew James Willmott, Danny Birch, Daniel Maund, and Jamie Shotton. Driving with llms: Fusing object-level vector modality for explainable autonomous driving. *arXiv preprint arXiv:2310.01957*, 2023.

Can Cui, Yunsheng Ma, Sung-Yeon Park, Zichong Yang, Yupeng Zhou, Peiran Liu, Juanwu Lu, Juntong Peng, Jiaru Zhang, Ruqi Zhang, et al. LLM4AD: Large language models for autonomous driving – concept, review, benchmark, experiments, and future trends. *arXiv preprint arXiv:2410.15281*, 2024.

Erfei Cui, Wenhai Wang, Zhiqi Li, Jiangwei Xie, Haoming Zou, Hanming Deng, Gen Luo, Lewei Lu, Xizhou Zhu, and Jifeng Dai. Drivemlm: Aligning multi-modal large language models with behavioral planning states for autonomous driving. *Visual Intelligence*, 3(1):22, 2025. doi: 10.1007/s44267-025-00095-w.

Daniel Dauner, Marcel Hallgarten, Andreas Geiger, and Kashyap Chitta. Parting with misconceptions about learning-based vehicle motion planning. In *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 1268–1281. PMLR, 2023.

Pim de Haan, Dinesh Jayaraman, and Sergey Levine. Causal confusion in imitation learning. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Nature*, 645:633–638, 2025. doi: 10.1038/s41586-025-09422-z. arXiv:2501.12948.

Haoyu Fu, Diankun Zhang, Zongchuang Zhao, Jianfeng Cui, Dingkang Liang, Chong Zhang, Dingyuan Zhang, Hongwei Xie, Bing Wang, and Xiang Bai. Orion: A holistic end-to-end autonomous driving framework by vision-language instructed action generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 24823–24834, 2025.

Anurag Ghosh, Srinivasa Narasimhan, Manmohan Chandraker, and Francesco Pittaluga. Rad-lad: Rule and language grounded autonomous driving in real-time. *arXiv preprint arXiv:2603.28522*, 2026.

Marcel Hallgarten, Julian Zapata, Martin Stoll, Katrin Renz, and Andreas Zell. Can vehicle motion planning generalize to realistic long-tail scenarios? In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5388–5395, 2024. doi: 10.1109/IROS58592.2024.10803052.

Deepti Hegde, Rajeev Yasarla, Hong Cai, Shizhong Han, Apratim Bhattacharyya, Shweta Mahajan, Litian Liu, Risheek Garrepalli, Vishal M. Patel, and Fatih Porikli. Distilling multi-modal large language models for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 27575–27585, 2025.

Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, et al. Planning-oriented autonomous driving. In *CVPR*, 2023.

Yidong Huang, Jacob Sansom, Ziqiao Ma, Felix Gervits, and Joyce Chai. Drivlme: Enhancing llm-based autonomous driving agents with embodied and social experiences. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3153–3160. IEEE, 2024.

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.

Bo Jiang, Shaoyu Chen, Qing Xu, Bencheng Liao, Jiajie Chen, Helong Zhou, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. Vad: Vectorized scene representation for efficient autonomous driving. In *ICCV*, 2023.

Bo Jiang, Shaoyu Chen, Bencheng Liao, Xingyu Zhang, Wei Yin, Qian Zhang, Chang Huang, Wenyu Liu, and Xinggang Wang. Senna: Bridging large vision-language models and end-to-end autonomous driving. *arXiv preprint arXiv:2410.22313*, 2024.

Napat Karnchanachari, Dimitris Geromichalos, Kok Seang Tan, Nanxiang Li, Christopher Eriksen, Shakiba Yaghoubi, Noushin Mehdipour, Gianmarco Bernasconi, Whye Kit Fong, Yiluan Guo, et al. Towards learning-based planning: The nuplan benchmark for real-world autonomous driving. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 629–636. IEEE, 2024.

Tamera Lanham, Anna Chen, Ansh Radhakrishnan, Benoit Steiner, Carson Denison, Danny Hernandez, Dustin Li, Esin Durmus, Evan Hubinger, Jackson Kernion, et al. Measuring faithfulness in chain-of-thought reasoning. *arXiv preprint arXiv:2307.13702*, 2023.

Yue Li, Meng Tian, Dechang Zhu, Jiangtong Zhu, Zhenyu Lin, Zhiwei Xiong, and Xinhai Zhao. Drive-r1: Bridging reasoning and planning in vlms for autonomous driving with reinforcement learning. *arXiv preprint arXiv:2506.18234*, 2025.

