EXPLORING WHAT TO PRESERVE FOR REASONING AND ACTION IN STRUCTURED PRUNING OF DRIVING VLAS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reasoning vision–language–action (VLA) models for autonomous driving use chain-of-thought (CoT) reasoning to guide action generation, but incur substantial memory and computational costs. We study structured pruning for these models, focusing on preserving both scene understanding and trajectory prediction. Our analysis of Taylor importance in Alpamayo-1.5-10B shows that reasoning and trajectory objectives assign different importance to the same vision–language model (VLM) structures. In the action expert, MLP channels exhibit more stable importance rankings across denoising steps than query heads, and a small fraction of channels accounts for most of the measured importance. Based on these findings, we propose MoOP, which combines importance ranks from both objectives for VLM pruning and aggregates stepwise importance to select a fixed subset of expert MLP channels. At 24.0% total parameter reduction, VLM-only pruning with MoOP retains 94.0% of the dense model's LingoQA score and achieves the highest closed-loop scene scores among methods at the same budget. Removing 93.75% of expert MLP channels increases total parameter reduction to 39.4%, with nearly unchanged open-loop errors relative to VLM-only pruning and competitive closed-loop performance. The resulting model achieves a $1.37\times$ end-to-end inference speedup and 36.6% lower peak memory than the dense model, without post-pruning fine-tuning.

1 INTRODUCTION

Reasoning vision–language–action (VLA) models for autonomous driving (NVIDIA et al., 2025c; Zhou et al., 2025; Fu et al., 2025; Yuan et al., 2026) extend action prediction with explicit chain-of-thought (CoT) reasoning (Wei et al., 2022) to understand complex driving situations and explain driving decisions. Alpamayo (NVIDIA et al., 2025c), for example, conditions a flow-based (Lipman et al., 2023) action expert on context from its backbone, which includes Chain-of-Causation (CoC) reasoning generated by a vision–language model (VLM). Incorporating CoC in action denoising process improves trajectory prediction and closed-loop driving over a trajectory-only baseline, particularly in challenging scenarios (NVIDIA et al., 2025c). However, this procedure involves substantial computation: a large VLM generates CoC tokens autoregressively, and an action expert runs repeatedly to predict trajectories through iterative denoising. As autonomous driving requires continuous action updates under rapidly changing scene conditions, this computational bottleneck presents a critical challenge for real-time deployment (Li et al., 2026). Thus, reducing their memory footprints and computational cost is critical.

Structured model pruning offers a promising direction to reduce this cost by removing unnecessary parameters based on objective-driven importance scores (Ma et al., 2023; Li et al., 2025; Muralidharan et al., 2024). However, applying structured pruning to reasoning VLA is particularly challenging. Unlike standard VLM, reasoning VLA leverages a shared VLM backbone for two different roles (*i.e.,* scene-level CoC reasoning and action expert conditioning), while the action expert module operates under different mechanism (*i.e.*, iterative denoising steps). Deciding what to prune, therefore, calls for a nuanced assessment of structural importance across both distinct training objectives and differing architectural components.

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.

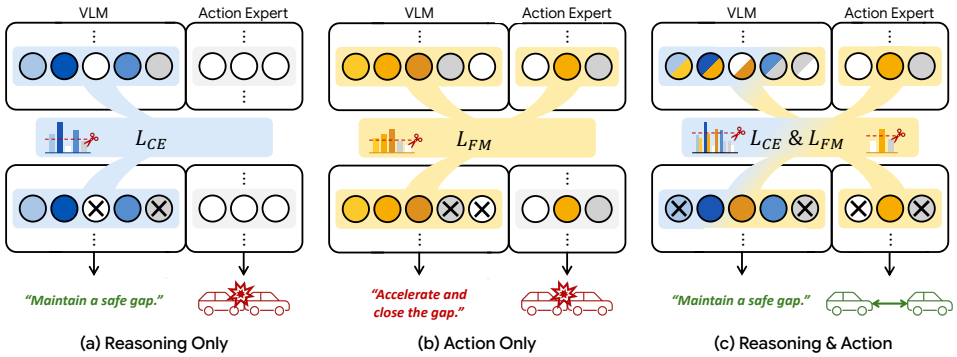


Figure 1: **Overview of Mixture of Objective Pruning (MoOP).**

To derive a tailored pruning strategy for reasoning VLA, we examine Alpamayo (NVIDIA et al., 2025c), a representative VLA framework for autonomous driving that predicts action trajectories along with CoC reasoning traces. Specifically, we assess the importance of structural units for both reasoning and action predictions using Taylor importance (Molchanov et al., 2017; 2019; LeCun et al., 1989; Hassibi et al., 1993), which estimates the loss change induced by removing each unit. We further analyze the distribution of importance scores within the action expert across iterative denoising steps, accounting for its distinct operating mechanism from the autoregressive VLM backbone.

Briefly, our analysis on the VLM backbone reveals that parameters important for reasoning and those important for action generation are only partially aligned, such that neither objective fully represents the other. Thus, identifying pruning candidates based solely on a single objective leads to either CoC collapse or a sharp decline in action prediction performance, as described in Figure. 1. These findings suggest that preserving both scene understanding and action generation requires accounting for both objectives during pruning.

For the action expert, a key consideration is that parameter importance may vary across denoising steps. Examining the distribution of action importance over denoising steps, we find that the importance metrics for attention heads shift with the step, whereas MLP channel importance remains nearly constant. Furthermore, while the importance score for Q-heads is broadly distributed, Figure. 5c discloses that the top 10% of MLP channels account for approximately 98.2% of the total measured importance.

Motivated by these findings, we propose Mixture-of-Objective Pruning (MoOP), which jointly considers the dual objectives (e.g., CoC generation and action prediction) and the distinct architectural characteristics of both VLA modules. For the VLM backbone, where the two distinct objectives exhibit only partial alignment, MoOP identifies parameter importance using their maximum importance and eliminates those that are critical to neither CoC reasoning nor trajectory generation. In the action expert, MoOP instead prunes dense MLP channels that do not contribute consistently to prediction across denoising steps. At a matched 24.0% total parameter reduction, our MoOP matches or outperforms the compared methods on open-loop trajectory metrics and achieves the highest driving QA and closed-loop scene scores. Adding 93.75% MLP-channel pruning in every action expert layer brings total parameter reduction to 39.4%, while retaining competitive trajectory prediction and closed-loop driving performance. The resulting model achieves a 1.37× end-to-end inference speedup and 36.6% lower peak memory than the dense model, without post-pruning fine-tuning.

Our main contributions are as follows:

- We demonstrate that VLM pruning must simultaneously account for both reasoning and action generation to preserve scene understanding and driving capability.
- We discover that a small subset of MLP channels in the action expert is sufficient to maintain robust action prediction quality.
- We propose MoOP, which achieves state-of-the-art results by preserving performance at significantly higher pruning ratios than existing methods.

2 Related Work

Reasoning Vision–Language–Action Models for Autonomous Driving. Driving VLAs connect scene understanding and language reasoning with action prediction. DriveVLM (Tian et al., 2024) integrates scene analysis with hierarchical planning, while ORION (Fu et al., 2025) conditions a generative planner on an LLM-generated planning token. AutoVLA (Zhou et al., 2025) unifies reasoning and discrete actions within an autoregressive model. Alpamayo (NVIDIA et al., 2025c) generates Chain-of-Causation (CoC) reasoning and conditions a separate action expert on the VLM context to predict continuous trajectories. We study how to prune this architecture while preserving its contributions to reasoning and trajectory prediction.

Pruning Language and Vision-Language Models. Language-model pruning methods estimate importance through gradients or weight–activation statistics, as in LLM-Pruner (Ma et al., 2023) and Wanda (Sun et al., 2024), and optimize layerwise sparsity allocation, as in Týr-the-Pruner (Li et al., 2025). For VLMs, SlimVLM (Wen et al., 2026) accounts for visual-token redundancy and module sensitivity. Reasoning-aware approaches further adapt compression to the generation process: RAC (Lucas et al., 2026) reconstructs activations from model-generated CoT (Wei et al., 2022) traces, while MuCRASP (Dutta & Aditya, 2026) considers reasoning-critical transitions and cross-modal alignment. Our work examines a shared VLM that supports both textual reasoning and continuous action generation, combining importance measured under the two objectives.

VLA Compression. VLA compression has explored structural pruning, quantization, and inference acceleration. For robotic manipulation, EfficientVLA (Yang et al., 2025) combines layer pruning, token selection, and action-head caching; CLP (Nguyen et al., 2026) removes redundant VLM and action-head layers before fine-tuning; and QVLA (Xu et al., 2026) allocates channelwise precision using action sensitivity. In autonomous driving, Depth-Wise Probing and Pruning (Babu et al., 2026) removes ORION decoder layers based on changes in the planning token, while FlashDrive (Li et al., 2026) accelerates Alpamayo through caching and speculative decoding. We combine CoC- and trajectory-based VLM pruning with action-expert MLP pruning guided by importance stability and concentration across denoising steps, without post-pruning fine-tuning.

