ACTIONGUIDE: LEARNING TRANSFERABLE ACTION-GROUNDED REASONING FOR CLOSED-LOOP DRIVING

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce ActionGuide-Bench, a unified evaluation framework for connecting action-grounded reasoning in Multimodal Large Language Models (MLLMs) to closed-loop driving utility. Moving beyond isolated open-loop judgments, ActionGuide-Bench evaluates leading MLLMs at two complementary levels: an action-grounded evaluation comprising Evidence Grounding, Consequence Prediction, Trajectory Verification, and Corrective Guidance, and a closed-loop evaluation measuring how these capabilities influence planner decisions during execution. ActionGuide-Bench casts the MLLM as an online guide to a frozen base planner, verifying proposed trajectories and providing corrective guidance when the available behaviors are unsuitable, thereby exploiting the utility of MLLM judgments without retraining the underlying motion policy. To strengthen these capabilities, we further curate ActionGuide-Train, which connects decision-critical contexts with alternative ego behaviors, execution outcomes, and corrective actions through rule-based synthesis and planner rollouts. Comprehensive evaluation reveals three findings: (1) **Effective Guidance from MLLMs:** pretrained open-source MLLMs can already improve closed-loop driving as online guides, despite limited action-grounded reasoning; (2) **Transferable Behavior Knowledge:** behavior-level driving knowledge learned from simulation transfers effectively to real-world scenes; and (3) **Capability–Utility Alignment:** stronger and more balanced action-grounded competence is closely associated with better closed-loop utility. Together, these results position action-grounded reasoning as a key interface between multimodal understanding and concrete action control, translating visual-semantic knowledge into effective closed-loop behavior.

1 INTRODUCTION

Multimodal large language models (MLLMs) (Qwen Team, 2026) possess rich visual and semantic knowledge, offering considerable potential for autonomous driving. Recent studies have leveraged these capabilities for complex traffic-scene interpretation (Sima et al., 2024; Tian et al., 2025), while driving benchmarks systematically evaluate visual grounding and scene understanding through question answering (Xie et al., 2025). However, real-world driving requires more than understanding: an agent must reason about whether a proposed motion is appropriate, what consequences it may induce, and how the vehicle should adjust when it is unsuitable. This raise the fundemental question whether strong scene understanding translates into effective guidance for concrete *ego-vehicle actions*.

Recent work advances behavior-grounded reasoning through trajectory evaluation (Song et al., 2025) and counterfactual reasoning (Wang et al., 2025a), yet evaluates MLLMs primarily at isolated open-loop decision points (Dauner et al., 2024; Sun et al., 2026). Closed-loop benchmarks such as Bench2Drive (Jia et al., 2024) evaluate interactive driving policies, while MLLM-derived reward optimization (Chen et al., 2026) measures the resulting policy rather than the online utility of MLLM judgments themselves. It therefore remains unclear whether these judgments can effectively guide a base planner during closed-loop execution (NVIDIA et al., 2025a), where actions alter subsequent observations and errors accumulate. This motivates our central question: *How can we **evaluate** and **harness** MLLMs as driving guides to improve closed-loop driving?*

To investigate this capability-to-utility gap, we introduce **ActionGuide-Bench**, a unified benchmark connecting action-grounded reasoning in MLLMs with closed-loop driving utility. As illustrated

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

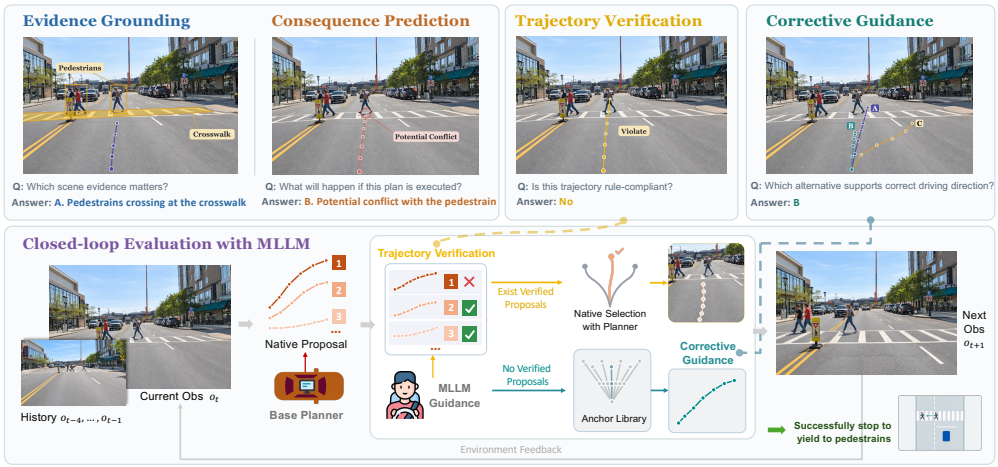

Figure 1: Overview of **ActionGuide-Bench**. The evaluation framework decomposes the driving system into a base planner and an MLLM guide and evaluates their interaction at two complementary levels. **Top**: *Action-grounded evaluation* measures four capabilities, connecting scene evidence and anticipated consequences to trajectory verification and corrective guidance. **Bottom**: *Closed-loop evaluation* operationalizes Trajectory Verification and Corrective Guidance within the planner.

in Figure 1, an MLLM acts as an online guide to a base planner, evaluating proposed trajectories and providing corrective guidance when necessary. ActionGuide-Bench evaluates this role at two complementary levels: an **action-grounded evaluation** spanning Evidence Grounding, Consequence Prediction, Trajectory Verification, and Corrective Guidance, and a **closed-loop evaluation** that measures whether MLLM guidance improves planner decisions during execution. Our results show effectiveness of MLLM as the guidance to improve closed-loop driving without policy retraining, yet their action-grounded reasoning remain limited reliability, with the best open-source model reaching only 41% overall accuracy.

To strengthen action-grounded reasoning and study its closed-loop utility, we curate ActionGuide-Train, which connects decision-critical contexts with alternative ego behaviors, execution outcomes, and corrective actions. Built in CARLA (Dosovitskiy et al., 2017), it combines controlled compliant–violating contrasts from rule-based synthesis with planner-generated behaviors and their outcomes collected from closed-loop rollouts, paired with simulator-verified labels and grounded rationales. Fine-tuning on the curated dataset yields ACTIONGUIDE, improving overall action-grounded accuracy by 12.04% over its pretrained counterpart and raising the base planner's closed-loop at-fault-collision scene score by 5.84.

Comprehensive analysis of leading MLLMs on ActionGuide-Bench yields three key insights:

- **Explicit reasoning better unlocks pretrained driving knowledge.** Pretrained MLLMs provide substantially better closed-loop guidance when their visual-semantic knowledge is explicitly organized around decision-relevant evidence and ego behavior.
- **Behavior-level driving knowledge transfers across sim-to-real.** For MLLMs, simulation-based supervision transfers effectively to real-world scenarios requiring semantic and behavioral reasoning, showing that behavior-level driving knowledge learned in simulation can generalize across domains.
- **Broader action-grounded competence aligns with closed-loop utility.** Models with stronger action-grounded capabilities tend to achieve better closed-loop performance, suggesting that effective driving guidance benefits from jointly grounding scene evidence, action consequences, and behavioral decisions.

In summary, we present a unified framework for evaluating and training MLLMs as closed-loop driving guides. By linking action-grounded reasoning to execution outcomes, our work provides both a diagnostic benchmark and a practical pathway for improving closed-loop driving systems,

establishing a foundation for multimodal models that translate visual-semantic knowledge into reliable decisions in interactive physical environments.

2 BENCHMARKING MLLM FOR DRIVING GUIDANCE

ActionGuide-Bench evaluates MLLMs as driving guides through two complementary components: *action-grounded evaluation* assesses their ability to identify relevant scene evidence and reason about proposed driving behaviors, while *closed-loop evaluation* measures whether their judgments improve driving performance. We first formulate the closed-loop system comprising a base planner and an MLLM (Section 2.1), then define the action-grounded tasks that assess the MLLM guidance capabilities (Section 2.2). Finally, we define the guidance interface that integrates MLLM judgments into the base planner's decision process for closed-loop evaluation. (Section 2.3).

2.1 PRELIMINARY: CLOSED-LOOP DRIVING FORMULATION

We consider a closed-loop driving system $\Pi_{\phi,\theta}$ comprising a base planner with parameters ϕ and an MLLM with parameters θ. At each replanning step t, the system receives driving context

$$H_t = (\mathcal{V}_t, e_t, r_t),$$

where $\mathcal{V}_t$ denotes the visual observation context, comprising recent front-view history together with current multi-view camera observations, e_t is the current ego state, and r_t denotes route information. Here, t indexes replanning steps. The system predicts a trajectory

$$\tau_t = \Pi_{\phi,\theta}(H_t) = \big(\mathbf{p}_{t,1}, \ldots, \mathbf{p}_{t,N}\big), \tag{1}$$

where $\mathbf{p}_{t,\ell} \in \mathbb{R}^2$ denotes the predicted ego position at future time offset $\ell\delta$, expressed in the ego coordinate frame at replanning step t. Here, N is the number of predicted waypoints and δ_{wp} is the temporal interval between consecutive waypoints, yielding a prediction horizon of $N\delta$.

The system operates in a receding-horizon manner: the controller tracks an interpolated prefix of the predicted trajectory over an execution interval Δ_{exec}, which may be shorter than the waypoint sampling interval δ_{wp}. After executing this prefix, the system acquires updated visual observations, ego state, and route information, forming the next context

$$H_{t+1} = (V_{t+1}, e_{t+1}, r_{t+1}), \qquad \tau_{t+1} = \Pi_{\phi,\theta}(H_{t+1}). \tag{2}$$

This feedback loop makes subsequent predictions depend on observations obtained after executing earlier decisions. We define the action-grounded tasks for evaluating MLLM capabilities in Section 2.2 and specify the guidance interface between the MLLM and base planner in Section 2.3.