Zhenxin Li, Kailin Li, Shihao Wang, Shiyi Lan, Zhiding Yu, Yishen Ji, Zhiqi Li, Ziyue Zhu, Jan Kautz, Zuxuan Wu, Yu-Gang Jiang, and José M. Álvarez. Hydra-mdp: End-to-end multimodal planning with multi-target hydra-distillation. *arXiv preprint arXiv:2406.06978*, 2024.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Yunsheng Ma, Can Cui, Xu Cao, Wenqian Ye, Peiran Liu, Juanwu Lu, Amr Abdelraouf, Rohit Gupta, Kyungtae Han, Aniket Bera, et al. Lampilot: An open benchmark dataset for autonomous driving with language model programs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15141–15151, 2024.

Ming Nie, Renyuan Peng, Chunwei Wang, Xinyue Cai, Jianhua Han, Hang Xu, and Li Zhang. Reason2drive: Towards interpretable and chain-based reasoning for autonomous driving. In *European Conference on Computer Vision*, pp. 292–308. Springer, 2024.

OpenAI. Introducing openai o1. https://openai.com/o1/, 2024.

Katrin Renz, Long Chen, Elahe Arani, and Oleg Sinavski. Simlingo: Vision-only closed-loop autonomous driving with language-action alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11993–12003, 2025.

Hao Sha, Yao Mu, Yuxuan Jiang, Li Chen, Chenfeng Xu, Ping Luo, Shengbo Eben Li, Masayoshi Tomizuka, Wei Zhan, and Mingyu Ding. Languagempc: Large language models as decision makers for autonomous driving. *arXiv preprint arXiv:2310.03026*, 2023.

Hao Shao, Yuxuan Hu, Letian Wang, Guanglu Song, Steven L. Waslander, Yu Liu, and Hongsheng Li. Lmdrive: Closed-loop end-to-end driving with large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15120–15130, 2024.

Michal Shlapentokh-Rothman, Yu-Xiong Wang, and Derek Hoiem. Visual program distillation with template-based augmentation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, 2025.

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*, pp. 256–274. Springer, 2024.

Xurui Song, Shuo Huai, Jingjing Jiang, Jiayi Kong, and Jun Luo. More than meets the eye? uncovering the reasoning-planning disconnect in training vision-language driving models. *arXiv preprint arXiv:2510.04532*, 2025.

Jiacheng Tang, Zhiyuan Zhou, Zhuolin He, Jia Zhang, Kai Zhang, and Jian Pu. Causal-vad: De-confounding end-to-end autonomous driving via causal intervention. *arXiv preprint arXiv:2603.18561*, 2026.

Ximeng Tao, Pardis Taghavi, Dimitar Filev, Reza Langari, and Gaurav Pandey. Navidrivevlm: Decoupling high-level reasoning and motion planning for autonomous driving. *arXiv preprint arXiv:2603.07901*, 2026.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL `https://qwen.ai/blog?id=qwen3.5`.

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. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 4698–4726. PMLR, 2025.

Martin Treiber and Arne Kesting. Modeling lane-changing decisions with mobil. In Cécile Appert-Rolland, François Chevoir, Philippe Gondret, Sylvain Lassarre, Jean-Patrick Lebacque, and Michael Schreckenberg (eds.), *Traffic and Granular Flow '07*, pp. 211–221, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. ISBN 978-3-540-77074-9.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Hengli Wang, Peide Cai, Yuxiang Sun, Lujia Wang, and Ming Liu. Learning interpretable end-to-end vision-based motion planning for autonomous driving with optical flow distillation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 13731–13737. IEEE, 2021.

Tianqi Wang, Enze Xie, Ruihang Chu, Zhenguo Li, and Ping Luo. Drivecot: Integrating chain-of-thought reasoning with end-to-end driving. *arXiv preprint arXiv:2403.16996*, 2024.

Yan Wang, Wenjie Luo, Junjie Bai, Yulong Cao, Tong Che, Ke Chen, Yuxiao Chen, Jenna Diamond, Yifan Ding, Wenhao Ding, et al. Alpamayo-r1: Bridging reasoning and action prediction for generalizable autonomous driving in the long tail. *arXiv preprint arXiv:2511.00088*, 2025.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pp. 24824–24837, 2022.

Shuo Xing, Chengyuan Qian, Yuping Wang, Hongyuan Hua, Kexin Tian, Yang Zhou, and Zhengzhong Tu. Openemma: Open-source multimodal model for end-to-end autonomous driving. In *Proceedings of the Winter Conference on Applications of Computer Vision (WACV) Workshops*, pp. 1001–1009, 2025.