3 Preliminaries

3.1 Alpamayo - Reasoning VLA for Autonomous Driving

Alpamayo (NVIDIA et al., 2025c) combines a reasoning VLM backbone (NVIDIA et al., 2025a) with a flow-based (Lipman et al., 2023) action expert (Figure 2). Given multimodal input $x = (\mathcal{I}, e, p)$ which consists of multi-camera images $\mathcal{I}$, ego history e, and a text prompt p, the VLM autoregressively generates a Chain–of–Causation (CoC) $y = (y_1, \ldots, y_T)$ that links a driving decision to its causal factors. The VLM thus processes one sequence of images, ego-history and prompt text tokens followed by CoC tokens. In each layer ℓ of the action expert, action tokens attend jointly to their own keys and values and to the full key–value (KV) cache of VLM layer ℓ. Conditioned on $\mathbf{h}_\theta(x, y) = \{(K_\ell, V_\ell)\}_{\ell=0}^{L-1}$, the expert denoises Gaussian noise in S steps into acceleration and curvature values, converted into a future ego trajectory.

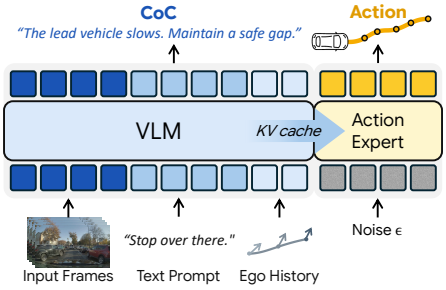


Figure 2: **Alpamayo architecture.** The VLM encodes frames, prompt and ego history, generates the CoC, and its KV cache conditions the action expert, which denoises noise ϵ into a trajectory.

Training objectives. The VLM p_θ is trained with cross-entropy (CE) on a ground-truth CoC y, and the action expert v_ϕ with flow matching (FM) (Lipman et al., 2023) on a normalized ground-truth action a, using $z_t = (1-t)\epsilon + ta$ with $\epsilon \sim \mathcal{N}(0, I)$, $t \sim \mathcal{U}(0, 1)$ and target velocity $v_t^* = a - \epsilon$:

$$\mathcal{L}_{\text{CE}} = -\frac{1}{T}\sum_{j=1}^{T} \log p_\theta(y_j \mid x, y_{<j}), \qquad \mathcal{L}_{\text{FM}} = \mathbb{E}_{t,\epsilon} \left\| v_\phi(z_t, t; \mathbf{h}_\theta(x, y)) - v_t^* \right\|_F^2 . \tag{1}$$

3.2 Structured Pruning and Taylor Importance

Structured pruning. Structured pruning reduces model size by removing computational units with low estimated importance, together with their parameters (Ma et al., 2023; Li et al., 2025; Muralidharan et al., 2024). We prune two kinds of units, attention query heads and intermediate MLP channels, in both the VLM and the action expert. Removing query head u deletes its d_h rows of W_ℓ^{Q} and the matching columns of W_ℓ^{O}; the key and value projections are kept, since several query heads share one KV head under grouped-query attention (Ainslie et al., 2023). Removing MLP channel u deletes row u of W_ℓ^{up} and W_ℓ^{gate} and column u of W_ℓ^{down}. Units are ranked by importance within each layer and unit type, the lowest-ranked ones are removed according to the budget, and the remaining weights are left unchanged.

Taylor importance. First-order Taylor importance estimates how much a loss changes when a unit is removed (Molchanov et al., 2017; 2019). We attach a gate $\alpha_{\ell,u} = 1$ to the output of each unit: the head's attention output before W_ℓ^{O}, or the channel's activation before W_ℓ^{down}. Removing the unit sets $\alpha_{\ell,u} = 0$, which to first order changes the loss on clip c by $-\partial\mathcal{L}_c/\partial\alpha_{\ell,u}$. The importance of the unit under a loss $\mathcal{L}$ is

$$\frac{\partial \mathcal{L}_c}{\partial \alpha_{\ell,u}} = \sum_p \mathbf{o}_{\ell,u,p}^{\top} \frac{\partial \mathcal{L}_c}{\partial \mathbf{o}_{\ell,u,p}}, \qquad (2) \qquad\qquad I_{\mathcal{L}}(\ell,u) = \frac{1}{N}\sum_{c=1}^{N} \left| \frac{\partial \mathcal{L}_c}{\partial \alpha_{\ell,u}} \right|, \qquad (3)$$

where $\mathbf{o}_{\ell,u,p}$ is the unit's output at token position p, the sum runs over all positions the unit processes, and N is the number of calibration clips.

4 Analysis

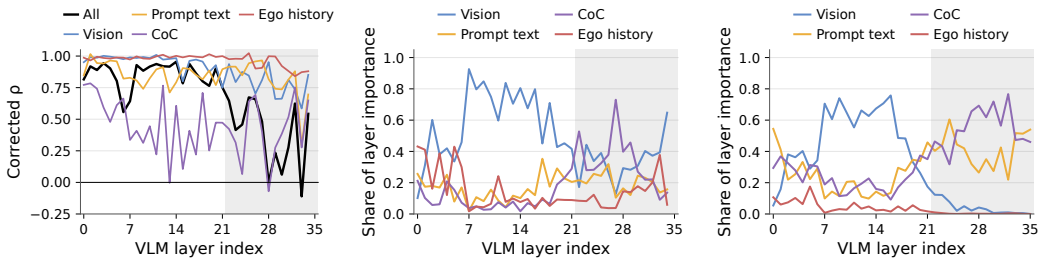
We analyze Alpamayo-1.5-10B (NVIDIA et al., 2025c) to examine how importance is distributed in the VLM backbone under the two objectives and in the action expert across denoising steps. We use 100 calibration clips selected from the PhysicalAI-AV training dataset (NVIDIA et al., 2025c) for Taylor importance computation. Details on calibration dataset sampling are in Appendix A.1.

4.1 Importance Score in VLM Backbone

> **Finding 1.** CE and FM importance rankings diverge in the late VLM layers. Even for the same generated CoC tokens, the two objectives favor different units for reasoning generation and action conditioning.

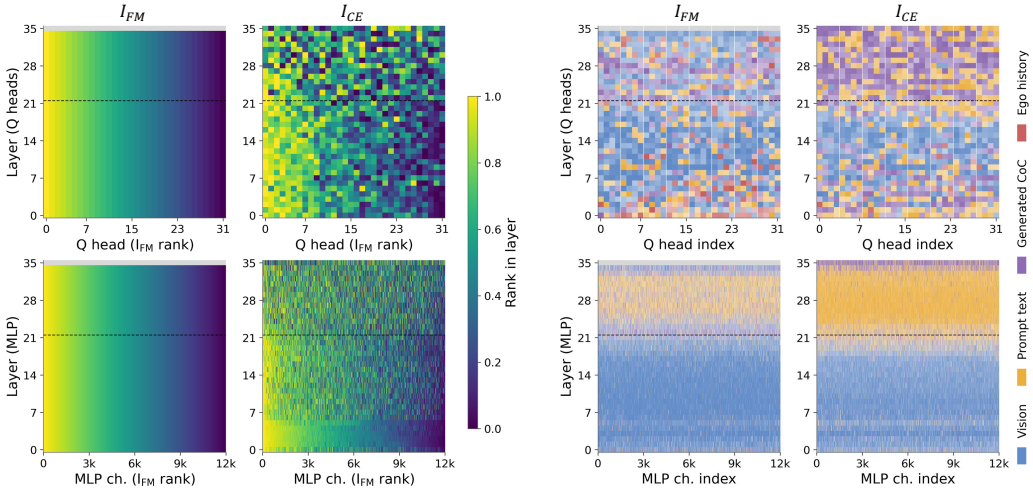
We first ask whether CE and FM importance identify the same VLM units for pruning. Within each layer, we compare the rankings induced by I_{CE} and I_{FM} using Spearman's ρ (Figure 3a). The rankings largely agree through layer 21 but diverge in later layers, suggesting that pruning with either objective alone can remove units important to the other. Figure 4a shows this comparison at the level of individual units. The units of each layer are sorted by their I_{FM} rank, so each row of the I_{FM} panel changes smoothly from the most to the least important unit. For both query heads and MLP channels, the I_{CE} panel departs from this order: units that I_{FM} ranks among the most important often receive low CE ranks, and vice versa. To distinguish disagreement across token types from disagreement at the same tokens, we also compute importance separately for vision, prompt-text, generated-CoC and ego-history tokens by restricting the token-wise sum in the gate gradient (Eq. 2) to each token type before computing the Taylor importance score (Eq. 3). The resulting rankings agree more closely for vision tokens, especially through layer 21, whereas agreement for generated CoC tokens remains low across much of the backbone. Thus, the disagreement is not solely due to the two objectives drawing importance from different token types.

We attribute this pattern to different token-wise importance distributions, as shown in Figure 3b and 3c. Both objectives place substantial importance on processing visual inputs in the early and middle layers, even though they may not rely on identical parameters, as indicated by their differing importance rank distributions. However, in the late layers, we observe a notable difference in that CE importance becomes concentrated on text tokens, while FM importance remains distributed across



(a) Rank correlation (b) I_{FM} by token type (c) I_{CE} by token type

Figure 3: **CE and FM importance differ in the late VLM layers.** (a) Rank correlation between the two objectives using all tokens (black) or one token type at a time (colored). (b, c) Layerwise importance shares for FM and CE, respectively, grouped into vision, prompt text, ego-history and CoC tokens.