2.2 ACTION-GROUNDED MLLM EVALUATION

Our action-grounded evaluation comprises four tasks: *Evidence Grounding*, *Consequence Prediction*, *Trajectory Verification*, and *Corrective Guidance*. The first two assess supporting capabilities for reasoning about a proposed ego behavior, while the latter two directly instantiate driving guidance. All tasks share the visual input $\mathcal{V}$ defined in Section 2.1, consisting of four historical front-view frames together with the current front, left, and right views. We omit the time index t in the task definitions below for brevity. All tasks are formulated as multiple-choice question answering with exactly one correct answer.

As shown in Figure 1, the first three tasks probe complementary capabilities for trajectory-conditioned reasoning. Given $\mathcal{V}$ and a proposed trajectory τ rendered as an overlay, **Evidence Grounding** identifies the scene elements and constraints relevant to assessing τ, such as a crossing pedestrian or a red traffic light. **Consequence Prediction** anticipates the likely outcome of executing τ, such as safe progress, a collision, or a lane departure. Building on these capabilities, **Trajectory Verification** produces a binary judgment $y_{\text{TV}} \in \{0, 1\}$, where 1 indicates that τ is acceptable in the current context. This decomposition evaluates whether the MLLM can connect decision-relevant evidence and anticipated consequences to the acceptability of a concrete ego action before execution.

Corrective Guidance addresses cases in which verification rejects all proposals from the base planner. In a closed-loop setting, even a local correction can alter the next observation and influence

subsequent planning decisions, potentially steering the system toward states where the base planner again produces acceptable proposals. Given $\mathcal{V}$ and an auxiliary anchor library $\mathcal{L}$, the model selects an anchor $y_{\mathrm{CG}} \in \mathcal{L}$ that provides a local corrective direction. The library consists of a fixed set of directional motion templates spanning hard left, light left, straight, light right, hard right, and stop. This task therefore probes whether action-grounded reasoning can support close-loop recovery.

2.3 MLLM–Planner Integration in Close-loop Evaluation

Within the closed-loop framework defined in Section 2.1, we first describe the base planner's inference process from trajectory proposal to selection. We then define how the MLLM provides action-grounded guidance through the *Trajectory Verification* and *Corrective Guidance* tasks.

Base Planning. We describe the base planner through two functional interfaces: a trajectory proposer and a trajectory selector. Given the driving context H_t, the trajectory proposer G_ϕ generates a set of candidate trajectories:

$$\mathcal{C}_t = G_\phi(H_t) = \{\tau_t^{(k)}\}_{k=1}^{K}, \qquad K \geq 1, \tag{3}$$

where K depends on the base planner and its proposal configuration.

The trajectory selector S takes H_t and a nonempty candidate set $\mathcal{C}$, and returns one trajectory according to the planner's native selection criterion, with $S(H_t, \mathcal{C}) \in \mathcal{C}$. The input set may be the original proposal set $\mathcal{C}_t$ or a filtered subset. This functional interface accommodates a dedicated selection module or an existing ranking rule. For a single-output planner, $K = 1$ and selection reduces to returning the sole candidate, i.e., $S(H_t, \{\tau\}) = \tau$. Without MLLM guidance, the selector operates on the original proposal set, yielding $\tau_t^{\mathrm{base}} = S(H_t, \mathcal{C}_t)$.

MLLM-guided Planning. Building on the proposal-and-selection interface, we incorporate two MLLM tasks from Section 2.2, as illustrated in the lower panel of Figure 1: *Trajectory Verification* evaluates native proposals before selection, while *Corrective Guidance* provides an alternative when none is accepted.

After G_ϕ generates $\mathcal{C}_t$, the MLLM performs **trajectory verification** on each candidate $\tau_t^{(k)}$, given $\mathcal{V}_t$ and the corresponding trajectory overlay. The resulting binary judgments $y_{\mathrm{TV},t}^{(k)} \in \{0, 1\}$ define the accepted subset

$$\mathcal{A}_t = \left\{ \tau_t^{(k)} \in \mathcal{C}_t \;\middle|\; y_{\mathrm{TV},t}^{(k)} = 1 \right\}. \tag{4}$$

When $\mathcal{A}_t \neq \varnothing$, the system follows the upper branch of the closed-loop diagram: the native selector S chooses from $\mathcal{A}_t$ rather than the full proposal set $\mathcal{C}_t$. When $\mathcal{A}_t = \varnothing$, the system instead follows the lower branch, invoking **corrective guidance** rather than selecting from the rejected proposals. Given $\mathcal{V}_t$ and the auxiliary anchor library $\mathcal{L}$, the MLLM selects $y_{\mathrm{CG},t} \in \mathcal{L}$ to specify a local corrective direction, which is realized under the current driving context as $\tau_t^{\mathrm{CG}} = R(H_t, y_{\mathrm{CG},t})$. This provides a corrective motion outside the planner's native candidate set. The combined planning decision is

$$\tau_t = \Pi_{\phi,\theta}(H_t) = \begin{cases} S(H_t, \mathcal{A}_t), & \mathcal{A}_t \neq \varnothing, \\ \tau_t^{\mathrm{CG}}, & \mathcal{A}_t = \varnothing. \end{cases} \tag{5}$$

Both branches return a trajectory for prefix execution under Section 2.1. The resulting environment feedback updates H_{t+1}, from which the base proposer generates new candidates and the same procedure repeats. Corrective guidance therefore changes the local motion while leaving subsequent planning to the base planner.

3 Learning Action-Grounded Driving Guidance

To train MLLMs for action-grounded driving reasoning, we construct ActionGuide-Train around the four tasks introduced in Section 2.2. All tasks are formulated as multiple-choice question answering with a single correct option. As illustrated in Figure 2, we construct training data in CARLA and use it for supervised fine-tuning of ActionGuide.

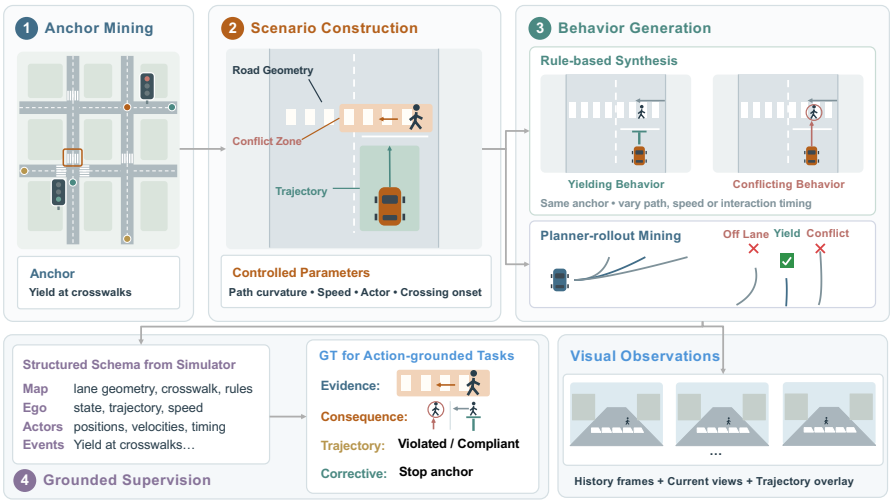

Figure 2: **Overview of our Simulation-based Data Engine.** We mine map anchors, construct controlled scenarios, and generate driving behaviors through rule-based synthesis and planner-rollout mining. Structured simulator states provide supervision for the four action-grounded tasks, paired with historical and current visual observations and trajectory overlays.

Scenario and Observation. We construct rule-grounded driving scenarios in CARLA based on traffic rules and interaction patterns covered in the California Driver's Handbook (California DMV, 2025). Each scenario targets a specific decision challenge, such as yielding to a crossing pedestrian. We instantiate the ego vehicle, surrounding traffic participants and their scripted behaviors, and the road context, and sample decision states near the point where the corresponding rule or interaction becomes consequential. For each sampled state, we record four historical front-view frames together with the current front, left, and right views, forming the visual input $\mathcal{V}$ defined in Section 2.2.

Behavior Generation. Given a shared scene context, we construct a set of candidate trajectories $\mathcal{T} = \{\tau^{(k)}\}_{k=1}^{K}$ from two complementary sources. **Rule-based synthesis** generates controlled variations of ego motion by modifying longitudinal progress, stopping behavior, lateral displacement, or interaction timing, producing both compliant and conflicting candidates under the same observation. These generators use only the available scene context and map information, without accessing future outcomes or task labels. In parallel, **planner-rollout mining** collects trajectories proposed by existing base planners together with challenging states encountered during planning, yielding candidates that better reflect their closed-loop behavior distribution, with trajectory labels determined from simulator state and rollout outcomes.

Task Supervision. We replay the generated trajectories in simulation and verify their behavior using structured simulator information, including map geometry, traffic signals, ego state, and surrounding actors. Each validated scene–trajectory pair $(\mathcal{V}, \tau)$ is then converted into supervision for the four tasks. The resulting task instances are converted into single-answer multiple-choice examples. Based on the curated dataset $\mathcal{D}$, we supervised fine-tune the MLLM using the standard autoregressive objective:

$$\mathcal{L}_{\mathrm{SFT}}(\theta) = -\sum_{(x_i, y_i)\in\mathcal{D}} \log p_\theta(y_i \mid x_i), \tag{6}$$

where x_i denotes the task-specific input and y_i the corresponding ground-truth answer. Additional details on task supervision, trajectory generation, validation rules, and more are provided in Section B.