Zhenhua Xu, Yujia Zhang, Enze Xie, Zhen Zhao, Yong Guo, Kenneth K. Y. Wong, Zhenguo Li, and Hengshuang Zhao. Drivegpt4: Interpretable end-to-end autonomous driving via large language model. *arXiv preprint arXiv:2310.01412*, 2023.

Zhenjie Yang, Xiaosong Jia, Hongyang Li, and Junchi Yan. Llm4drive: A survey of large language models for autonomous driving. *arXiv preprint arXiv:2311.01043*, 2023.

Yuqi Ye, Zijian Zhang, Junhong Lin, Shangkun Sun, Changhao Peng, and Wei Gao. AutoDrive-P^3: Unified chain of perception–prediction–planning thought via reinforcement fine-tuning. In *International Conference on Learning Representations*, 2026.

Jianhao Yuan, Shuyang Sun, Daniel Omeiza, Bo Zhao, Paul Newman, Lars Kunze, and Matthew Gadd. Rag-driver: Generalisable driving explanations with retrieval-augmented in-context learning in multi-modal large language model. In *Robotics: Science and Systems*, 2024.

Wenyuan Zeng, Wenjie Luo, Simon Suo, Abbas Sadat, Bin Yang, Sergio Casas, and Raquel Urtasun. End-to-end interpretable neural motion planner. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8660–8669, 2019.

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. In *Advances in Neural Information Processing Systems*, 2025.

A Appendix

A.1 Trace Schema and Teacher-Specific Signals

Table A1 summarizes the planner-state fields retained before text serialization. The goal is not to expose implementation-specific debug variables, but to preserve the decision evidence that shaped the selected trajectory.

Table A1: Trace fields retained for reasoning serialization.

Teacher	Raw trace fields	Serialized slots
IDM Planner	route context, lead-vehicle state, headway, desired gap, relative speed, acceleration update	scene context, active longitudinal constraint, gap status, final speed decision
IDM MOBIL Planner	IDM state, adjacent-lane interaction, lane-change incentive, gap acceptance, nearby-agent response	scene context, interactive agents, lateral candidate, selected maneuver
PDM-Closed Planner	proposal bank, safety-gate outcomes, progress and comfort scores, selected proposal, fallback status	candidate summary, rejection reasons, score evidence, selected trajectory

A.2 Closed-Loop Scenario Index Mapping

Figure 5 uses compact indices to keep the 59-scenario closed-loop teacher comparison readable. Table A2 lists the shuffled index assignment, the scenario family, the selected teacher, and the corresponding best CCS value. Missing teacher entries in the source CSV are filled with deterministic low values in $[0, 0.25]$ for visualization only; the mapping table records the teacher selected after that visualization-time fill.

Table A2: Scenario-index mapping for the closed-loop teacher-quality heatmap. The selected teacher is computed after filling missing teacher entries with deterministic low-score random values for visualization only.

Index	Scenario family	Selected teacher	CCS
S01	low_magnitude_speed	IDM	0.378
S02	near_trafficcone_on_driveable	PDM-Closed	0.926
S03	accelerating_at_traffic_light_with_lead	IDM	0.451
S04	accelerating_at_traffic_light_without_lead	PDM-Closed	0.978
S05	following_lane_with_lead	IDM	0.173
S06	stationary_in_traffic	IDM	0.324
S07	following_lane_with_slow_lead	IDM MOBIL Planner	0.821
S08	on_stopline_traffic_light	PDM-Closed	0.781
S09	changing_lane_to_left	IDM	0.611
S10	changing_lane	IDM	0.498
S11	medium_magnitude_speed	IDM	0.537
S12	stopping_at_stop_sign_without_lead	IDM MOBIL Planner	0.850
S13	behind_pedestrian_on_driveable	IDM	0.313
S14	starting_right_turn	IDM	0.434
S15	traversing_crosswalk	IDM MOBIL Planner	0.904
S16	starting_protected_noncross_turn	IDM	0.478
S17	near_barrier_on_driveable	IDM MOBIL Planner	0.740
S18	accelerating_at_crosswalk	IDM	0.471
S19	following_lane_without_lead	IDM MOBIL Planner	0.969
S20	stationary_at_traffic_light_without_lead	PDM-Closed	0.990
S21	on_intersection	PDM-Closed	0.967
S22	near_pedestrian_on_crosswalk_with_ego	IDM	0.440
S23	high_magnitude_jerk	IDM	0.401