(a) Rank within layer (b) Dominant token type

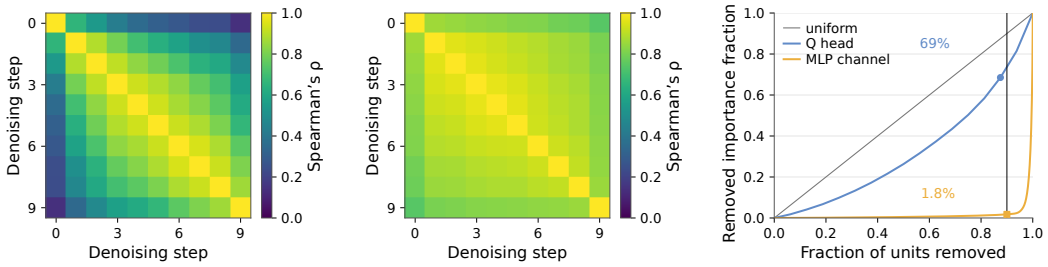
Figure 4: **CE and FM importance differ for both query heads and MLP channels.** Each cell is a query head (top) or an MLP channel (bottom), with layers 0 to 35 from bottom to top. (a) Rank of each unit within its layer, ordered by I_{FM} rank; the closer the I_{CE} panel is to the I_{FM} ramp, the more the two rankings agree. (b) Token type contributing most to each unit's importance; paler colors mean the importance is spread over several types.

vision, text and ego-history tokens. The query-head maps in Figure 4b show this distinction at the level of individual heads: late-layer heads derive most of their CE importance from prompt-text or generated-CoC tokens, while their dominant sources of FM importance also include vision and ego-history tokens.

The persistent disagreement at generated CoC tokens reveals a further distinction between the objectives. The same unit outputs support next-token prediction under CE but provide conditioning information to the action expert through the KV cache under FM (Section 3.1). A unit needed to continue the reasoning sequence therefore need not be equally important for generating actions from that reasoning, and vice versa. CE and FM importance capture these different uses of the backbone, motivating a pruning criterion that accounts for both objectives.

4.2 Importance Score in Action Expert

Finding 2. Compared with query heads, MLP channels have more stable importance rankings across denoising steps and concentrate importance in fewer units.



(a) Q head rank similarity (b) MLP channel rank similarity (c) Cumulative importance

Figure 5: **Denoising stepwise rank stability and importance concentration in the action expert.**

The action expert reuses its weights across denoising steps, so removing a unit affects the entire generation process. We thus examine whether the same units remain important across steps. Let $I_{\mathrm{FM}}(\ell, u, s)$ denote the importance of unit u in layer ℓ, obtained by applying the Taylor criterion (Eq. 3) to the FM loss evaluated at denoising step s. For each unit type, we compute the pairwise Spearman correlations between step-specific rankings and average them over layers (Figure. 5a and 5b). To illustrate, we find that query-head rankings are consistent between nearby steps but diverge as the steps grow farther apart, while MLP-channel rankings are consistent across the steps.

We next examine how importance is distributed among units over the full denoising process. We summarize each unit's importance by averaging its step-specific scores:

$$\bar{I}_{\mathrm{FM}}(\ell, u) = \frac{1}{S} \sum_{s=0}^{S-1} I_{\mathrm{FM}}(\ell, u, s), \tag{4}$$

where $S = 10$ and the bar denotes averaging over denoising steps. Each step contributes equally with weight $1/S$, without additional timestep-dependent weighting. For each unit type, let U_ℓ be the number of units in layer ℓ and let $\pi_\ell(j)$ denote the index of the unit with the j-th smallest mean score. Thus, π_ℓ is a permutation of the unit indices satisfying

$$\bar{I}_{\mathrm{FM}}\big(\ell, \pi_\ell(1)\big) \le \cdots \le \bar{I}_{\mathrm{FM}}\big(\ell, \pi_\ell(U_\ell)\big).$$

We define the cumulative importance fraction $C(r)$ as the fraction of total importance assigned to the lowest-scoring fraction r of units, averaged over layers:

$$C(r) = \frac{1}{L} \sum_{\ell=0}^{L-1} \frac{\sum_{j=1}^{\lfloor rU_\ell \rfloor} \bar{I}_{\mathrm{FM}}\big(\ell, \pi_\ell(j)\big)}{\sum_{u=1}^{U_\ell} \bar{I}_{\mathrm{FM}}(\ell, u)}, \tag{5}$$

where L is the number of expert layers, $r \in [0, 1]$ is the fraction of units considered for removal, and $\lfloor rU_\ell \rfloor$ is the corresponding integer count. A small $C(r)$ means that removing these units discards little of the measured importance. Figure 5c shows that a large fraction of MLP channels accounts for little of the total importance, while query-head importance is distributed more broadly.

Together, these findings support targeting MLP channels for expert pruning: their stable rankings make a fixed selection of channels suitable across denoising steps, and their concentrated importance allows many low-scoring channels to be removed. Query heads offer less scope for this approach because their relative importance changes across steps and fewer heads have negligible scores.

5 Method

Our approach uses CE and FM importance jointly to select VLM units and step-averaged FM importance to select action expert MLP channels. In both components, selection is performed within each layer and unit type under a specified pruning budget. Applying the same pruning budget in every layer keeps comparisons focused on the scoring rule rather than on budget allocation.

5.1 VLM Pruning with CE and FM Importance

Section 4.1 shows that CE and FM importance can favor different VLM units. To account for both objectives without requiring their scores to have comparable scales, we combine their within-layer

Table 1: **Open-loop evaluation on PhysicalAI-AV (NVIDIA et al., 2025c).**

Method	Expert Pruning	Pruning Ratio	Test minADE@6 (m) ↓	Test minFDE@6 (m) ↓	OOD-val minADE@6 (m) ↓	OOD-val minFDE@6 (m) ↓
Baseline	✗	0%	0.842	2.267	1.000	2.496
Wanda	✗	24%	2.975	8.269	2.790	6.884
LLM-Pruner	✗	24%	1.451	3.889	1.662	4.153
Týr-the-Pruner	✗	24%	**0.950**	2.570	1.177	2.990
FM-only	✗	24%	1.008	2.720	1.178	2.960
CE-only	✗	24%	1.584	4.500	1.546	4.010
MoOP (Ours)	✗	24%	**0.950**	2.546	1.119	2.851
MoOP (Ours)	✓	39.4%	0.951	**2.539**	**1.118**	**2.844**

ranks. We apply the following rule separately to query heads and MLP channels, using the Taylor importance scores from Eq. 3.

Let U_ℓ be the number of units of the selected type in layer ℓ. Following the sorted-index notation in Section 4.2, let $\pi_\ell^o(j)$ denote the index of the unit with the j-th smallest importance score under objective $o \in \{\mathrm{CE}, \mathrm{FM}\}$. We define the corresponding ranks by

$$\begin{aligned} I_o\big(\ell, \pi_\ell^o(1)\big) \le \cdots \le I_o\big(\ell, \pi_\ell^o(U_\ell)\big), \\ \mathrm{rank}^o\big(\ell, \pi_\ell^o(j)\big) = j, \qquad j = 1, \ldots, U_\ell. \end{aligned} \tag{6}$$

Ties are broken by increasing unit index. The least important unit has rank 1, and the most important has rank U_ℓ. We then define the combined rank score $R(\ell, u)$ as:

$$R(\ell, u) = \max\Big(\mathrm{rank}^{\mathrm{CE}}(\ell, u), \mathrm{rank}^{\mathrm{FM}}(\ell, u)\Big). \tag{7}$$

This maximum gives a unit high priority for retention if it is important to either objective; low maximum scores identify units that rank low under both. Since taking the maximum can give multiple units the same score, we note that R is a pruning score rather than a unique ordering.

For a target pruning ratio r, we sort units by increasing R and remove the first $k_\ell(r)$ units:

$$\begin{aligned} \sigma_\ell = \mathrm{argsort}_{u \in \{1, \ldots, U_\ell\}} R(\ell, u), \qquad k_\ell(r) = \left\lfloor r U_\ell + \tfrac{1}{2} \right\rfloor, \\ \mathcal{P}_\ell(r) = \{\sigma_\ell(j) : 1 \le j \le k_\ell(r)\}. \end{aligned} \tag{8}$$

Here, $\sigma_\ell(j)$ is the index of the unit at sorted position j, with score ties again broken by increasing unit index, and $\mathcal{P}_\ell(r)$ is the set of units to remove. All units outside $\mathcal{P}_\ell(r)$ are retained.

5.2 Action Expert MLP Pruning

In the action expert, we prune intermediate MLP channels using FM importance. The CoC prediction loss depends on the VLM but not on the action expert parameters (Eq. 1), so expert pruning requires only the FM criterion. Because the expert reuses its weights throughout denoising, we score each channel using the step-averaged importance $\bar{I}_{\mathrm{FM}}(\ell, u)$ in Eq. 4. This score summarizes the estimated FM-loss sensitivity to removing a channel across the full denoising process.