4 Experiments

In this section, we evaluate MLLMs on four action-grounded reasoning capabilities and examine whether these capabilities generalize to real-world driving judgments and translate into closed-loop utility in AlpaSim.

Table 1: Action-grounded evaluation on the four tasks of ActionGuide-Bench. **Bold** indicates the best performance and underline the second-best. Yes Ratio is diagnostic and has no preferred direction. Overall accuracy is the unweighted mean of the accuracies on the four tasks.

Model	Evidence	Consequence	Trajectory		Corrective	Overall
	Acc.	Acc.	Acc.	Yes Ratio (%)	Acc.	Acc.
Driving VLAs						
Alpamayo-1.5-10B	27.41	31.20	50.13	99.87	14.70	30.86
ReCogDrive	25.43	37.76	50.80	2.12	16.95	32.74
Qwen-Drive-1.0	30.62	45.12	**69.23**	52.65	21.19	41.54
Open-source MLLMs						
Qwen2.5-VL-7B	25.68	29.60	50.00	0.00	15.63	30.23
DriveFusion QA	36.54	16.96	57.03	30.64	19.21	32.43
Qwen3-VL-8B	26.67	42.72	50.80	0.80	14.57	33.69
Qwen3-VL-32B	41.73	52.32	51.86	2.39	17.48	40.85
Qwen3.5-4B	33.09	42.40	64.46	43.10	18.54	39.62
InternVL3.5-8B	27.65	37.92	54.77	11.67	19.34	34.92
Cosmos-Reason2-8B	44.94	42.08	60.34	68.17	18.28	41.41
ActionGuide (ours)	**54.57**	**65.76**	61.01	54.77	**32.45**	**53.45**

4.1 Experimental Setup

Evaluation Dataset and Metrics. For **action-grounded evaluation**, we construct the evaluation set from the validation splits of nuScenes (Caesar et al., 2020), Waymo Open Motion Dataset (Ettinger et al., 2021), and Argoverse 2 (Wilson et al., 2023), and report accuracy on all four action-grounded tasks. For **closed-loop evaluation**, we evaluate the complete system in AlpaSim (NVIDIA et al., 2025a) on 131 curated decision-critical scenarios emphasizing traffic-rule reasoning, temporal context, and interactions with other road users. We report Scene Score as the primary metric. Additional scenario and evaluation details are provided in Section A.

MLLM Baselines. In the action-grounded evaluation (Table 1), our baselines span leading general-purpose MLLMs, driving-informed MLLMs, and driving-specialized VLAs. We include general-purpose MLLMs such as Qwen2.5-VL-7B (Bai et al., 2025b), Qwen3-VL-8B (Bai et al., 2025a) and Qwen3-VL-32B (Bai et al., 2025a), Qwen3.5-4B (Qwen Team, 2026), InternVL3.5-8B (Wang et al., 2025b), and Cosmos-Reason2-8B (NVIDIA, 2026). We also evaluate DriveFusion QA (Samir & Team, 2026), an MLLM incorporating driving-specific knowledge. Finally, we compare against driving-specialized vision-language-action models (VLAs), including Alpamayo-1.5-10B (NVIDIA et al., 2025b), ReCogDrive (Li et al., 2026), and Qwen-Drive-1.0 (Zhou et al., 2026).

Driving Planners. For the controlled closed-loop evaluation of MLLM guidance in Table 2, we use DrivoR (Kirby et al., 2026) as the primary base planner. DrivoR produces multiple candidate trajectories at each planning step, providing a shared proposal set on which different MLLMs can be compared under the same guidance interface. We additionally evaluate a diverse set of state-of-the-art driving planners in Table 5, including Alpamayo-1.5 (NVIDIA et al., 2025b), AutoVLA (Zhou et al., 2025), ReCogDrive (Li et al., 2026), DrivoR (Kirby et al., 2026), MindDrive (Fu et al., 2026), and Qwen-Drive-1.0 (Zhou et al., 2026).

Implementation Details. We generate training data in CARLA 0.9.16 (Dosovitskiy et al., 2017). Each sample contains four historical front-view frames over the preceding 2 s and the current front, front-left, and front-right views. Candidate trajectories consist of eight waypoints at 0.5 s intervals over a 4 s horizon and are generated from the current state, observation history, and map context. We fine-tune Cosmos-Reason2-8B with supervised fine-tuning, updating the LLM backbone while freezing the vision encoder and aligner. Training uses bfloat16 on 8 GPUs for one epoch with AdamW, a learning rate of 1×10^{-5}, 0.05 warmup, weight decay of 0.1, and an effective batch size of 16.

Table 2: Closed-loop performance of MLLM guides in AlpaSim, with DrivoR as the base planner.

Method	Collision all (%)↓	Collision at-fault (%)↓	Off-road (%)↓	Scene Score all↑	Scene Score at-fault↑
DrivoR	41.98	38.17	22.90	42.56	43.66
+ Qwen2.5-VL-7B	38.17	36.64	22.90	44.72	45.38
+ DriveFusion QA	38.93	38.17	23.66	44.57	44.57
+ Qwen3-VL-8B	37.40	36.64	25.19	45.26	45.26
+ Qwen3.5-4B	37.40	36.64	26.72	45.27	45.27
+ InternVL3.5-8B	37.40	36.64	24.43	44.86	44.86
+ Qwen3-VL-32B	**36.64**	35.88	24.43	45.45	45.45
+ Cosmos-Reason2-8B	39.69	35.88	21.37	45.36	47.92
+ **ActionGuide (ours)**	37.40	**35.11**	**20.61**	**48.18**	**49.50**

4.2 Action-Grounded Evaluation Results

Table 1 shows that ActionGuide achieves the strongest overall action-grounded performance among all evaluated open-source models, reaching 53.45% overall accuracy. Compared with its pretrained Cosmos-Reason2-8B counterpart, it improves Evidence Grounding by 9.63%, Consequence Prediction by 23.68% and Corrective Guidance by 14.22%, demonstrating the effectiveness of our curated action-grounded supervision. Since training uses simulation while evaluation is conducted on real-world driving data, these gains also indicate that the learned behavior-level knowledge transfers across domains. Existing open-source models, however, remain limited in their action-grounded capabilities, with the strongest baseline reaching only about 41% overall accuracy. Their Trajectory Verification behavior is particularly uneven: Yes Ratios range from near 0% to nearly 100%, showing that similar classification accuracy can correspond to very different intervention tendencies. Among driving-specialized VLAs, Qwen-Drive-1.0 achieves the strongest Trajectory Verification accuracy 69.23%, yet its closed-loop performance in Table 5 remains relatively limited. This gap suggests that unified language–action modeling alone does not guarantee consistent action-grounded reasoning and closed-loop execution, motivating explicit integration of MLLM guidance into the driving system.

4.3 Closed-Loop Driving Results

We next evaluate whether action-grounded MLLM capabilities translate into closed-loop driving utility. We keep DrivoR as the base planner and vary only the MLLM guide, evaluating all models under the same Trajectory Verification and Corrective Guidance interface.

As shown in Table 2, pretrained MLLMs provide measurable benefits over the frozen planner. In particular, Cosmos-Reason2-8B increases the at-fault-collision scene score from 43.66 to 47.92, while reducing both collision-all and at-fault collision rates by 2.29%. In the right panel of Figure 3, a per-scene analysis further shows that it improves 18 of 131 scenarios, with 13 of these improvements eliminating at least one collision or off-road outcome. The gains are concentrated in challenging semantic and interaction-heavy scenarios, including signal transitions and unprotected left turns.

ActionGuide further improves closed-loop performance, reaching 48.18 Scene Score (all) and 49.50 at-fault-collision scene score, while reducing at-fault collision and off-road rates to 35.11% and 20.61%, respectively. Relative to the frozen DrivoR planner, this corresponds to a +5.84 improvement in at-fault-collision scene score. These results show that strengthening action-grounded reasoning provides additional closed-loop utility beyond pretrained multimodal knowledge. For broader context, we report same-protocol closed-loop performance of additional state-of-the-art standalone planners in Table 5.

5 Analysis

Beyond aggregate performance, ActionGuide-Bench enables an in-depth analysis of how and where the action-grounded capabilities of MLLMs translate into closed-loop driving utility.

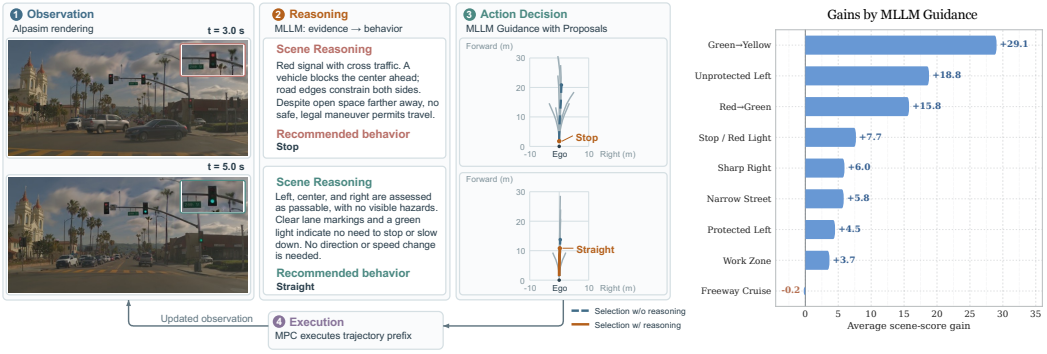

Figure 3: **Left:** Example of explicit scene reasoning for closed-loop action selection, where the MLLM maps visual evidence to a recommended behavior and planner proposal. **Right:** Average Scene Score gains from Cosmos-Reason2-8B guidance over DrivoR across scenario categories.