Index	Scenario family	Selected teacher	CCS
S24	high_lateral_acceleration	IDM	0.875
S25	starting_straight_stop_sign_intersection_traversal	IDM	0.555
S26	stationary_at_crosswalk	IDM	0.465
S27	starting_protected_cross_turn	IDM	0.556
S28	stopping_at_crosswalk	IDM	0.453
S29	on_traffic_light_intersection	IDM MOBIL Planner	0.861
S30	stopping_at_stop_sign_with_lead	IDM MOBIL Planner	0.277
S31	high_magnitude_speed	IDM	0.633
S32	starting_straight_traffic_light_intersection_traversal	IDM	0.483
S33	stationary	PDM-Closed	0.452
S34	behind_long_vehicle	PDM-Closed	0.986
S35	traversing_intersection	IDM	0.552
S36	traversing_narrow_lane	IDM	0.616
S37	traversing_traffic_light_intersection	IDM	0.910
S38	near_pedestrian_on_crosswalk	IDM	0.497
S39	on_stopline_crosswalk	IDM	0.453
S40	near_multiple_vehicles	IDM	0.591
S41	starting_high_speed_turn	IDM MOBIL Planner	0.770
S42	accelerating_at_stop_sign_no_crosswalk	IDM MOBIL Planner	0.570
S43	starting_left_turn	PDM-Closed	0.974
S44	starting_low_speed_turn	IDM MOBIL Planner	0.744
S45	stopping_at_traffic_light_without_lead	PDM-Closed	0.976
S46	crossed_by_vehicle	IDM MOBIL Planner	0.485
S47	near_high_speed_vehicle	IDM	0.616
S48	stopping_at_traffic_light_with_lead	IDM MOBIL Planner	0.666
S49	behind_bike	IDM	0.768
S50	stopping_with_lead	IDM MOBIL Planner	0.271
S51	starting_unprotected_noncross_turn	IDM	0.507
S52	near_long_vehicle	IDM MOBIL Planner	0.903
S53	accelerating_at_traffic_light	IDM MOBIL Planner	0.420
S54	stationary_at_traffic_light_with_lead	IDM	0.302
S55	changing_lane_to_right	PDM-Closed	0.965
S56	starting_unprotected_cross_turn	IDM	0.554
S57	on_stopline_stop_sign	PDM-Closed	0.925
S58	accelerating_at_stop_sign	IDM MOBIL Planner	0.787
S59	waiting_for_pedestrian_to_cross	IDM	0.526

A.3 Prompt Template

The model receives the same user-side driving prompt for all compared variants. Only the assistant target changes across planning-only, generic-reasoning, and planner-grounded-reasoning settings.

```
You are an autonomous driving agent.  Given multi-view
camera images, route context, and the recent ego-state
history, predict the ego trajectory for the next 5 seconds.
If reasoning is requested, explain the active driving
constraint, summarize the key candidate evaluations, and
then output future waypoints.
```

For planner-grounded variants, the assistant target is serialized as planner reasoning followed by waypoints:

```
<REASONING>...  </REASONING> <PLANNING> [x_1,y_1], ...,
[x_10,y_10] </PLANNING>.
```

A.4 Reasoning Data Comparison Examples

To clarify what changes across our training-data variants, we compare three supervision styles used for the trajectory-prediction task: *without reasoning*, *concise reasoning*, and *detailed reasoning*. All three variants share the same scene input and the same target trajectory; they differ only in how much intermediate decision information is exposed to the model.

Table A3: Assistant-response length for representative training examples. The detailed variant is roughly three times longer than the concise variant and about nine times longer than the no-reasoning baseline, motivating our concise-vs.-detailed comparison in the main paper.

Example	No reasoning	Concise	Detailed
Starting to cruise on a free road	443	1,183	3,605
Car-following in dense traffic	421	1,542	3,823
Red-light response	427	1,567	3,597
Stopped in a queue	423	1,678	4,042

Table A3 highlights an important practical difference between the two reasoning variants. Detailed chain-of-thought supervision provides richer descriptions, but it is substantially more expensive in tokens. Concise planner-grounded reasoning is therefore attractive because it preserves the action-relevant structure of the planner trace while remaining much cheaper to train and decode.