The stable MLP importance rankings in Section 4.2 support retaining a fixed set of channels for all steps. The concentration of MLP importance motivates evaluating high channel-removal ratios. Within each expert layer, we first average the step-specific scores and then rank the channels once, applying Eq. 6 to $\bar{I}_{\mathrm{FM}}$ with $o = \mathrm{FM}$. We remove the lowest-ranked channels using the same selection rule as Eq. 8, with $R(\ell, u) = \mathrm{rank}^{\mathrm{FM}}(\ell, u)$. Expert attention is retained because query-head importance is more broadly distributed and its rankings vary across steps. Pruning only the intermediate MLP channels preserves the residual-stream dimension and the layerwise KV-conditioning interface with the VLM, while reducing the MLP width at every denoising step.

Table 2: **Closed-loop evaluation in AlpaSim (NVIDIA et al., 2025b).**

Method	Expert Pruning	Origin150 Scene Score ↑	Origin150 Offroad ↓	Origin150 At-fault Coll. ↓	Hard100 Scene Score ↑	Hard100 Offroad ↓	Hard100 At-fault Coll. ↓
Baseline	✗	0.750	0.070	0.040	0.510	0.265	0.030
Wanda	✗	0.621	0.093	0.147	0.406	**0.140**	0.060
LLM-Pruner	✗	<u>0.811</u>	**0.037**	0.057	0.575	<u>0.160</u>	0.035
Týr-the-Pruner	✗	0.786	0.087	<u>0.030</u>	0.586	0.215	**0.020**
FM-only	✗	0.783	0.087	0.033	0.546	0.260	<u>0.025</u>
CE-only	✗	0.660	0.090	0.073	0.416	0.200	0.040
MoOP(Ours)	✗	**0.828**	<u>0.067</u>	**0.023**	<u>0.595</u>	0.190	0.030
MoOP(Ours)	✓	0.793	0.083	<u>0.030</u>	**0.600**	0.208	**0.020**



Figure 6: **Qualitative comparison of closed-loop evaluation.** Top shows the driving trajectory, while the bottom shows front and side views from the vehicle.

6 Experiments

6.1 Experimental Setup

Implementation details. We prune Alpamayo-1.5-10B (NVIDIA et al., 2025c) using 100 calibration clips as illustrated in Section 4, without post-pruning fine-tuning. VLM pruning uses $r = 0.4$ in all 36 layers, removing 13 of 32 query heads and approximately 40% of MLP channels, for 24.0% total parameter reduction. For the expert pruning process, we remove 93.75% of MLP channels in each of its 36 layers, retaining 516 of $8{,}256$ channels and all attention heads, for 39.4% total reduction. Details of the baseline methods are provided in Appendix A.2.

Evaluation protocol. Open-loop evaluation uses PhysicalAI-AV (NVIDIA et al., 2025c) Test500 and OOD-Val262, reporting the minimum average and final displacement errors over six 6.4 s trajectories (minADE@6 and minFDE@6, in meters). Closed-loop evaluation uses AlpaSim (NVIDIA et al., 2025b) Origin150 and Hard100, reporting route-progress scene score with off-road and at-fault collision penalties, and the rates of these failures. Driving VQA uses LingoQA (Marcu et al., 2024), reporting the percentage of answers accepted by Lingo-Judge and the mean NLL of generated answer tokens. Higher scene score and Lingo-Judge score, and lower errors, failure rates, and NLL, are better. Dataset construction and full protocols are in Appendix A.3.

6.2 Main Results

Open- and closed-loop driving. At 24.0% parameter reduction, MoOP outperforms CE-only and FM-only on all four open-loop metrics and both closed-loop scene scores (Tables 1 and 2). On Hard100, its scene score exceeds the two variants by 0.179 and 0.049, respectively. Compared with Týr-the-Pruner, it lowers OOD-Val262 minADE@6 and minFDE@6 by 4.9% and 4.6% and achieves the highest scene scores among methods at this budget. Figure 6 illustrates the corresponding driving behavior. Joint importance thus benefits both trajectory prediction and driving under closed-loop feedback.

Table 3: **Ablation for action expert MLP pruning on MoOP.**

Expert MLP Removed	Total Removed	Test500		OOD-val262	
		minADE@6 (m) ↓	minFDE@6 (m) ↓	minADE@6 (m) ↓	minFDE@6 (m) ↓
0.0%	24.0%	0.950	2.546	1.119	2.851
75.0%	36.3%	0.950	2.543	1.118	2.848
87.5%	38.4%	0.950	2.542	1.118	2.846
93.75%	39.4%	0.951	2.539	1.118	2.844

Table 5: **Inference latency and memory usage.**

Model	Expert MLP Removed	Latency				Memory	
		Prefill (ms)	Decode (ms/token)	Denoise (ms)	End-to-end (ms)	Params (B)	Peak (GiB)
Baseline	–	691.3	27.39	212.3	1341.8 (1.000×)	11.08	22.487 (1.000×)
MoOP	0%	515.6	19.53	204.3	1032.4 (1.300×)	8.42	17.448 (1.289×)
	75%	514.4	19.52	160.3	987.0 (1.359×)	7.05	14.899 (1.509×)
	87.5%	513.4	19.52	153.8	979.6 (1.370×)	6.82	14.473 (1.554×)
	93.75%	514.1	19.54	150.9	977.8 (1.372×)	6.71	14.261 (1.577×)

Driving VQA evaluation. MoOP exceeds CE-only and FM-only by 38.6 and 31.8 Lingo-Judge points, and the strongest external baseline, LLM-Pruner, by 24.0 points (Table 4). It remains 4.4 points below the dense model. MoOP also achieves a mean NLL that is 0.061 lower than that of CE-only.These gains support considering both objectives to preserve driving-scene understanding.

Action expert pruning. Additional expert pruning changes open-loop errors by at most 0.007 m across tested ratios (Table 3); at 93.75% MLP removal, scene score changes by −0.035 on Origin150 and +0.005 on Hard100 (Table 2). This adds 15.4 percentage points of parameter reduction while largely preserving driving performance.

Table 4: **Driving VQA evaluation on LingoQA (Marcu et al., 2024).**

Method	LingoQA ↑	NLL ↓
Baseline	73.2	0.175
Wanda	9.2	2.183
LLM-Pruner	<u>44.8</u>	0.462
Týr-the-Pruner	34.2	0.516
FM-only	37.0	0.586
CE-only	30.2	<u>0.313</u>
MoOP (Ours)	**68.8**	**0.252**

6.3 Cost Analysis

Combined pruning yields a 1.37× speedup and 36.6% lower peak memory than the dense model (Table 5). Expert pruning further reduces denoising latency by 53.4 ms relative to VLM-only pruning, with nearly unchanged prefill and decoding latency. Compressing the expert therefore reduces the repeated cost of action generation beyond the savings from VLM pruning.

7 Conclusion

We presented MoOP, a structured pruning method that accounts for the roles of the VLM and action expert in reasoning VLAs. Controlled comparisons show that using both reasoning and trajectory importance better preserves driving QA, trajectory prediction, and closed-loop scene scores than using either objective alone. For the action expert, we exploit the stability and concentration of MLP importance across denoising steps to identify channels for removal. On Alpamayo, jointly pruning the VLM and action expert reduces total parameters by 39.4% while retaining competitive open- and closed-loop driving performance. The model achieves a 1.37× end-to-end inference speedup and 36.6% lower peak memory than the dense model, without post-pruning fine-tuning.

AI USE STATEMENT

We used generative AI tools for the following tasks with required disclosure: implementing methods (importance computation, pruning, analysis code and the adaptation of baselines), selecting and reformatting data subsets, translating author-written drafts into English, and limited feedback on the design of our analyses. We did not use them to generate synthetic data, develop theoretical frameworks or propose hypotheses; mathematical claims and proofs are not applicable to this work. We also used them for figures, other code, the code release, drafting and editing parts of the paper, literature search and summarization, and reference formatting. The research question and the proposed method were conceived by the authors. All AI-assisted text was reviewed and revised by the authors, all reported numbers were checked against the raw experiment outputs, and AI-written code was verified by inspecting the implementation and checking that its results were consistent. We take full responsibility for the content of this paper.

REPRODUCIBILITY STATEMENT

Our pruning involves no training or fine-tuning and is fully determined by the importance scores, so our results can be reproduced from the scores, the selection rules and the evaluation protocol. Section 3.2 defines the Taylor importance, and Section 4 and Appendix A.1 describe the calibration clips used to compute it. Section 5 specifies how VLM units and expert MLP channels are selected, and Section 6.1 gives the per-layer budgets, the baselines and their calibration, the benchmarks and metrics, and the hardware used for latency measurements. Appendices A.3.1 and A.3.2 describe how the open-loop clips and closed-loop scenes were selected. All models, datasets and simulators we use are publicly available.

REFERENCES

Joshua Ainslie, James Lee-Thorp, Michiel De Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 4895–4901, 2023.

Harisankar Babu, Benjamin Coors, Christopher Lang, Hendrik Berkemeyer, Tamim Asfour, and Simon Föll. Depth-wise probing and pruning of the planning token in a driving vision-language-action model. *arXiv preprint arXiv:2608.07361*, 2026.