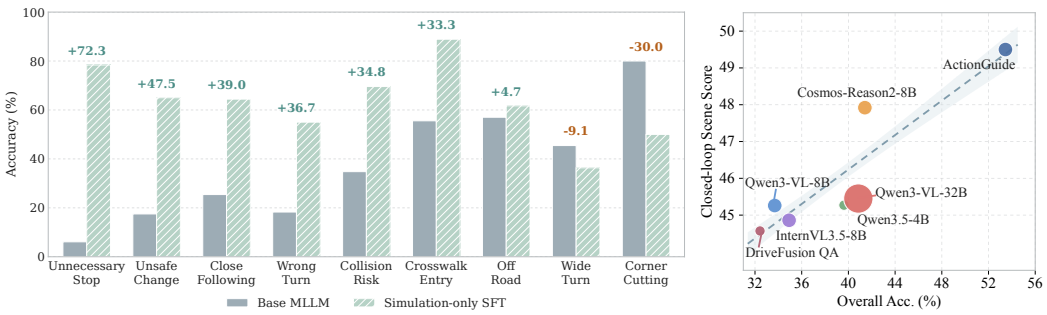

Figure 4: **Left:** Sim-to-real transfer from simulation-only action-grounded supervision across representative driving behaviors. **Right:** Correlation between action-grounded evaluation performance and closed-loop at-fault-collision scene score across MLLMs.

Explicit Reasoning Better Elicits Pretrained Driving Knowledge. We evaluate whether explicitly eliciting reasoning improves the closed-loop utility of a pretrained MLLM. As illustrated in the left panel of Figure 3, we prompt the MLLM first reasons about traffic controls and traversability, then converts this understanding into a recommended driving behavior from diverse planner proposals. Across 28 traffic-light and yielding scenarios, the AlpaSim at-fault Scene Score improves from 19.33 with DrivoR alone, to 20.93 with direct proposal-wise verification, and further to 31.24 with explicit reasoning. The result suggests that the pretrained knowledge from MLLMs becomes more actionable when explicitly connected to behavioral reasoning and action decision. This finding further motivates training supervision that couples scene evidence, behavior reasoning, and concrete driving decisions.

Behavior-Level Knowledge Transfers across Sim-to-Real. From Table 1, although all training data are collected in simulation, the learned knowledge transfers well to real-world scenes, improving overall performance from 41.41 to 53.45. A finer-grained analysis in the left panel of Figure 4 further reveals that the gains are especially large on scenarios requiring high-level driving knowledge and semantic behavioral reasoning, such as unnecessary stopping, collision risk, and crosswalk interaction. In contrast, geometry-sensitive cases such as wide turns and corner cutting show degration, which depends more on precise scene geometry and camera parameters that are harder to transfer because of visual domain gap. This suggests that simulation is particularly effective for MLLM to learn abstract, behavior-level driving knowledge, consistent with prior findings that simulated experience can improve real-world generalization and robustness in driving systems (Tian et al., 2026).

Broader Action-Grounded Competence Aligns with Closed-Loop Utility. We examine whether action-grounded capabilities measured by Table 1 are associated with downstream closed-loop performance. As shown in the right panel of Figure 4, overall accuracy of action-grounded evaluation is strongly correlated with closed-loop at-fault-collision Scene Score across seven MLLMs (Pearson

$r = 0.901$, Spearman $\rho = 0.964$). Even Evidence Grounding alone shows a strong positive association ($r = 0.858$, $\rho = 0.750$), suggesting that identifying decision-relevant evidence is indicative of downstream driving utility. The stronger correlation of overall performance suggests that closed-loop guidance is better reflected by comprehensive action-grounded competence across all four tasks.

6 Related Work

Vision–language reasoning and action generation for driving. Recent driving MLLMs increasingly connect semantic reasoning to action. DriveVLM combines scene understanding with hierarchical planning, while Senna separates language-level decisions from precise trajectory prediction and Senna-2 further aligns these decisions with an end-to-end driving policy (Tian et al., 2025; Jiang et al., 2024; Song et al., 2026). SimLingo explicitly couples vision–language understanding with closed-loop driving through language–action alignment (Renz et al., 2025), and ReCogDrive transfers learned driving cognition from an MLLM to a diffusion-based motion planner (Li et al., 2026). These works show that semantic reasoning can be aligned with driving actions. ActionGuide instead asks how much additional closed-loop value can be attributed to semantic judgment when motion generation remains with a separately trained planner.

Trajectory evaluation, reward modeling, and planner guidance. A parallel line of work learns to evaluate proposed driving behavior or uses semantic feedback to influence planning. DrivoR separates trajectory generation from learned candidate scoring (Kirby et al., 2026). DriveCritic learns context-aware MLLM-based trajectory preferences (Song et al., 2025); DriveReward trains a vision–language reward model from counterfactual trajectory supervision and applies it to trajectory scoring and policy optimization (Chen et al., 2026); and DriveJudge combines contextual MLLM reasoning with physically grounded rules for driving evaluation (Sun et al., 2026). VLMPlanner uses visual-language reasoning to guide an existing motion planner (Tang et al., 2025). Together, these studies establish that learned evaluation and semantic feedback can influence planning. What remains less isolated is the contribution of semantic judgment itself when the underlying motion planner is held fixed, and how that contribution depends on the actions available to the planner.

Driving Reasoning and Closed-loop Evaluation. Open-loop trajectory error can be a weak proxy for closed-loop driving quality and can be affected by ego-state shortcuts (Li et al., 2024), motivating closed-loop benchmarks such as Bench2Drive (Jia et al., 2024). DriveBench evaluates visual grounding and reasoning reliability for driving MLLMs (Xie et al., 2025), while MetaVQA combines VQA-based embodied scene understanding with closed-loop navigation and shows that VQA fine-tuning can improve sequential driving behavior (Wang et al., 2025c). Bench2Drive-VL further evaluates behavior-grounded MLLM reasoning along closed-loop driving rollouts (Jia et al., 2026). Rather than evaluating reasoning capability and driving performance in isolation, ActionGuide-Bench connects action-grounded judgments with closed-loop behavior, enabling analysis of where MLLM guidance improves driving across scenarios and how action-grounded competence relates to closed-loop utility.

7 Conclusion

In this paper, we study how action-grounded reasoning in MLLMs translates into closed-loop driving utility. We introduce ActionGuide-Bench to evaluate four action-grounded capabilities and connect them to online planner guidance, together with ActionGuide-Train for simulation-based supervision. Within this evaluation framework, existing MLLMs shows effectiveness to improve closed-loop driving as guides, while ActionGuide further improves both action-grounded reasoning and downstream driving performance. Our analyses show that behavior-level knowledge learned in simulation transfers well to real-world driving judgments, and that stronger action-grounded competence is associated with greater closed-loop utility. These findings highlight action grounding as a key interface between multimodal understanding and embodied control. By connect visual-semantic knowledge with concrete scene evidence, their consequences, and closed-loop feedback, ActionGuide takes a step toward *transferable MLLM reasoning for closed-loop driving*.

AI USE STATEMENT

Generative AI tools were used to assist with coding and language editing. Large language models were also used to generate structured reasoning annotations in our data pipeline. All AI-assisted content were reviewed and verified by the authors, who take full responsibility for the final work.

ETHICS STATEMENT

This work studies multimodal reasoning for autonomous driving, a safety-critical domain. Our experiments are conducted primarily on offline datasets and simulated closed-loop environments and should not be interpreted as evidence of readiness for real-world deployment. We use existing datasets according to their intended research use and do not collect new personally identifiable information or conduct human-subject studies.

REPRODUCIBILITY STATEMENT

The main paper describes the proposed framework, training procedure, and evaluation setup. Additional details on data construction, task definitions, validation rules, model configurations, training hyperparameters, and evaluation protocols are provided in the main paper and supplementary material to support reproducibility.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report, 2025a. URL https://arxiv.org/abs/2511.21631.

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. *arXiv preprint arXiv:2502.13923*, 2025b.

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 *2020 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 11621–11631. IEEE, 2020.

California DMV. *California Driver's Handbook*. State of California, Department of Motor Vehicles, 2025. URL https://www.dmv.ca.gov/portal/handbook/california-driver-handbook/. DL 600, Rev. 6/2025.

Qimao Chen, Fang Li, Yuechen Luo, Zehan Zhang, Haiyang Sun, Fangzhen Li, Bing Wang, Guang Chen, Yang Ji, Jiong Deng, Hongwei Xie, Hangjun Ye, Long Chen, and Yi Zhang. DriveReward: A comprehensive dataset and generative vision-language reward model for autonomous driving. *arXiv preprint arXiv:2606.08525*, 2026. URL https://arxiv.org/abs/2606.08525.

Daniel Dauner, Marcel Hallgarten, Tianyu Li, Xinshuo Weng, Zhiyu Huang, Zetong Yang, Hongyang Li, Igor Gilitschenski, Boris Ivanovic, Marco Pavone, Andreas Geiger, and Kashyap Chitta. Navsim: Data-driven non-reactive autonomous vehicle simulation and benchmarking. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pp. 1–16, 2017. URL https://proceedings.mlr.press/v78/dosovitskiy17a.html.

Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R. Qi, Yin Zhou, Zoey Yang, Aur'elien Chouard, Pei Sun, Jiquan Ngiam, Vijay Vasudevan, Alexander McCauley, Jonathon Shlens, and Dragomir Anguelov. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9710–9719, October 2021.