Example A: Free-road acceleration. **Sample id:** `1ae27458a23f5e93_74`. All three targets below use the same input prompt and the same final waypoint sequence.

```
Input prompt:
You are an autonomous driving agent. You have access to multi-view camera
images of a vehicle: (1) front view (which you should focus on with the
most
attention) <image>, (2) front right view <image>, and (3) front left view
<image>. Your task is to do your best to predict future waypoints for the
vehicle over the next 10 timesteps, given the vehicle's intent inferred
from
the images. Provided are the previous ego vehicle status. Positive x
means
forward direction while positive y means leftwards. The data is presented
in
the format [x, y]:(t-2.0s) [-0.03, 0.00], Acceleration: X -0.04, Y 0.02
m/s^2,
Velocity: X -0.01, Y 0.00 m/s, (t-1.5s) [-0.03, 0.00], Acceleration: X
-0.01,
Y -0.01 m/s^2, Velocity: X -0.00, Y 0.00 m/s, (t-1.0s) [-0.04, 0.00],
Acceleration: X -0.02, Y 0.01 m/s^2, Velocity: X -0.00, Y 0.00 m/s,
(t-0.5s) [-0.03, 0.00], Acceleration: X 0.46, Y -0.02 m/s^2, Velocity: X
0.04,
Y 0.00 m/s, (t-0.0s) [0.0, 0.0], Acceleration: X 1.40, Y -0.01 m/s^2,
Velocity: X 0.24, Y -0.00 m/s

No-reasoning answer:
<PLANNING>Predicted future movement details for the next 5 seconds
(sampled at
0.5-second intervals), including BEV location in x and y directions (in
meters). Positive x means forward direction while positive y means
leftwards.
The output is formatted as [x, y]: [16.14, -28.14], [16.20, -28.51],
[16.29, -29.12], [16.43, -29.98], [16.60, -31.09], [16.81, -32.44],
[17.06, -34.04], [17.36, -35.88], [17.69, -37.96], [18.05, -40.29]
</PLANNING>

Concise-reasoning answer:
<REASONING>
```

```
[Route] Following planned lane sequence. 35 lane segments, route found.
Target
speed: 10.0 m/s.
[Scene] 22 objects within detection range. Current speed: 0.2 m/s.
[t=0.5s] Free road ahead. Accelerate +1.00 m/s^2.
[t=1.0s] Free road ahead. Accelerate +1.00 m/s^2.
[t=1.5s] Free road ahead. Accelerate +1.00 m/s^2.
[t=2.0s] Free road ahead. Accelerate +1.00 m/s^2.
[t=2.5s] Free road ahead. Accelerate +1.00 m/s^2.
[t=3.0s] Free road ahead. Accelerate +0.99 m/s^2.
[t=3.5s] Free road ahead. Accelerate +0.99 m/s^2.
[t=4.0s] Free road ahead. Accelerate +0.98 m/s^2.
[t=4.5s] Free road ahead. Accelerate +0.97 m/s^2.
[t=5.0s] Free road ahead. Accelerate +0.95 m/s^2.
[Summary] Free road driving. Accelerated from 0.2 to 5.2 m/s.
</REASONING>
<PLANNING>Predicted future movement details for the next 5 seconds
(sampled at
0.5-second intervals), including BEV location in x and y directions (in
meters). Positive x means forward direction while positive y means
leftwards.
The output is formatted as [x, y]: [16.14, -28.14], [16.20, -28.51],
[16.29, -29.12], [16.43, -29.98], [16.60, -31.09], [16.81, -32.44],
[17.06, -34.04], [17.36, -35.88], [17.69, -37.96], [18.05, -40.29]
</PLANNING>

Detailed-reasoning answer:
<REASONING>
I need to plan my trajectory for the next 5 seconds. Let me analyze the
current
driving situation step by step.

**Perception & Route:**
My route consists of 35 lane segments and was successfully planned. The
target
speed for this road is 10.0 m/s.
I detect 22 objects in my surroundings. My current speed is 0.2 m/s.

**Step-by-step Planning:**

Step 1 (t=0.5s): No obstacles in my planned path ahead. The road is
clear.
My current speed (0.2 m/s) is below the target (10.0 m/s), so I should
accelerate to reach the desired speed.
Applying throttle: acceleration = +1.00 m/s^2. Speed after this step: 0.7
m/s.

Step 2 (t=1.0s): No obstacles in my planned path ahead. The road is
clear.
My current speed (0.7 m/s) is below the target (10.0 m/s), so I should
accelerate to reach the desired speed.
Applying throttle: acceleration = +1.00 m/s^2. Speed after this step: 1.2
m/s.

Step 3 (t=1.5s): No obstacles in my planned path ahead. The road is
clear.
My current speed (1.2 m/s) is below the target (10.0 m/s), so I should
accelerate to reach the desired speed.
Applying throttle: acceleration = +1.00 m/s^2. Speed after this step: 1.7
m/s.

Step 4 (t=2.0s): No obstacles in my planned path ahead. The road is
clear.
My current speed (1.7 m/s) is below the target (10.0 m/s), so I should
accelerate to reach the desired speed.
```

```
Applying throttle: acceleration = +1.00 m/s^2. Speed after this step: 2.2
m/s.

Step 5 (t=2.5s): No obstacles in my planned path ahead. The road is
clear.
My current speed (2.2 m/s) is below the target (10.0 m/s), so I should
accelerate to reach the desired speed.
Applying throttle: acceleration = +1.00 m/s^2. Speed after this step: 2.7
m/s.

Step 6 (t=3.0s): No obstacles in my planned path ahead. The road is
clear.
My current speed (2.7 m/s) is below the target (10.0 m/s), so I should
accelerate to reach the desired speed.
Applying throttle: acceleration = +0.99 m/s^2. Speed after this step: 3.2
m/s.

Step 7 (t=3.5s): No obstacles in my planned path ahead. The road is
clear.
My current speed (3.2 m/s) is below the target (10.0 m/s), so I should
accelerate to reach the desired speed.
Applying throttle: acceleration = +0.99 m/s^2. Speed after this step: 3.7
m/s.

Step 8 (t=4.0s): No obstacles in my planned path ahead. The road is
clear.
My current speed (3.7 m/s) is below the target (10.0 m/s), so I should
accelerate to reach the desired speed.
Applying throttle: acceleration = +0.98 m/s^2. Speed after this step: 4.2
m/s.

Step 9 (t=4.5s): No obstacles in my planned path ahead. The road is
clear.
My current speed (4.2 m/s) is below the target (10.0 m/s), so I should
accelerate to reach the desired speed.
Applying throttle: acceleration = +0.97 m/s^2. Speed after this step: 4.7
m/s.

Step 10 (t=5.0s): No obstacles in my planned path ahead. The road is
clear.
My current speed (4.7 m/s) is below the target (10.0 m/s), so I should
accelerate to reach the desired speed.
Applying throttle: acceleration = +0.95 m/s^2. Speed after this step: 5.2
m/s.

**Decision Summary:**
This is a free-road driving scenario with no immediate obstacles. I
accelerated
from 0.2 to 5.2 m/s over the planning horizon.
</REASONING>
<PLANNING>Predicted future movement details for the next 5 seconds
(sampled at
0.5-second intervals), including BEV location in x and y directions (in
meters). Positive x means forward direction while positive y means
leftwards.
The output is formatted as [x, y]: [16.14, -28.14], [16.20, -28.51],
[16.29, -29.12], [16.43, -29.98], [16.60, -31.09], [16.81, -32.44],
[17.06, -34.04], [17.36, -35.88], [17.69, -37.96], [18.05, -40.29]
</PLANNING>
```