Aritra Dutta and Somak Aditya. Mucrasp: Multimodal chain-of-thought reasoning aware structured pruning. *arXiv preprint arXiv:2605.25842*, 2026.

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 *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 24823–24834. IEEE, 2025.

Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

B. Hassibi, D.G. Stork, and G.J. Wolff. Optimal brain surgeon and general network pruning. In *IEEE International Conference on Neural Networks*, pp. 293–299 vol.1, 1993. doi: 10.1109/ICNN.1993.298572.

Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989.

Guanchen Li, Yixing Xu, Zeping Li, Ji Liu, Xuanwu Yin, Dong Li, and Emad Barsoum. Týr-the-pruner: Structural pruning LLMs via global sparsity distribution optimization. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=rAuRLePL2R.

Zekai Li, Yihao Liang, Hongfei Zhang, Jian Chen, Yesheng Liang, and Zhijian Liu. Flash-drive: Flash vision-language-action inference for autonomous driving. *arXiv preprint arXiv:2608.12932*, 2026.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Ryan Lucas, Kayhan Behdin, Zhipeng Wang, Qingquan Song, Shao Tang, and Rahul Mazumder. Reasoning models can be accurately pruned via chain-of-thought reconstruction. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=tyGfwG6xTh.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. Llm-pruner: On the structural pruning of large language models. *Advances in neural information processing systems*, 36:21702–21720, 2023.

Ana-Maria Marcu, Long Chen, Jan Hünermann, Alice Karnsund, Benoit Hanotte, Prajwal Chidananda, Saurabh Nair, Vijay Badrinarayanan, Alex Kendall, Jamie Shotton, et al. Lingoqa: Visual question answering for autonomous driving. In *European Conference on Computer Vision*, pp. 252–269. Springer, 2024.

Pavlo Molchanov, Stephen Tyree, Tero Karras, Timo Aila, and Jan Kautz. Pruning convolutional neural networks for resource efficient inference, 2017. URL https://arxiv.org/abs/1611.06440.

Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. Importance estimation for neural network pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

Saurav Muralidharan, Sharath Turuvekere Sreenivas, Raviraj Joshi, Marcin Chochowski, Mostofa Patwary, Mohammad Shoeybi, Bryan Catanzaro, Jan Kautz, and Pavlo Molchanov. Compact language models via pruning and knowledge distillation. *Advances in Neural Information Processing Systems*, 37:41076–41102, 2024.

Gia-Binh Nguyen, Trong-Bao Ho, Thien-Loc Ha, Khoa Vo, Philip Lund Møller, Quang T Nguyen, Long Dinh, Tung M Luu, Tuan Dam, Vu Duong, et al. Finetuning vision-language-action models requires fewer layers than you think. *arXiv preprint arXiv:2606.20246*, 2026.

NVIDIA, Alisson Azzolini, Hannah Brandon, Prithvijit Chattopadhyay, Huayu Chen, Jinju Chu, Yin Cui, Jenna Diamond, Yifan Ding, Francesco Ferroni, Rama Govindaraju, Jinwei Gu, Siddharth Gururani, Imad El Hanafi, Zekun Hao, Jacob Huffman, Jingyi Jin, Brendan Johnson, Rizwan Khan, George Kurian, Elena Lantz, Nayeon Lee, Zhaoshuo Li, Xuan Li, Tsung-Yi Lin, Yen-Chen Lin, Ming-Yu Liu, Andrew Mathau, Yun Ni, Lindsey Pavao, Wei Ping, David W. Romero, Misha Smelyanskiy, Shuran Song, Lyne Tchapmi, Andrew Z. Wang, Boxin Wang, Haoxiang Wang, Fangyin Wei, Jiashu Xu, Yao Xu, Xiaodong Yang, Zhuolin Yang, Xiaohui Zeng, and Zhe Zhang. Cosmos-reason1: From physical common sense to embodied reasoning. *arXiv preprint arXiv:2503.15558*, 2025a.

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 2025b. 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*, 2025c.

Mingjie Sun, Zhuang Liu, Anna Bair, and Zico Kolter. A simple and effective pruning approach for large language models. In *International Conference on Learning Representations*, volume 2024, pp. 4942–4964, 2024.

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 *CoRL*, pp. 4698–4726, 2024. URL https://proceedings.mlr.press/v270/tian25c.html.

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 Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=_VjQlMeSB_J.

Yaozhi Wen, Jialong Guo, Zhenliang Ni, Han Shu, and Xinghao Chen. Slimvlm: Sensitivity-aware dynamic structured pruning with adaptive visual token selection for efficient vision-language models. *arXiv preprint arXiv:2608.03580*, 2026.

Yuhao Xu, Yantai Yang, Zhenyang Fan, Yufan Liu, Yuming Li, Bing Li, and Zhipeng Zhang. Qvla: Not all channels are equal in vision-language-action model's quantization. *arXiv preprint arXiv:2602.03782*, 2026.

Yantai Yang, Yuhao Wang, Zichen Wen, Luo Zhongwei, Chang Zou, Zhipeng Zhang, Chuan Wen, and Linfeng Zhang. EfficientVLA: Training-free acceleration and compression for vision-language-action models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=SELYlDHZk2.

Zhenlong Yuan, Chengxuan Qian, Jing Tang, Rui Chen, Zijian Song, Lei Sun, Xiangxiang Chu, Yujun Cai, Dapeng Zhang, and Shuo Li. Autodrive-r²: Incentivizing reasoning and self-reflection capacity for VLA model in autonomous driving. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=KVWaCzJrrq.

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 *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=28qUA2bSe5.

Exploring What to Preserve for Reasoning and Action in Structured Pruning of Driving VLAs

Appendix

OVERVIEW

This appendix describes calibration data selection, compared methods, and the sampling and evaluation protocols for each benchmark. Qualitative results complement the quantitative evaluation in the main paper. The content is organized as follows:

- **Sec. A**: Experimental details.
  - **Sec. A.1**: Construction of the *Calib100* calibration set from the training split by greedy matching of collection-metadata distributions.
  - **Sec. A.2**: External pruning baselines, single-objective variants, and expert-pruning settings.
  - **Sec. A.3**: Evaluation protocols and data sampling for open-loop trajectory prediction, closed-loop driving, and LingoQA.
- **Sec. B**: Qualitative results.
  - **Sec. B.1**: Qualitative comparison on LingoQA driving question answering, grouped by signal state, road users and hazards, and actions at zebra crossings.
  - **Sec. B.2**: Qualitative comparison of open-loop trajectory predictions and the reasoning generated with them, between the Baseline and MoOP.
  - **Sec. B.3**: Qualitative comparison of closed-loop rollouts in AlpaSim, covering scenes where other pruning methods cause at-fault collisions or leave the road.
  - **Sec. B.4**: Qualitative discussion of multi-answer cases in closed-loop evaluation, where a scene with more than one valid route is scored against a single recorded path.

A EXPERIMENTAL DETAILS

A.1 CALIBRATION SET SELECTION

We construct Calib100 from the official training split of the PhysicalAI-AV dataset (NVIDIA et al., 2025c). Of the 153,625 training clips, we exclude 1,450 clips listed in the dataset's OOD reasoning annotations, leaving 152,175 candidates. We select 100 clips using seeded greedy matching of collection-metadata distributions. The target distributions are computed over the full training split before these exclusions.

The procedure matches the marginal distributions of six categorical attributes: country, platform class, time of day, season, month, and radar configuration, with respective weights of 4.0, 2.0, 2.0, 1.5, 1.0, and 1.0. For a selected set $\mathcal{S}$ and a reference split $\mathcal{R}$, we measure their discrepancy as

$$J_{\mathcal{R}}(\mathcal{S}) = \frac{1}{\sum_{a\in\mathcal{A}} w_a} \sum_{a\in\mathcal{A}} w_a \sum_{v\in\mathcal{V}_a} \left|\widehat{p}_{a,v}(\mathcal{S}) - p_{a,v}(\mathcal{R})\right|, \tag{9}$$

where $\mathcal{A}$ is the attribute set, $\mathcal{V}_a$ is the set of categories for attribute a, and $\widehat{p}_{a,v}(\mathcal{S})$ and $p_{a,v}(\mathcal{R})$ are the corresponding category proportions. Starting from an empty set, we repeatedly add the unselected candidate that minimizes the resulting discrepancy. Ties are broken uniformly at random using a NumPy (Harris et al., 2020) random generator initialized with seed 42. Calib100 consists of the first 100 clips in this greedy sequence.

Time of day is classified as daytime for $6 \leq \text{hour} < 18$ and nighttime otherwise. Season is derived from calendar month: December–February for winter, March–May for spring, June–August for summer, and September–November for fall, using the same convention for all countries. Missing values form a separate category for each attribute. The selected set attains a weighted discrepancy of 0.0306 under Eq. 9. Each clip uses a fixed window start of $t_0 = 5.1$ s. Selection uses only collection metadata; driving actions and model outputs are not used to choose clips.