Haoyu Fu, Diankun Zhang, Zongchuang Zhao, Jianfeng Cui, Hongwei Xie, Bing Wang, Guang Chen, Hangjun Ye, Dingkang Liang, and Xiang Bai. Minddrive: A vision-language-action model for autonomous driving via online reinforcement learning. In *European Conference on Computer Vision*, pp. 503–522. Springer, 2026.

Xiaosong Jia, Zhenjie Yang, Qifeng Li, Zhiyuan Zhang, and Junchi Yan. Bench2Drive: Towards multi-ability benchmarking of closed-loop end-to-end autonomous driving. *arXiv preprint arXiv:2406.03877*, 2024. URL https://arxiv.org/abs/2406.03877.

Xiaosong Jia, Yuqian Shao, Zhenjie Yang, Qifeng Li, Zhiyuan Zhang, and Junchi Yan. Bench2Drive-VL: Benchmarks for closed-loop autonomous driving with vision-language models. *arXiv preprint arXiv:2604.01259*, 2026. URL https://arxiv.org/abs/2604.01259.

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. URL https://arxiv.org/abs/2410.22313.

Ellington Kirby, Alexandre Boulch, Yihong Xu, Yuan Yin, Gilles Puy, Éloi Zablocki, Andrei Bursuc, Spyros Gidaris, Renaud Marlet, Florent Bartoccioni, Anh-Quan Cao, Nermin Samet, Tuan-Hung Vu, and Matthieu Cord. Driving on registers. In *CVPR*, 2026.

Yongkang Li, Kaixin Xiong, Xiangyu Guo, Fang Li, Sixu Yan, Gangwei Xu, Lijun Zhou, Long Chen, Haiyang Sun, Bing Wang, Kun Ma, Guang Chen, Hangjun Ye, Wenyu Liu, and Xinggang Wang. Recogdrive: A reinforced cognitive framework for end-to-end autonomous driving. In *International Conference on Learning Representations*, volume 2026, pp. 157518–157556, 2026.

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*, pp. 14864–14873, 2024.

NVIDIA. Cosmos-reason2-8b. https://huggingface.co/nvidia/Cosmos-Reason2-8B, 2026. Hugging Face model card.

NVIDIA, Yulong Cao, Riccardo de Lutio, Sanja Fidler, Guillermo Garcia Cobo, Zan Gojcic, Maximilian Igl, Boris Ivanovic, Peter Karkus, Janick Martinez Esturo, Marco Pavone, Aaron Smith, Ellie Tanimura, Michal Tyszkiewicz, Michael Watson, Qi Wu, and Le Zhang. Alpasim: A modular, lightweight, and data-driven research simulator for autonomous driving, October 2025a. URL https://github.com/NVlabs/alpasim.

NVIDIA, Yan Wang, Wenjie Luo, Junjie Bai, Yulong Cao, Tong Che, Ke Chen, Yuxiao Chen, Jenna Diamond, Yifan Ding, Wenhao Ding, Liang Feng, Greg Heinrich, Jack Huang, Peter Karkus, Boyi Li, Pinyi Li, Tsung-Yi Lin, Dongran Liu, Ming-Yu Liu, Langechuan Liu, Zhijian Liu, Jason Lu, Yunxiang Mao, Pavlo Molchanov, Lindsey Pavao, Zhenghao Peng, Mike Ranzinger, Ed Schmerling, Shida Shen, Yunfei Shi, Sarah Tariq, Ran Tian, Tilman Wekel, Xinshuo Weng, Tianjun Xiao, Eric Yang, Xiaodong Yang, Yurong You, Xiaohui Zeng, Wenyuan Zhang, Boris Ivanovic, and Marco Pavone. Alpamayo-R1: Bridging reasoning and action prediction for generalizable autonomous driving in the long tail. *arXiv preprint arXiv:2511.00088*, 2025b.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5. Model card: https://huggingface.co/Qwen/Qwen3.5-4B.

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. URL https://arxiv.org/abs/2503.09594.

Omar Samir and DriveFusion Team. Drivefusionqa: A vision language language model for autonomous driving, 2026. URL https://huggingface.co/DriveFusion/DriveFusionQA.

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*, 2024. URL https://arxiv.org/abs/2312.14150.

Jingyu Song, Zhenxin Li, Shiyi Lan, Xinglong Sun, Nadine Chang, Maying Shen, Joshua Chen, Katherine A. Skinner, and Jose M. Alvarez. DriveCritic: Towards context-aware, human-aligned evaluation for autonomous driving with vision-language models. *arXiv preprint arXiv:2510.13108*, 2025. URL https://arxiv.org/abs/2510.13108.

Yuehao Song, Shaoyu Chen, Hao Gao, Yifan Zhu, Weixiang Yue, Jialv Zou, Bo Jiang, Zihao Lu, Yu Wang, Qian Zhang, and Xinggang Wang. Senna-2: Aligning VLM and end-to-end driving policy for consistent decision making and planning. *arXiv preprint arXiv:2603.11219*, 2026. URL https://arxiv.org/abs/2603.11219.

Xinglong Sun, Kevin Xie, Jenny Schmalfuss, Despoina Paschalidou, Xiuming Zhang, Sanja Fidler, Kashyap Chitta, and Jose M. Alvarez. DriveJudge: Rethinking autonomous driving evaluation with vision-language models. *arXiv preprint arXiv:2606.17362*, 2026. URL https://arxiv.org/abs/2606.17362.

Zhipeng Tang, Sha Zhang, Jiajun Deng, Chenjie Wang, Guoliang You, Yuting Huang, Xinrui Lin, and Yanyong Zhang. VLMPlanner: Integrating visual language models with motion planning. *arXiv preprint arXiv:2507.20342*, 2025. URL https://arxiv.org/abs/2507.20342.

Haochen Tian, Tianyu Li, Haochen Liu, Jiazhi Yang, Yihang Qiu, Guang Li, Junli Wang, Yinfeng Gao, Zhang Zhang, Liang Wang, Hangjun Ye, Long Chen, and Hongyang Li. Simscale: Learning to drive via real-world simulation at scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

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, 2025. URL https://proceedings.mlr.press/v270/tian25c.html.

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, 2025a. URL https://arxiv.org/abs/2504.04348.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, Zhaokai Wang, Zhe Chen, Hongjie Zhang, Ganlin Yang, Haomin Wang, Qi Wei, Jinhui Yin, Wenhao Li, Erfei Cui, Guanzhou Chen, Zichen Ding, Changyao Tian, Zhenyu Wu, Jingjing Xie, Zehao Li, Bowen Yang, Yuchen Duan, Xuehui Wang, Zhi Hou, Haoran Hao, Tianyi Zhang, Songze Li, Xiangyu Zhao, Haodong Duan, Nianchen Deng, Bin Fu, Yinan He, Yi Wang, Conghui He, Botian Shi, Junjun He, Yingtong Xiong, Han Lv, Lijun Wu, Wenqi Shao, Kaipeng Zhang, Huipeng Deng, Biqing Qi, Jiaye Ge, Qipeng Guo, Wenwei Zhang, Songyang Zhang, Maosong Cao, Junyao Lin, Kexian Tang, Jianfei Gao, Haian Huang, Yuzhe Gu, Chengqi Lyu, Huanze Tang, Rui Wang, Haijun Lv, Wanli Ouyang, Limin Wang, Min Dou, Xizhou Zhu, Tong Lu, Dahua Lin, Jifeng Dai, Weijie Su, Bowen Zhou, Kai Chen, Yu Qiao, Wenhai Wang, and Gen Luo. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency, 2025b. URL https://arxiv.org/abs/2508.18265.

Weizhen Wang, Chenda Duan, Zhenghao Peng, Yuxin Liu, and Bolei Zhou. Embodied scene understanding for vision language models via metavqa. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22453–22464. IEEE, 2025c.

Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting, 2023. URL https://arxiv.org/abs/2301.00493.

Shaoyuan Xie, Lingdong Kong, Yuhao Dong, Chonghao Sima, Wenwei Zhang, Qi Alfred Chen, Ziwei Liu, and Liang Pan. Are VLMs ready for autonomous driving? an empirical study from the reliability, data, and metric perspectives. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025. URL https://arxiv.org/abs/2501.04003.

Xin Zhou, Zongchuang Zhao, Zhibo Yang, Mingsheng Li, Humen Zhong, Shuai Bai, et al. Qwen-Drive-1.0: An initial step towards a vision-language foundation model for autonomous driving. *arXiv preprint arXiv:2609.00111*, 2026. URL https://arxiv.org/abs/2609.00111.

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. URL https://papers.nips.cc/paper_files/paper/2025/hash/2843fccca5bedd369a4764848b9bd546-Abstract-Conference.html.

APPENDIX

We provide additional details on the evaluation dataset (Section A), training-data construction (Section B), supplementary experiments (Section C), and qualitative results (Section D).

A EVALUATION DETAILS

A.1 ACTION-GROUNDED MLLM EVALUATION

A.1.1 DATA STATISTICS

Dataset Composition. The action-grounded evaluation set contains 2,539 questions constructed from 318 driving scenes drawn from the validation splits of nuScenes (Caesar et al., 2020), Waymo Open Motion Dataset (Ettinger et al., 2021), and Argoverse 2 Sensor (Wilson et al., 2023). It comprises 405 Evidence Grounding, 625 Consequence Prediction, 755 Corrective Guidance, and 754 Trajectory Verification questions. Figure 5 summarizes the task distribution and contribution of each source dataset.