Example B: Red-light stopping. **Sample id:** `115e26ae2da15108_74`. This example is more representative of the paper's main claim because the intermediate supervision explicitly names the active traffic constraint and shows how it shapes the resulting motion.

```
Input prompt:
You are an autonomous driving agent. You have access to multi-view camera
images of a vehicle: (1) front view (which you should focus on with the
most
attention) <image>, (2) front right view <image>, and (3) front left view
<image>. Your task is to do your best to predict future waypoints for the
vehicle over the next 10 timesteps, given the vehicle's intent inferred
from
the images. Provided are the previous ego vehicle status. Positive x
means
forward direction while positive y means leftwards. The data is presented
in
the format [x, y]:(t-2.0s) [-10.26, -0.15], Acceleration: X -1.15, Y
-1.13
m/s^2, Velocity: X 6.12, Y 0.11 m/s, (t-1.5s) [-7.39, -0.06],
Acceleration:
X -0.84, Y -0.54 m/s^2, Velocity: X 5.70, Y 0.04 m/s, (t-1.0s) [-4.71,
-0.02],
Acceleration: X -0.94, Y -0.31 m/s^2, Velocity: X 5.39, Y 0.10 m/s,
(t-0.5s) [-2.22, -0.01], Acceleration: X -1.23, Y -0.11 m/s^2, Velocity:
X 4.88, Y 0.09 m/s, (t-0.0s) [0.0, 0.0], Acceleration: X -1.05, Y -0.63
m/s^2,
Velocity: X 4.34, Y 0.09 m/s

No-reasoning answer:
<PLANNING>Predicted future movement details for the next 5 seconds
(sampled at
0.5-second intervals), including BEV location in x and y directions (in
meters). Positive x means forward direction while positive y means
leftwards.
The output is formatted as [x, y]: [2.17, -0.17], [4.27, -0.18], [6.18,
-0.18],
[7.78, -0.19], [8.98, -0.20], [9.76, -0.21], [10.20, -0.21], [10.42,
-0.21],
[10.52, -0.21], [10.57, -0.21]</PLANNING>

Concise-reasoning answer:
<REASONING>
[Route] Following planned lane sequence. 10 lane segments, route found.
Target
speed: 10.0 m/s.
[Scene] 37 objects within detection range. 1 red light(s) on route.
Current
speed: 4.3 m/s.
[t=0.5s] Red light at 11.6m. Gap tight (11.6m vs 12.9m needed).
Decelerate
-0.28 m/s^2.
[t=1.0s] Red light at 9.4m. Gap tight (9.4m vs 12.4m needed). Decelerate
-0.75 m/s^2.
[t=1.5s] Red light at 7.3m. Gap tight (7.3m vs 11.0m needed). Decelerate
-1.25 m/s^2.
[t=2.0s] Red light at 5.4m. Gap tight (5.4m vs 8.8m needed). Decelerate
-1.61 m/s^2.
[t=2.5s] Red light at 3.8m. Gap tight (3.8m vs 6.2m needed). Decelerate
-1.66 m/s^2.
[t=3.0s] Red light at 2.6m. Gap tight (2.6m vs 4.1m needed). Decelerate
-1.36 m/s^2.
[t=3.5s] Red light at 1.9m. Gap tight (1.9m vs 2.5m needed). Decelerate
-0.89 m/s^2.
[t=4.0s] Red light at 1.4m. Gap tight (1.4m vs 1.7m needed). Decelerate
-0.47 m/s^2.
[t=4.5s] Red light at 1.2m. Gap tight (1.2m vs 1.3m needed). Decelerate
-0.22 m/s^2.
[t=5.0s] Red light at 1.1m. Gap tight (1.1m vs 1.1m needed). Maintain
speed
```

```
(-0.10 m/s^2).
[Summary] Responding to red traffic light. Decelerated from 4.3 to 0.0
m/s.
</REASONING>
<PLANNING>Predicted future movement details for the next 5 seconds
(sampled at
0.5-second intervals), including BEV location in x and y directions (in
meters). Positive x means forward direction while positive y means
leftwards.
The output is formatted as [x, y]: [2.17, -0.17], [4.27, -0.18], [6.18,
-0.18],
[7.78, -0.19], [8.98, -0.20], [9.76, -0.21], [10.20, -0.21], [10.42,
-0.21],
[10.52, -0.21], [10.57, -0.21]</PLANNING>

Detailed-reasoning answer:
<REASONING>
I need to plan my trajectory for the next 5 seconds. Let me analyze the
current
driving situation step by step.