A.2 Compared Methods

All compared models start from Alpamayo-1.5-10B. The dense model, denoted Baseline, is evaluated without pruning. For the main VLM comparison, all pruning methods remove 24.0% of the full model's parameters and leave the action expert intact. We evaluate every model without post-pruning fine-tuning, so recovery training does not account for differences between methods.

LLM-Pruner. LLM-Pruner (Ma et al., 2023) identifies groups of dependent structures and estimates their importance using gradient information. Removing these groups together preserves the connectivity of the remaining network. We apply this structural pruning baseline to the VLM and omit the optional LoRA recovery stage used in the original method.

Wanda. Wanda (Sun et al., 2024) scores a weight by the product of its absolute magnitude and the norm of its input activations. The original method removes low-scoring weights to produce unstructured sparsity. In our comparison, we use Wanda's importance criterion to select structured VLM units, enabling comparison with methods that physically remove query heads and MLP channels.

Týr-the-Pruner. Týr-the-Pruner (Li et al., 2025) optimizes the distribution of sparsity across layers under an overall parameter budget. It constructs a supernet of locally pruned candidates at multiple sparsity levels and uses a coarse-to-fine search to select their combination. This provides a comparison with global sparsity allocation, whereas MoOP uses a uniform per-layer budget and combines importance from two objectives. The total parameter reduction is matched in our VLM comparison.

Single-objective variants. CE-only and FM-only isolate the effect of the pruning objective. They use I_{CE} and I_{FM}, respectively, to rank units within each layer and unit type, then remove the lowest-ranked units. Both variants match MoOP in the Taylor criterion, calibration set, pruning targets, and per-layer removal counts. They therefore test whether either objective alone suffices to preserve the VLM's reasoning and action-conditioning roles.

MoOP and additional expert pruning. MoOP combines CE and FM ranks using the maximum-rank rule in Section 5.1. We separately assess additional expert MLP pruning on top of this VLM, using step-averaged FM importance as described in Section 5.2. The expert ablation removes 75%, 87.5%, or 93.75% of MLP channels while retaining expert attention (Table 3). The main configuration uses 93.75% expert MLP removal, reaching 39.4% total parameter reduction. Results with and without expert pruning are distinguished in the main tables.

A.3 Evaluation Protocols and Data Sampling

We evaluate trajectory prediction, closed-loop driving, and driving question answering. Within each benchmark, all models receive the same evaluation examples and use the same evaluation settings. Data selection and metric computation are described together below for each benchmark.

A.3.1 Open-Loop Trajectory Prediction

Test500. We select 500 clips from the official test split using the same metadata attributes, weights, greedy procedure, and seed as Calib100. The reference distributions are computed over all 61,599 test clips. We exclude 12 test clips belonging to an earlier 50-clip calibration set, leaving 61,587 candidates. The OOD exclusion list removes no additional clips from this split. Test500 consists of the first 500 clips in the greedy sequence and attains a weighted discrepancy of 0.0055 relative to the full test split. As in calibration, each clip uses $t_0 = 5.1$ s. Test500 and Calib100 are clip-disjoint.

OOD-Val262. For out-of-distribution evaluation, we use all 262 clips in the validation split provided by the PhysicalAI-AV (NVIDIA et al., 2025c) OOD sample, without additional subsampling. OOD-Val262 is clip-disjoint from both Calib100 and Test500.

Evaluation protocol and metrics. For each clip, the model generates six candidate ego trajectories over 6.4 s from the recorded driving context. We compare these predictions with the logged

future trajectory without executing them in a simulator. Let $\hat{\mathbf{p}}_{i,k,t} \in \mathbb{R}^2$ and $\mathbf{p}_{i,t}$ denote the predicted and ground-truth positions in the same coordinate frame, where i indexes clips, k indexes the six candidates, and t indexes H future timestamps. For N clips, we report

$$\begin{aligned} \text{minADE@6} &= \frac{1}{N}\sum_{i=1}^{N} \min_{1\le k\le 6} \frac{1}{H}\sum_{t=1}^{H} \left\|\hat{\mathbf{p}}_{i,k,t} - \mathbf{p}_{i,t}\right\|_2, \\ \text{minFDE@6} &= \frac{1}{N}\sum_{i=1}^{N} \min_{1\le k\le 6} \left\|\hat{\mathbf{p}}_{i,k,H} - \mathbf{p}_{i,H}\right\|_2. \end{aligned} \tag{10}$$

The minima are taken independently for the two metrics before averaging over clips, so the selected candidates may differ. Both errors are measured in meters, and lower values indicate closer agreement with the recorded trajectory. We use $N = 500$ for Test500 and $N = 262$ for OOD-Val262.

A.3.2 Closed-Loop Driving

Both closed-loop evaluation sets are drawn from the 913 scenes in AlpaSim's `public_2601` release (NVIDIA et al., 2025b). Scene artifacts are pinned by UUID to this release to avoid substituting renders from later releases.

Origin150. Origin150 contains 150 scenes randomly sampled from the parent suite.

Hard100. We construct a separate set of 100 scenes using a difficulty score derived from ground-truth ego motion. From each scene's ego pose track sampled at 10 Hz, we compute the mean speed $\bar{v}_i$ and the accumulated absolute yaw change $\sum_t |\Delta\psi_{i,t}|$. The score for scene i is

$$h_i = z_{\mathcal{P}}(\bar{v}_i) + z_{\mathcal{P}}\left(\sum_t |\Delta\psi_{i,t}|\right), \tag{11}$$

where $z_{\mathcal{P}}$ standardizes each feature using its mean and standard deviation over the full parent suite $\mathcal{P}$ of 913 scenes. No model output enters this score.

We exclude all Origin150 scenes and any scene whose ground-truth ego path is shorter than 5 m. Among the remaining scenes, we retain the 100 with the highest scores. Hard100 is therefore disjoint from Origin150, and the two sets together cover 250 scenes. Hard100 uses two rollouts per scene, yielding 200 rollouts. Its selection criterion emphasizes speed and turning demands and does not explicitly stratify scenes by collision risk.

Evaluation protocol. In AlpaSim, the model repeatedly predicts actions from simulated observations, and the executed actions update the ego state in a scene reconstructed from recorded logs. Each model is evaluated on the same scenes with two rollouts per scene, giving 300 rollouts for Origin150 and 200 for Hard100. Metrics are averaged separately within each set; because every scene has two rollouts, scenes receive equal weight.

Metrics. We report scene score, off-road rate, and at-fault collision rate using AlpaSim's evaluator (NVIDIA et al., 2025b). Let b_j and c_j be binary indicators of any off-road event and any at-fault collision in rollout j, respectively. Multiple occurrences within a rollout count once. AlpaSim's at-fault criterion includes front and lateral impacts and excludes rear collisions. Let $q_j \in [0, 1]$ be the normalized route-progress score computed with AlpaSim's progress normalization and saturation rules. For M rollouts, the metrics are

$$\begin{aligned} \text{SceneScore} &= \frac{1}{M}\sum_{j=1}^{M} q_j(1-b_j)(1-c_j), \\ \text{OffRoadRate} &= \frac{1}{M}\sum_{j=1}^{M} b_j, \qquad \text{AtFaultCollisionRate} = \frac{1}{M}\sum_{j=1}^{M} c_j. \end{aligned} \tag{12}$$

An off-road event or at-fault collision sets the corresponding rollout's scene score to zero. All metrics lie on a 0–1 scale; higher scene score and lower failure rates are better. Route progress is measured against the recorded path, which can penalize a different but valid route; Appendix B.4 illustrates such cases.

A.3.3 Driving Question Answering on LingoQA

Evaluation data and inputs. We use the full official LingoQA evaluation split (Marcu et al., 2024): 500 questions from 100 segments, with five questions per segment and two human reference answers per question. No questions are subsampled or filtered. Each segment provides five front-camera frames indexed 0–4. We use the last four (1–4), closest to the question time, to match Alpamayo-1.5's four-frame input per camera. All four are labelled as the front camera, and the question is passed verbatim through the model's native VQA template, without additional style instructions or ego-trajectory inputs.

Evaluation protocol and metrics. Evaluation is zero-shot, with no training on LingoQA. Answers are decoded greedily with a maximum of 128 tokens. We use the official Lingo-Judge checkpoint, take the higher of the two reference-conditioned logits, and count an answer as correct when this logit exceeds zero. The reported score is the percentage of correct answers over all 500 questions. We also report the mean negative log-likelihood (NLL) of generated answer tokens, conditioned on the input and preceding tokens. NLL measures confidence in the model's own answers; correctness is assessed by Lingo-Judge.

B Qualitative Results

This section shows qualitative examples from the three evaluation protocols of the main paper: driving question answering on LingoQA (§B.1), open-loop trajectory prediction (§B.2), and closed-loop driving in AlpaSim (§B.3). §B.4 closes with a small class of multi-answer scenes, in which the closed-loop score can under-rate safe driving.

All compressed models are pruned from the same Alpamayo 1.5-10B model (NVIDIA et al., 2025c), shown as Baseline. Wanda (Sun et al., 2024), Týr-the-Pruner (Li et al., 2025), and LLM-Pruner (Ma et al., 2023) are the external pruning baselines. CE-only and FM-only are single-objective variants of our criterion that score units with only the reasoning cross-entropy or only the flow-matching trajectory loss. MoOP is our dual-objective method.