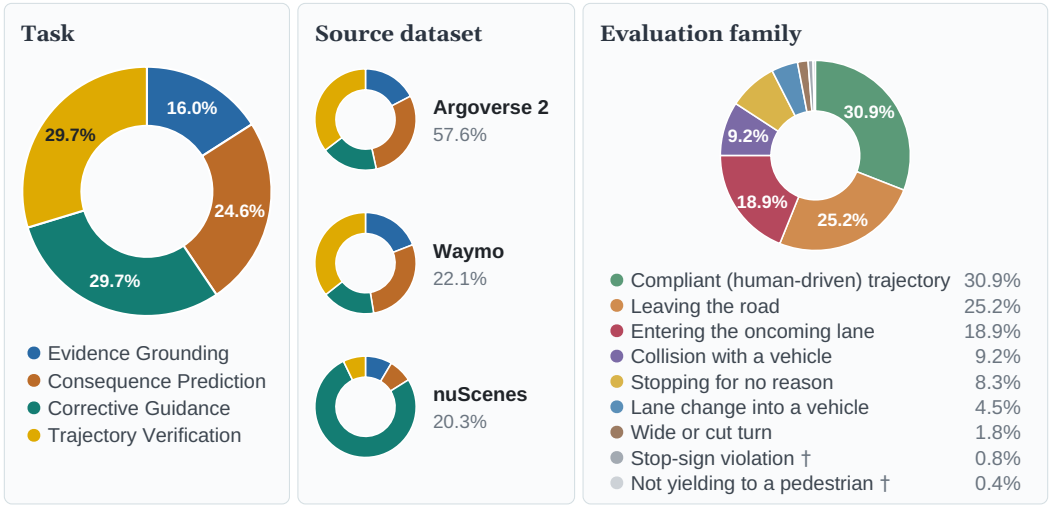

Figure 5: Composition of the action-grounded evaluation set. **Left**: distribution of the four tasks across 2,539 questions. **Middle**: task composition within each source dataset, together with the dataset's overall share of the benchmark. **Right**: distribution of evaluation families for Evidence Grounding, Consequence Prediction, and Trajectory Verification (1,784 questions).

Evaluation-family Coverage. Evidence Grounding, Consequence Prediction, and Trajectory Verification are organized into 9 behavioral families covering compliant trajectories and common driving failures, including road departure, wrong-way motion, collisions, unnecessary stopping, unsafe lane changes, turning errors, and traffic-rule violations. Corrective Guidance is treated separately and is grouped by the direction of the correct candidate, as shown in Figure 6.

A.1.2 EVALUATION CONSTRUCTION

The evaluation dataset is constructed using a pipeline similar to Figure 2. However, when constructing examples from existing real-world data, the observed scene elements must be treated as fixed conditions, unlike CARLA-based data construction where these elements can be explicitly controlled or synthesized. The following steps show the complete pipeline.

Observation and Trajectory representation. Each evaluation instance is anchored at a timestamp t_0 in a source driving log. For Evidence Grounding, Consequence Prediction, and Trajectory

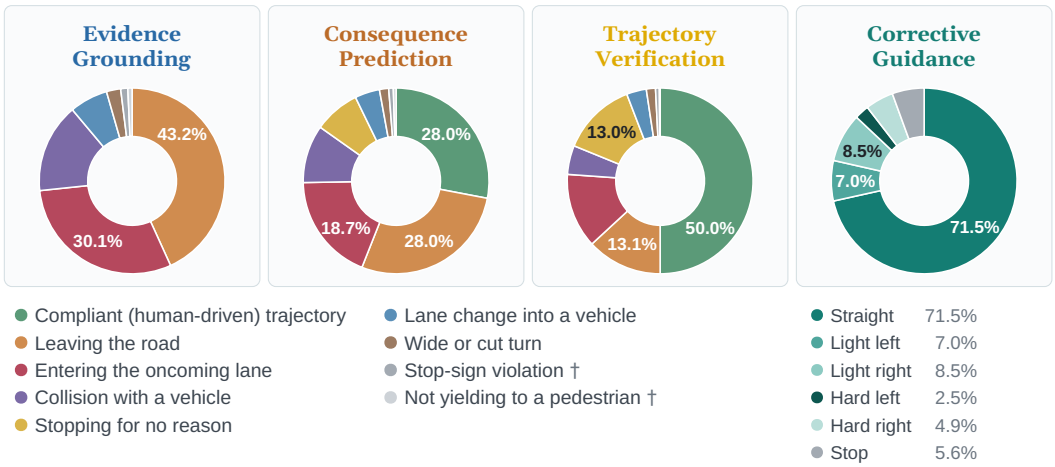

Figure 6: Evaluation-family distribution within each action-grounded task. The first three tasks are grouped by behavioral family, while Corrective Guidance is grouped by the direction of the correct candidate.

Verification, the model receives four historical front-camera frames at -2.0, -1.5, -1.0, and -0.5 s, followed by the current front, front-left, and front-right views. Corrective Guidance draws its candidates on the current front image. A candidate trajectory is represented by eight ego-frame waypoints sampled at 0.5 s intervals over a 4 s horizon and is rendered on the current front image. If the trajectory remains too close to the ego vehicle to yield a reliable image-space overlay, we instead provide its eight waypoint coordinates in the prompt.

Anchor and Candidate Construction. We construct evaluation questions in a category-driven manner. For each evaluation family, we first identify locations in the source map where the corresponding behavior can be meaningfully instantiated; we refer to each such candidate-generation point as an *anchor* (s, t_0). Anchors are sampled at 1.5 s intervals and retained only if the logged 4 s ego future remains on the drivable surface, makes at least 4 m of progress, and does not overlap a recorded road user.

For each retained anchor, we generate candidate ego trajectories according to the evaluation families supported by the local scene context. The source observations and recorded behavior of other road users remain fixed, while only the proposed ego trajectory is modified. Behavioral violations such as leaving the road, entering the oncoming lane, stopping for no reason, lane changes into a vehicle, and wide or cut turns are synthesized through controlled trajectory perturbations. Event-dependent families, including stop-sign violations and failures to yield to a pedestrian, are instantiated only when the corresponding event and visual evidence are present in the source log.

Candidate Validation. Before forming a benchmark question, each generated candidate is validated against the source map and the recorded trajectories of surrounding actors to ensure that it realizes the intended behavioral category without introducing a confounding violation. The validation criteria are category-specific. For example, a leaving-the-road candidate must exit the drivable surface without first entering an opposing lane, whereas an oncoming-lane candidate must occupy a lane whose heading opposes the ego direction. Interaction-dependent candidates are replayed against the recorded actor trajectories to verify that the intended conflict is induced by the modified ego trajectory. After validation, the source scene together with the validated ego candidate forms a complete scene–trajectory pair for downstream question construction. Additional ground-truth and ambiguity controls are described in Section A.1.3.

Task Derivation. The validated scene–trajectory pairs are converted into the four action-grounded tasks. Trajectory Verification produces one binary acceptability question for each candidate trajectory. Evidence Grounding asks which visible scene element is relevant to the candidate's primary validated

Task	Ground truth	How established	Ambiguity control
Evidence Grounding	Relevant scene evidence	Derived from the validated behavior associated with the candidate.	Required evidence must be visible in the model input; co-valid distractors are removed.
Consequence Prediction	Outcome of executing the candidate	Determined from map geometry, recorded actor tracks, and the validated candidate behavior.	Options follow a common outcome definition; cases with multiple valid outcomes are excluded.
Trajectory Verification	Binary trajectory acceptability	Logged compliant trajectories provide positive examples; validated behavioral violations provide negative examples.	Indeterminate cases are removed, and positive and negative labels are balanced.
Corrective Guidance	Appropriate directional correction	Obtained by matching the verified logged future to the closest direction in the fixed six-way candidate set.	Candidate order and visualization are randomized independently of the correct answer.

Table 3: Ground-truth construction and ambiguity controls for the four action-grounded tasks.

violation, while Consequence Prediction asks what would occur if that candidate were executed. Because stopping for no reason has no scene element to point to, it yields no Evidence Grounding question. Corrective Guidance instead presents a six-way anchor library (hard left, light left, straight, light right, hard right, and stop) and asks which candidate, executed now, best takes the vehicle in the appropriate direction; the ground-truth trajectory replaces the nearest member of the library and serves as the answer. The construction of answer choices and controls against presentation shortcuts is described in Section A.1.3.

A.1.3 Ground Truth and Shortcut Controls

Ground-truth validation. We use the original ego future trajectory as the compliant reference and construct counterfactual candidates by modifying only the ego trajectory. Each candidate is validated against the source map and recorded actor tracks using category-specific geometric and interaction checks. We retain a candidate only when the validated outcome matches the behavior targeted by the question, thereby avoiding ambiguous cases in which an incidental event could be mistaken for the intended violation. Table 3 summarizes how ground truth is established and how ambiguity is controlled for each task.

Ambiguity and Representation Controls. When an anchor supports multiple violations, a fixed priority rule selects the event used to form the question. Distractors are drawn from a shared pool of driving-relevant alternatives, and exclusion rules remove options that could be simultaneously true. Answer positions and trajectory colors are assigned independently of candidate semantics, and all alternatives within a question use the same presentation format. To limit near-duplicate questions, questions that share the same scene, evaluation family, and answer are separated by at least 5 s.

A.1.4 Human Quality Control

Although the evaluation set is generated through rule-based construction and automatic validation, we further perform human quality control to identify residual ambiguities in scene selection, candidate construction, and answer formulation. We uniformly sample 300 instances across task–scenario combinations and ask six expert annotators to review them using the same visual input and question presented to the model. Annotators judge whether the constructed question is well supported by the scene and whether the designated answer is unambiguous.