**Perception & Route:**
My route consists of 10 lane segments and was successfully planned. The
target
speed for this road is 10.0 m/s.
I detect 37 objects in my surroundings. Crucially, 1 red traffic light(s)
are
present on my route, which I must respect. My current speed is 4.3 m/s.

**Step-by-step Planning:**

Step 1 (t=0.5s): A red traffic light is 11.6m ahead. I must stop before
the
intersection.
The current gap (11.6m) is less than the safe stopping distance (12.9m).
I need
to brake.
Applying brake: deceleration = -0.28 m/s^2. Speed after this step: 4.2
m/s.

Step 2 (t=1.0s): A red traffic light is 9.4m ahead. I must stop before
the
intersection.
The current gap (9.4m) is less than the safe stopping distance (12.4m). I
need
to brake.
Applying brake: deceleration = -0.75 m/s^2. Speed after this step: 3.8
m/s.

Step 3 (t=1.5s): A red traffic light is 7.3m ahead. I must stop before
the
intersection.
The current gap (7.3m) is less than the safe stopping distance (11.0m). I
need
to brake.
Applying brake: deceleration = -1.25 m/s^2. Speed after this step: 3.2
m/s.

Step 4 (t=2.0s): A red traffic light is 5.4m ahead. I must stop before
the
intersection.
The current gap (5.4m) is less than the safe stopping distance (8.8m). I
need
to brake.
```

```
Applying brake: deceleration = -1.61 m/s^2. Speed after this step: 2.4
m/s.

Step 5 (t=2.5s): A red traffic light is 3.8m ahead. I must stop before
the
intersection.
The current gap (3.8m) is less than the safe stopping distance (6.2m). I
need
to brake.
Applying brake: deceleration = -1.66 m/s^2. Speed after this step: 1.6
m/s.

Step 6 (t=3.0s): A red traffic light is 2.6m ahead. I must stop before
the
intersection.
The current gap (2.6m) is less than the safe stopping distance (4.1m). I
need
to brake.
Applying brake: deceleration = -1.36 m/s^2. Speed after this step: 0.9
m/s.

Step 7 (t=3.5s): A red traffic light is 1.9m ahead. I must stop before
the
intersection.
The current gap (1.9m) is less than the safe stopping distance (2.5m). I
need
to brake.
Applying brake: deceleration = -0.89 m/s^2. Speed after this step: 0.4
m/s.

Step 8 (t=4.0s): A red traffic light is 1.4m ahead. I must stop before
the
intersection.
The current gap (1.4m) is less than the safe stopping distance (1.7m). I
need
to brake.
Applying brake: deceleration = -0.47 m/s^2. Speed after this step: 0.2
m/s.

Step 9 (t=4.5s): A red traffic light is 1.2m ahead. I must stop before
the
intersection.
The current gap (1.2m) is less than the safe stopping distance (1.3m). I
need
to brake.
Applying brake: deceleration = -0.22 m/s^2. Speed after this step: 0.1
m/s.

Step 10 (t=5.0s): A red traffic light is 1.1m ahead. I must stop before
the
intersection.
The current gap (1.1m) is less than the safe stopping distance (1.1m). I
need
to brake.
I am already stopped, waiting for the light to change.
Holding steady: acceleration = -0.10 m/s^2. Speed after this step: 0.0
m/s.

**Decision Summary:**
This is a red traffic light requiring me to stop. I decelerated from 4.3
to
0.0 m/s to maintain safety.
</REASONING>
<PLANNING>Predicted future movement details for the next 5 seconds
(sampled at
```

```
0.5-second intervals), including BEV location in x and y directions (in
meters). Positive x means forward direction while positive y means
leftwards.
The output is formatted as [x, y]: [2.17, -0.17], [4.27, -0.18], [6.18,
-0.18],
[7.78, -0.19], [8.98, -0.20], [9.76, -0.21], [10.20, -0.21], [10.42,
-0.21],
[10.52, -0.21], [10.57, -0.21]</PLANNING>
```