B.1 Driving Question Answering on LingoQA

Each panel shows four frames of the input clip, the question, the two reference answers, and one answer per model. Each answer is scored by Lingo-Judge (Marcu et al., 2024) on a 0–100 scale, shown at the right of its row. A ✓ and green row marks an answer the judge accepts (score above 50), and a × and red row marks one it rejects. Answers longer than two lines are truncated with an ellipsis. The examples are grouped in pairs by the skill they test: signal state (Fig. 7), road users and hazards (Fig. 8), and actions at zebra crossings (Fig. 9).

(a)



Q: Are there any visible traffic signals at this intersection? If so, what are their color?

Reference: Yes, there are visible traffic signals and they are green. / Yes, green.

Baseline	Yes, there are traffic lights at this intersection, and they are currently green, allowing me to proceed straight ahead.	✓ 94.2
Wanda	Low	✗ 3.1
Týr-the-Pruner	No,	✗ 4.1
LLM-Pruner	[4	✗ 3.8
CE-only	Low	✗ 3.1
FM-only	No, there are no visible traffic lights at this intersection.	✗ 4.7
MoOP(Ours)	The traffic lights is green, but I am decelerating to avoid colliding with the stationary truck ahead.	✓ 88.0

(b)



Q: Are there any traffic signals present? If yes, what color are they displaying?

Reference: Yes, they are amber and then turn green. / Yes. The traffic lights change from amber to green.

Baseline	Yes, there are traffic lights present, and they are currently displaying green, indicating that it is safe for me to proceed straight ahead.	✓ 69.1
Wanda	0, no,405 for<i1518><i1518><i1518><i1518><i1517><i1517><i1517><i1517><i1 …	✗ 3.5
Týr-the-Pruner	There are no traffic signals	✗ 4.5
LLM-Pruner	[0	✗ 3.7
CE-only	Low, there are no traffic signals in view since they are not present in this scene	✗ 3.9
FM-only	Yes, the traffic lights are present, but it is the turning left, and it is the other vehicles in the scene, but it is not in the scene.	✗ 3.9
MoOP(Ours)	Yes, there are traffic lights and they are currently green, indicating that it is safe to proceed straight ahead, since they are green and I …	✓ 73.1

Figure 7: **LingoQA: signal state.** (a) FM-only states that there are no traffic lights. (b) Týr-the-Pruner and CE-only make the same error. MoOP reports the green state in both cases.

(a)



Q: What kind of vehicles are in front of you?

Reference: Cyclist. / There are cyclists in front of me.

Baseline	There is a bicycle ahead of me that I am following.	✓ 93.6
Wanda	45, in with a gap of cyclists ahead, to in7 of line from in7170, to have from in, Scene	✗ 15.0
Týr-the-Pruner	2,1,1,1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1,	✗ 4.1
LLM-Pruner	I am following a car ahead of me that is also turning left, but I am decelerating to maintain a safe distance from the cyclist in front of …	✗ 49.4
CE-only	Low	✗ 8.3
FM-only	I am decelerating to maintain a safe distance ahead of me and the traffic in front of me to safely turn left.	✗ 10.3
MoOP(Ours)	There is a bicycle ahead of me, which is why I am decelerating to maintain a safe distance from the cyclist in front of me.	✓ 90.1

(b)



Q: Can you spot any immediate hazard on the road ahead?

Reference: Yes. The pedestrian is still crossing the road. / Yes.

Baseline	No, I can't spot any immediate hazard on the road ahead, but I am decelerating because there is a pedestrian crossing the zebra crossing …	✗ 23.9
Wanda	Low	✗ 9.6
Týr-the-Pruner	No,	✗ 4.9
LLM-Pruner	The zebra crossing is clear, so there is no obstruction to my vehicle.	✗ 20.8
CE-only	No, there is no immediate hazard on the road ahead, as the zebra crossing is clear for our vehicle to turn right after the pedestrian, …	✗ 12.8
FM-only	No, I can see a slow down ahead, which is a speed of	✗ 5.1
MoOP(Ours)	Yes, I am decelerating because there is a pedestrian crossing the zebra crossing ahead, and I need to let them cross safely.	✓ 85.0

Figure 8: **LingoQA: road users and hazards.** (a) LLM-Pruner answers that it is following a car and mentions the cyclist only as a secondary detail. FM-only describes the ego vehicle's own maneuver and never names the cyclist. (b) The Baseline answers "No" even though its own justification mentions a pedestrian on the zebra crossing. MoOP answers "Yes" and names the pedestrian.

(a)

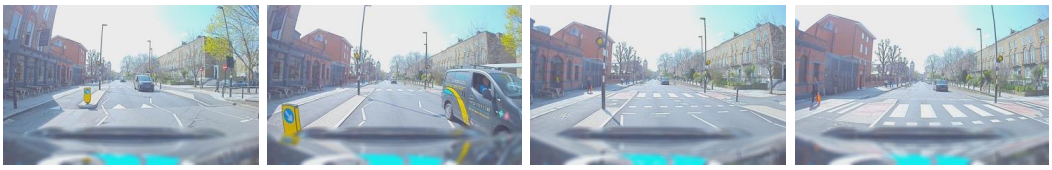


Q: What is the current action and its justification? Answer in the form "action, justification".

Reference: The car slows down as it approaches an intersection and a pedestrian is engaged on the zebra crossing ahead. / Thecar slows down, since it reaches an intersection and a pedestrian is engaged on the zebra crossing.

Baseline	I am continuing driving as the zebra crossing is clear, and there are no pedestrians crossing it at the moment.	✗ 9.1
Wanda	<i1486><i1487><i1484><i1484><i1501><i1501><i1526><i1518><i1518><i151 …	✗ 3.3
Týr-the-Pruner	2,3, 3,	✗ 7.0
LLM-Pruner	I am accelerating because there is a gap in the traffic ahead and it is safe to do so.	✗ 5.3
CE-only	Low	✗ 6.8
FM-only	I am turning right at the intersection since it is safe to do not.	✗ 11.7
MoOP(Ours)	I am decelerating to let the pedestrian crossing the zebra crossing ahead, as there is a pedestrian crossing the road.	✓ 93.5

(b)



Q: What is the current action and its justification? Answer in the form "action, justification".

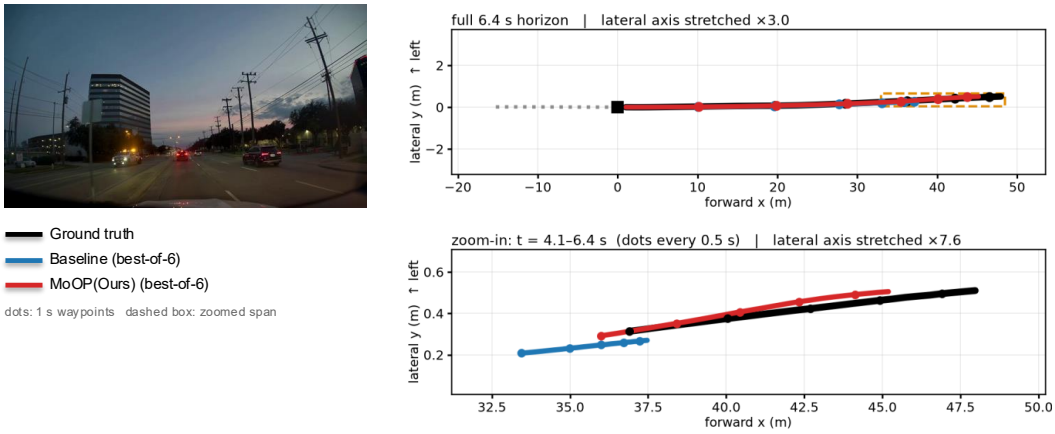
Reference: The car moves slightly right, as it reaches a width restriction traffic island, then continues driving as the zebra crossing is clear. / Slow down slightly for speed bump and zebra crossing, Zebra crossing is clear ahead.

Baseline	I am continuing driving as the zebra crossing is clear, and there are no pedestrians crossing.	✓ 83.9
Wanda	<i1486><i1484><i1484><i1484><i1484><i1484><i1484><i1484><i1484><i148 …	✗ 5.7
Týr-the-Pruner	4	✗ 14.7
LLM-Pruner	[0	✗ 16.5
CE-only	Low	✗ 14.8
FM-only	The current action and its just, as the traffic ahead of me since it is safe and the traffic ahead of me, as it is also stopped and it can be …	✗ 12.8
MoOP(Ours)	I am continuing to drive as the zebra crossing is clear, and there are no pedestrians crossing it is safe to continue driving at my current …	✓ 75.4

Figure 9: **LingoQA: actions at zebra crossings.** (a) The Baseline claims that the crossing is clear and continues. MoOP decelerates for the pedestrian, as in the reference. (b) When the crossing is clear, the Baseline and MoOP both continue. Every other model answers incorrectly: Wanda emits tokens, Týr-the-Pruner and LLM-Pruner return a fragment, CE-only answers “Low”, and FM-only writes an incoherent sentence.