Quality control is performed in two iterative rounds. After the first round, flagged cases are analyzed to identify systematic issues in the generation rules, candidate validators, or answer templates, and the corresponding construction logic is updated. A second round is then conducted on newly sampled



Figure 7: **Real-world Benchmark Cases:** Evidence Grounding & Consequence Prediction.

instances using the revised pipeline. This human-in-the-loop procedure allows annotation feedback to directly improve the benchmark construction rules before the evaluation set is finalized.

A.1.5 Evaluation Cases

Representative examples across the four evaluation tasks are shown in Figures 7 and 8.

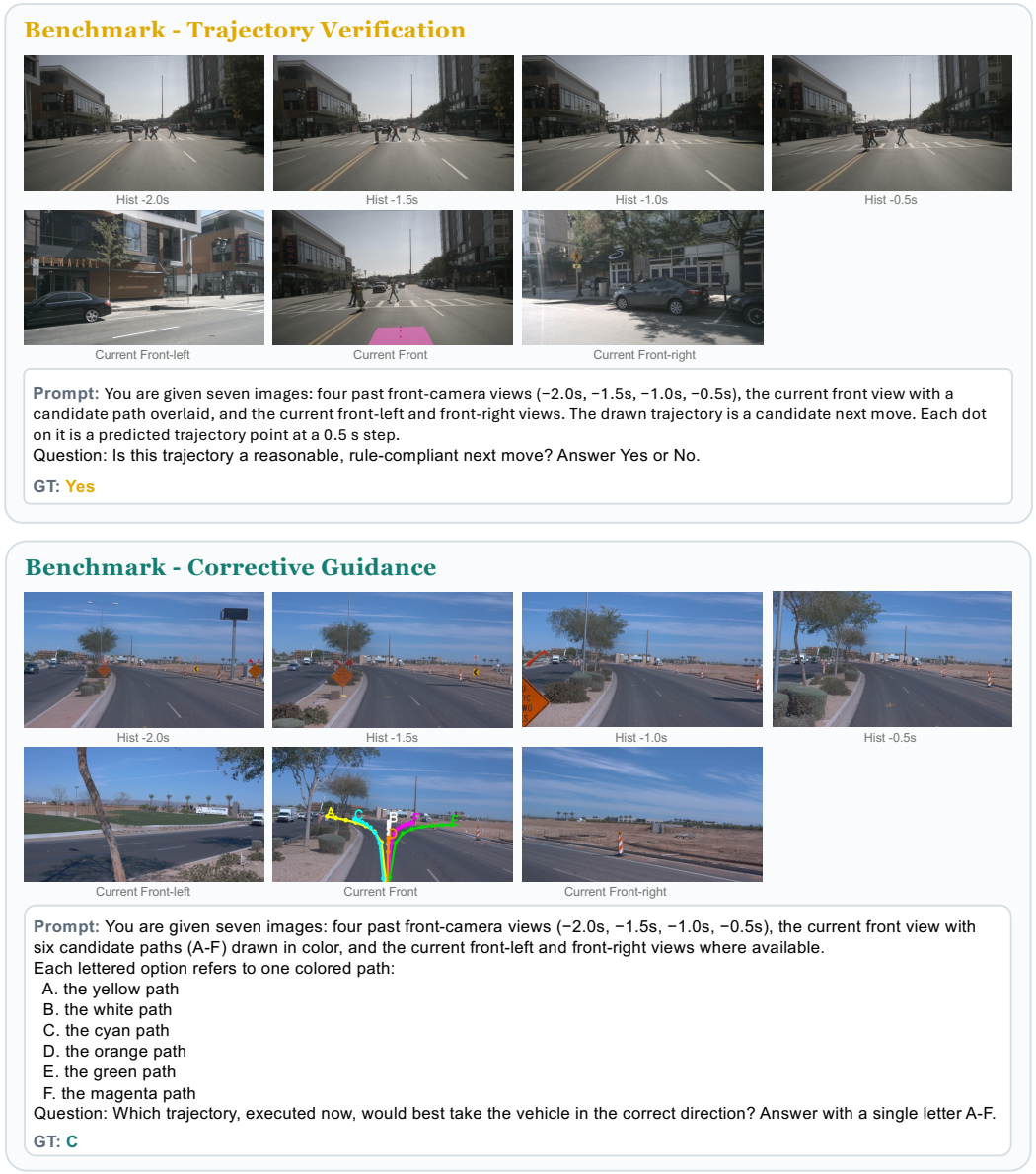

Figure 8: **Real-world Benchmark Cases:** Trajectory Verification & Corrective Guidance.

A.2 Closed-Loop Evaluation

A.2.1 AlpaSim Senarios

To focus the closed-loop evaluation on semantically rich and interaction-intensive settings, we manually curate AlpaSim scenarios that require reasoning about traffic-control state, interactions with other road users, roadway context, or maneuver context. The resulting set spans 18 scenario categories, including stop signs and signal transitions, pedestrians and school buses, work zones and narrow streets, protected and unprotected turns, freeway exits, parking starts, and sharp turns, while also retaining nominal settings such as freeway and city cruising and traffic jams. Table 4 provides the full category-level breakdown and corresponding AlpaSim case IDs. Categories are non-exclusive, so a single case may appear in multiple rows when it contains multiple decision-relevant elements.

Table 4: Human-curated AlpaSim cases used for the focused closed-loop evaluation. The benchmark contains 131 unique episodes. Categories are non-exclusive, and cases with multiple decision-relevant scene elements may appear in more than one row. Scene IDs are abbreviated to the first eight characters of their original AlpaSim UUIDs; these prefixes are unique within the dataset.

Scenario category	**Count**	**AlpaSim scene IDs**
Freeway Cruise	8	`ffd5706c, fea4c0bf, fe0c2168, fd0a220e, fbd72c7a, fbbadc60, fad0784a, fa1f799f`
Traffic Jam	5	`ffb26632, fe2979f4, e8a54ce0, 9c892e05, 8aaa1135`
City Cruise	13	`ff823af7, ff587666, ff25b4c3, fece9f19, fdfb5073, fce18b62, fbdce016, fba5d7c7, fb90f9e6, fb6d1a18, fad840eb, faa097ac, fa507a17`
Pedestrians	13	`ff3ded58, db009e5d, b0e94b02, a96274d4, 8fd8f4f0, 82da2d46, 7b186d92, 4d9be4c7, 35e082a4, 18d28973, 127c4f85, 0b10bce8, 095cf563`
Turn Right On Red	4	`ff28f961, d101f9f0, 3232fdb7, 16fc3dfe`
Stop Sign / Red Light	13	`fe6c404c, fc9fde87, fadc73da, f9b7dcc6, f7209243, f005ac5f, e904e9c0, de8fb30e, e00752a1, d4c97cc1, d2fcf4b2, c9fccb24, 9fa1d6eb`
Red Light Turn Green	9	`fe240bc3, f0674f19, edcf41a4, e64757e8, cb09cf43, c7c58600, b42bee1f, a5c06f4f, a35a2bdf`
Green Light Turn Yellow	6	`fd22bde9, eaba3f6a, e66ebf31, bae463d3, 823d47b0, 0d4893f5`
Exit Freeway	2	`faa6cd0c, eab8240a`
Always Green Light	9	`f886551d, f8d31373, f5192992, f41175b5, f244c735, eac1bf3c, e8cf41e8, e883822a, e6c37af0`
Work Zone	12	`f095ecaa, eedc28f2, eda66f79, d1fca92f, ce0cb386, b1b170d0, 954c1983, 86dce47a, 7a824ffa, 2da7fb57, 22a92557, 0a18c5a4`
Protected Left Turn	10	`eaf4352c, e742759b, de8fb30e, c7c58600, a5c06f4f, 96eb4700, 93982b71, 7ce9e86e, 08990ec8, 05ecbbf5`
School Bus / Zone	2	`e8efc7cd, e7ddb7f0`
Start from Parking	2	`d2715023, 33fe94ae`
Unprotected Left Turn	4	`cc289222, 6f4fc882, 5775a998, edcf41a4`
Narrow Street	12	`b2d85ef6, b0e94b02, 9190bd3f, 8e9691d4, 89d4188e, 8701a102, 7ae6bec8, 55b7be99, 48a50c2a, 36626308, 3080d0eb, 048b974e`
Sharp Right Turn	9	`afa37955, a9c81fec, 9628d193, 90d1908c, 89d4188e, 82da2d46, 6bec4e02, 4bd800df, 14a02b32`
Shadow of Tree	5	`632d7f35, 6404193c, 5c871a72, 535b24e1, 48a48ad9`

A.2.2 Rollout Protocol and Statistical Estimation

All methods are evaluated on the same 131 AlpaSim scenarios with the same start time (1.7 s) and configured seed (42). We use AlpaSim v0.91.0 (commit `10cae1c2`) and its evaluation package v1.36.0 from the same checkout, invoked through the GS-World scorer adapter.

The DrivoR planner and AlpaSim MPC controller are called at 10 Hz; 0.1 s of the planned trajectory is executed before the next planner call. The MLLM makes a new selection at 2 Hz, with its selection held between calls. A rollout ends upon collision, deviation of more than 30 m from the reference route, route completion, reaching the end of the scenario log, or 300 simulation steps.

B Training Data

B.1 Data Composition

Our training corpus contains 16,799 simulation-generated samples spanning four tasks and 16 driving scenario categories. The task distribution is moderately skewed toward trajectory verification (33.9%) and consequence prediction (26.5%), followed by evidence grounding (20.6%) and corrective guidance (19.0%).