These full examples show exactly what changes across the three supervision variants. The scene input is identical, the final waypoint target is identical, and only the amount of exposed intermediate decision information changes. This is the contrast we use throughout the paper when analyzing whether richer planner-grounded reasoning improves planning quality.

A.5 Planner Variables and Motion Summaries

Auxiliary motion labels are derived from the planner output trajectory rather than from hand-written annotations. We use three label families for analysis: longitudinal mode (stop / slow / maintain / accelerate), lateral mode (lane follow / offset / lane change), and interaction outcome (yield / proceed / fallback). These labels are not the main target metric; the main experiments evaluate trajectory quality with ADE, FDE, heading error, and miss rate.

A.6 Additional Ablations and Compute Budget

The default training setup uses Qwen3.5-4B as the driving backbone, bf16 precision, full-parameter supervised fine-tuning, and an effective batch size of 16 on 8 GPUs. All reported variants use the same data split, preprocessing pipeline, optimization recipe, and trajectory decoder unless explicitly stated otherwise. The anonymized supplementary material contains the experiment configuration templates, data-construction entry points, evaluation scripts, and table-generation protocol needed to reproduce the reported comparisons. Per-run wall-clock time depends on the local simulator and storage backend, but the reported experiments are designed to be reproduced from the released configuration files without changing model architecture, data split, or metric definitions.

A.7 Asset Provenance, Release Scope, and Safeguards

Our experiments build on public driving benchmarks and simulator-generated planner traces derived from them. nuPlan and nuScenes provide the underlying scenario corpora and benchmark interfaces (Caesar et al., 2021; 2020), and the VLM backbone follows the public Qwen model family. The anonymized supplementary material documents the preprocessing pipeline, prompt templates, trace serialization schema, evaluation scripts, and reproduction instructions; any released derived assets are intended for research use under the terms of the underlying datasets and models. We do not claim that planner-grounded reasoning alone makes a driving system deployable. Dataset artifacts, prompts, and checkpoints should be used with simulator-only caveats, license notices for the underlying assets, and a clear statement that generated rationales are intended for analysis and research rather than direct safety guarantees.