The dataset covers a diverse set of driving behaviors and violations, including unsafe lane changes, off-road driving, wrong-way driving, stop-sign violations, red-light violations, pedestrian and vehicle

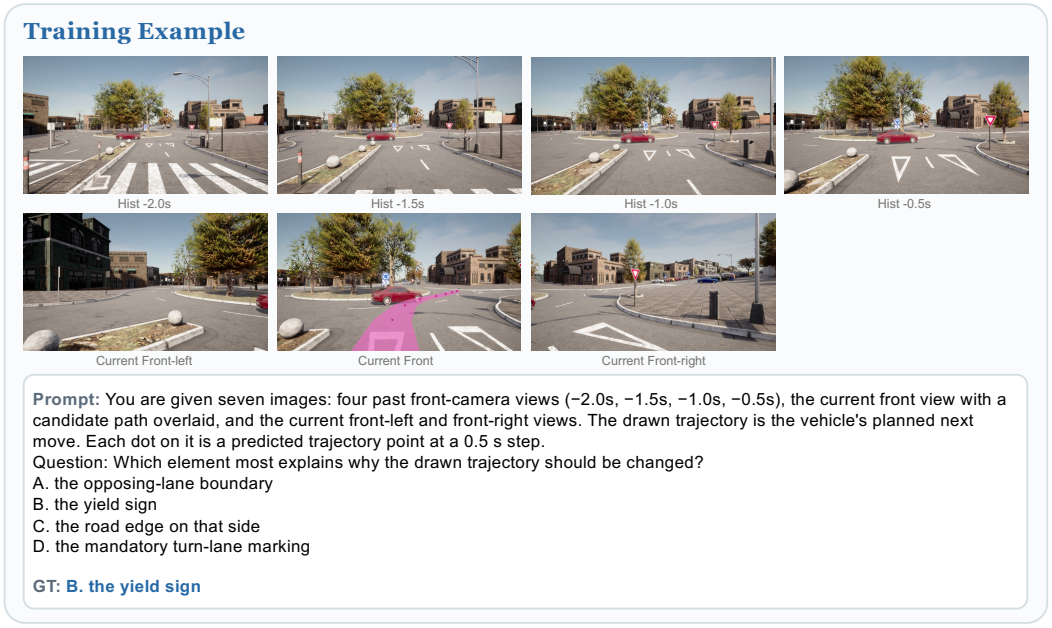

Figure 9: Train Data Example in ActionGuide-Train.

yielding, construction-zone avoidance, unnecessary stopping, and stalling after a full stop, together with compliant examples.

Scenario coverage is structured across tasks. 15 rule- and behavior-oriented categories are used for evidence grounding, consequence prediction, and trajectory verification, with 14 categories appearing in all three. Unnecessary stopping does not include evidence-grounding samples by construction. Corrective guidance is generated separately from a directional-selection category, contributing 3,189 samples and focusing on selecting an appropriate corrective direction.

B.2 Data Construction Details

Scenario Construction Training data are generated in CARLA using the same overall category-driven pipeline as the evaluation data: we first select a target driving behavior, construct candidate ego trajectories, and retain only examples that pass category-specific validation. The main difference is that, in simulation, the surrounding scene can also be actively constructed rather than being conditioned on elements already present in a real-world log.

For geometry- and rule-based behaviors, such as road departure, wrong-way driving, unnecessary stopping, and turning errors, we configure the road context and synthesize ego trajectories in the same manner as in the evaluation pipeline. For interaction-dependent behaviors, we additionally determine when and where the relevant interaction should occur. Given a candidate ego trajectory, we identify the spatial region and time interval in which a conflict or yielding decision is expected, and place the corresponding scene element, including a pedestrian, conflicting vehicle, parked vehicle, or construction obstacle. So that its motion intersects the ego behavior at the intended conflict point. This allows the simulator to generate controlled scene–behavior pairs with known interaction structure rather than relying on rare events to occur naturally.

Observation and Candidate Generation Each source observation contains a short front-view history together with the current front, front-left, and front-right views. Candidate trajectories are represented by eight waypoints over a four-second horizon and are generated from the current ego state, scene geometry, and the target behavior category. Trajectory overlays are rendered on the current front view, while historical and side views remain unchanged. The resulting observations are then converted into the four action-grounded training tasks described in Section 2.2.

Interaction-Aware Scene Placement For scenarios involving pedestrians or other vehicles, scene construction is conditioned on the candidate ego motion. We first estimate the future ego occupancy along the candidate trajectory and determine a target interaction window. The interacting element is then initialized with a position and motion profile such that the two trajectories reach the intended conflict region at compatible times.

This procedure is used to construct behaviors such as yielding to pedestrians, vehicle conflicts, unsafe lane changes, and obstacle avoidance. The interaction parameters are varied across examples, including the conflict location, relative arrival time, actor speed, and lateral placement, to avoid coupling a behavioral label to a single geometric template.

Validation and Labeling Candidate validation follows the same principle as the evaluation-data construction: labels are assigned from the behavior observed under category-specific geometric and interaction checks rather than from generator intent alone. Candidates are checked against road geometry, lane direction, surrounding actors, and scenario-specific traffic constraints. Interaction scenarios are additionally verified to ensure that the intended actor is indeed responsible for the relevant conflict or yielding decision.

Examples are retained only when the observed outcome matches the targeted behavior and the required visual evidence is available in the model input. Ambiguous cases, including trajectories that trigger multiple confounding violations or interactions whose timing cannot be determined reliably, are discarded. Figure 9 shows an example from our constructed training data.

B.3 Reasoning Generation

For Trajectory Verification, we additionally construct structured language reasoning that connects scene understanding to the final trajectory judgment as shown in Figure 10. Because CARLA provides access to the full simulator state, we first extract a structured scene schema describing the map, ego state and candidate trajectory, surrounding actors, and relevant traffic events. The schema includes information such as lane geometry and crosswalks, ego motion, actor positions and timing, and interaction events such as yielding or conflicts.

We provide both the rendered observations and this structured schema to Qwen3-VL-32B to generate a reasoning trace that follows the same progression as our action-grounded tasks: understanding the scene and temporal context, identifying the decision-relevant evidence, anticipating the consequence of executing the proposed trajectory, and finally determining whether the trajectory is acceptable. In this way, the generated reasoning explicitly connects scene understanding with Evidence Grounding and Consequence Prediction before arriving at the Trajectory Verification decision.

The simulator schema is used only as privileged supervision during reasoning generation. We instruct the teacher model to retain only information that can be supported by the visual input and to avoid simulator-only quantities, such as exact speeds, distances, or other numerical states that cannot be reliably inferred from the images. This yields visually grounded and structured reasoning while using simulator state to improve consistency and coverage.

C More Experiments

C.1 More Evaluation Results on AlpaSim

We further evaluate additional planners on our curated AlpaSim scenarios as a complementary study to Table 2. Comparing these results with the action-grounded evaluation in Table 1, we observe that action-grounded reasoning performance is not necessarily proportional to closed-loop scene score. For example, Alpamayo-1.5 achieves strong closed-loop driving performance despite relatively weaker explicit action-grounded reasoning, while models such as ReCogDrive and Qwen-Drive-1.0 exhibit different trade-offs between the two. This mismatch suggests that strong closed-loop driving does not necessarily imply that the model has learned an explicit, language-aligned understanding of driving behavior.



Figure 10: Generated Reasoning in ActionGuide-Train.

Table 5: Evaluation results for additional planners on AlpaSim.

Model	Collision all (%)↓	Collision at-fault (%)↓	Off-road (%)↓	Progress (%)↑	Scene Score all↑	Scene Score at-fault↑
ReCogDrive	62.6	61.1	**9.20**	30.2	13.1	13.2
MindDrive	64.1	62.6	10.7	38.4	16.1	16.9
AutoVLA	54.2	51.9	37.4	79.5	26.8	27.5
Qwen-Drive-1.0	45.8	42.7	25.2	68.5	33.2	35.2
DrivoR	42.0	38.2	22.9	77.1	42.6	43.7
Alpamayo-1.5	**16.8**	**14.5**	22.1	**85.3**	**63.6**	**63.8**

D QUALITATIVE EXAMPLES

Closed-Loop Repairs and Regressions Figure 11 shows representative closed-loop cases illustrating both the benefits and limitations of MLLM guidance. Gray trajectories denote proposals from the frozen DrivoR planner, while blue and red trajectories indicate the behaviors selected by DrivoR and Cosmos-Reason2-8B, respectively.

In the first case, DrivoR advances toward queued traffic, while the MLLM guide selects Stop. The intervention keeps the ego vehicle behind the queue and avoids the collision that occurs without guidance. This example shows how MLLM guidance can correct an unsafe planner selection.

The second case shows a regression introduced by MLLM guidance. DrivoR selects a slower forward behavior near a constrained roadside region, while the MLLM guide selects a more aggressive Go

behavior. The guided rollout subsequently leaves the drivable area. This example shows that MLLM intervention can also override a safer native planner decision.

The third case illustrates Corrective Guidance when none of the native proposals is accepted. The MLLM rejects all top-10 DrivoR proposals at the junction and selects the Stop anchor. The resulting correction allows the ego vehicle to yield to crossing traffic instead of committing to an unsuitable native proposal.

The final case highlights the distinction between a high-level action decision and its execution. The MLLM selects Stop behind a lead vehicle, but the ego vehicle retains residual forward speed and cannot stop immediately. The rollout therefore still results in a collision despite the appropriate high-level correction.

Together, these cases show that MLLM guidance can repair planner decisions and provide useful fallback behavior, while incorrect interventions and execution constraints can still lead to closed-loop failures.



Figure 11: Representative closed-loop repairs and failures with MLLM guidance over DrivoR.