B.2 Open-Loop Trajectory Prediction

Each panel compares the Alpamayo 1.5-10B (NVIDIA et al., 2025c) (Baseline) with MoOP on one clip from OOD-Val262 (Sec. A.3.1). At the top right of each panel, the best of six sampled trajectories for each model is plotted against the ground truth over the 6.4 s horizon. Below that, a zoomed view covers the dashed box. The reasoning each model generated appears beneath. A green row agrees with the reference reasoning and a red row does not. The *speed change* and *cruise* labels are scenario buckets derived only from the speed profile of the ground-truth path. A scene in which the vehicle slows only gradually is labeled *cruise*, such as creeping behind a stopped lead vehicle (Fig. 11a) or easing off behind a truck (Fig. 11b).



Speed change scene · minADE@6: 3.03 m → 0.94 m (gain 2.09 m)

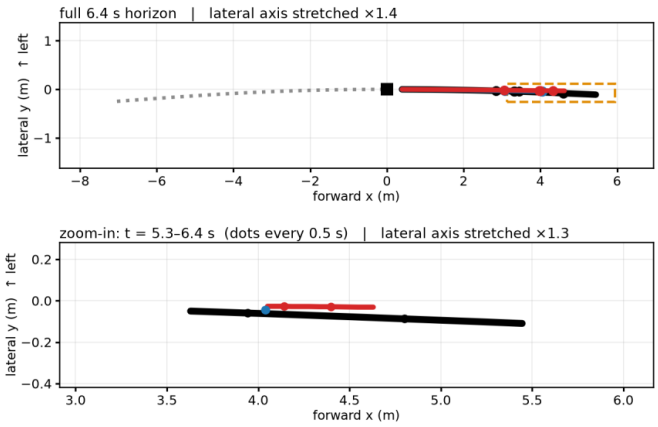
Reference: Decelerate to maintain a safe distance from the lead vehicle ahead.

Baseline minADE 3.03 m	Keep distance to the lead vehicle since it is directly ahead in our lane	✓
MoOP(Ours) minADE 0.94 m	Keep distance to the lead vehicle since it is directly ahead in our lane	✓

Figure 10: **Open loop: deceleration behind a lead vehicle.** Both models state the same, correct intent. The difference lies in the trajectory: the Baseline's best sample ends about 10 m short of the ground-truth endpoint, and MoOP's ends about 3 m short (minADE@6 3.03 m vs. 0.94 m).

(a)





Ground truth
Baseline (best-of-6)
MoOP(Ours) (best-of-6)

dots: 1 s waypoints dashed box: zoomed span

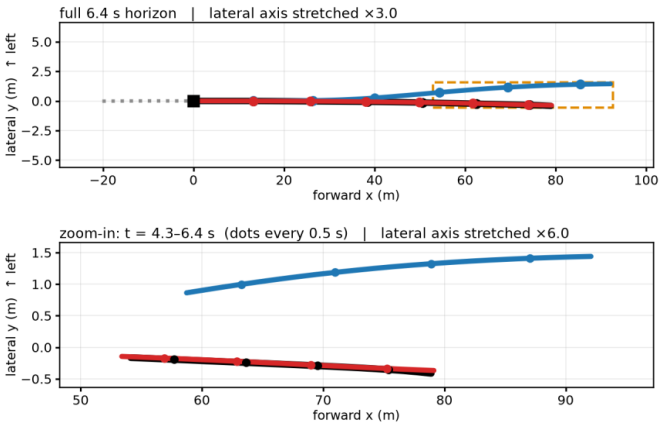
Cruise scene · minADE@6: 0.55 m → 0.47 m (gain 0.07 m)

Reference: Stop behind the lead vehicle in the same lane at the green traffic light.

Baseline minADE 0.55 m	Adapt speed to maintain a safe distance from the lead vehicle since it is stopped ahead	✓
MoOP(Ours) minADE 0.47 m	Keep distance to the lead vehicle since it is stopped ahead	✓

(b)





Ground truth
Baseline (best-of-6)
MoOP(Ours) (best-of-6)

dots: 1 s waypoints dashed box: zoomed span

Cruise scene · minADE@6: 3.72 m → 0.35 m (gain 3.37 m)

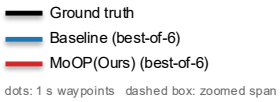
Reference: Decelerate to maintain a safe distance from the lead truck ahead.

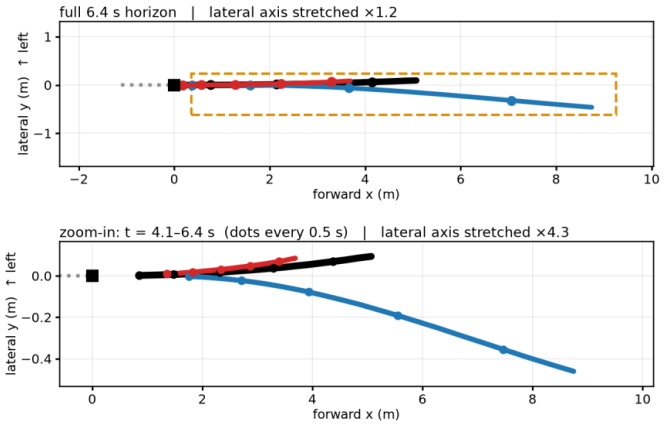
Baseline minADE 3.72 m	Wait for a gap because a car is coming from rear left in our lane blocking an immediate merge to the left	✗
MoOP(Ours) minADE 0.35 m	Keep distance to the lead vehicle since it is directly ahead in our lane	✓

Figure 11: **Open loop: following a lead vehicle.** (a) Both models reason about the stopped lead vehicle, and both best samples stop short of the ground truth. MoOP's stops closer, a small gain (0.55 m vs. 0.47 m), and the Baseline's trajectory is largely hidden beneath MoOP's. (b) The Baseline reasons that it must wait for a gap to merge left, and its best sample drifts about 1.5 m to the left while running ahead of the ground truth. MoOP reasons about keeping its distance to the lead vehicle, which is the truck ahead, and stays on the ground-truth path (3.72 m vs. 0.35 m).

(a)







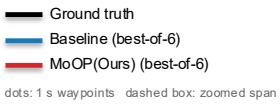
Speed change scene · minADE@6: 0.85 m → 0.32 m (gain 0.53 m)

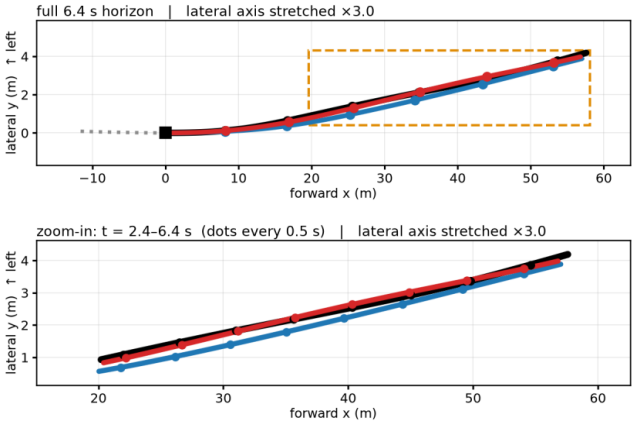
Reference: Stop to yield to the traffic on the main road.

Baseline minADE 0.85 m	Stop to yield to the cross-traffic vehicle since it is crossing our lane ahead	✓
MoOP(Ours) minADE 0.32 m	Stop to yield to the cross-traffic vehicle since it is crossing our lane ahead	✓

(b)







Cruise scene · minADE@6: 0.45 m → 0.20 m (gain 0.25 m)

Reference: Steer left to keep a safe distance from the cyclist on the right.

Baseline minADE 0.45 m	Nudge left due to the cyclist ahead in our lane	✓
MoOP(Ours) minADE 0.20 m	Nudge left due to the cyclist ahead in our lane	✓

Figure 12: **Open loop: yielding and passing.** Both models give identical reasoning in both cases. MoOP's trajectory places the stop (a) and the lateral offset (b) closer to the ground truth (0.85 m vs. 0.32 m and 0.45 m vs. 0.20 m).

B.3 CLOSED-LOOP DRIVING

Each figure includes the Baseline and MoOP, while other methods are shown only if they exhibit an at-fault event in the scene. Detailed behaviors and failure cases are discussed in the individual figure captions.

In the collision figures (all except Fig. 14), the top row shows two front-camera frames from the AlpaSim (NVIDIA et al., 2025b) simulation at the indicated times. Below them is one bird's-eye strip per method, showing:

- ego poses in the method's color (lighter is earlier),
- the plan held at the critical moment (solid),
- the human-driven path (dashed gray), and
- other agents (gray).

A $\times$ marks the ego at the method's critical moment, which is a collision or, for Týr-the-Pruner in Fig. 20, leaving the road. The red outline marks the vehicle that the method struck. In a strip without a collision, it marks a vehicle struck by another method, drawn for reference. Each strip's title gives the rollout's outcome, and the line below it quotes the method's reasoning at the critical moment. Fig. 14 uses a separate layout, described in its caption.

For most cases, closed-loop evaluation faithfully scores the driving performance. In §B.4, we will discuss some minor exceptions we observed, a scene that admits more than one valid route.

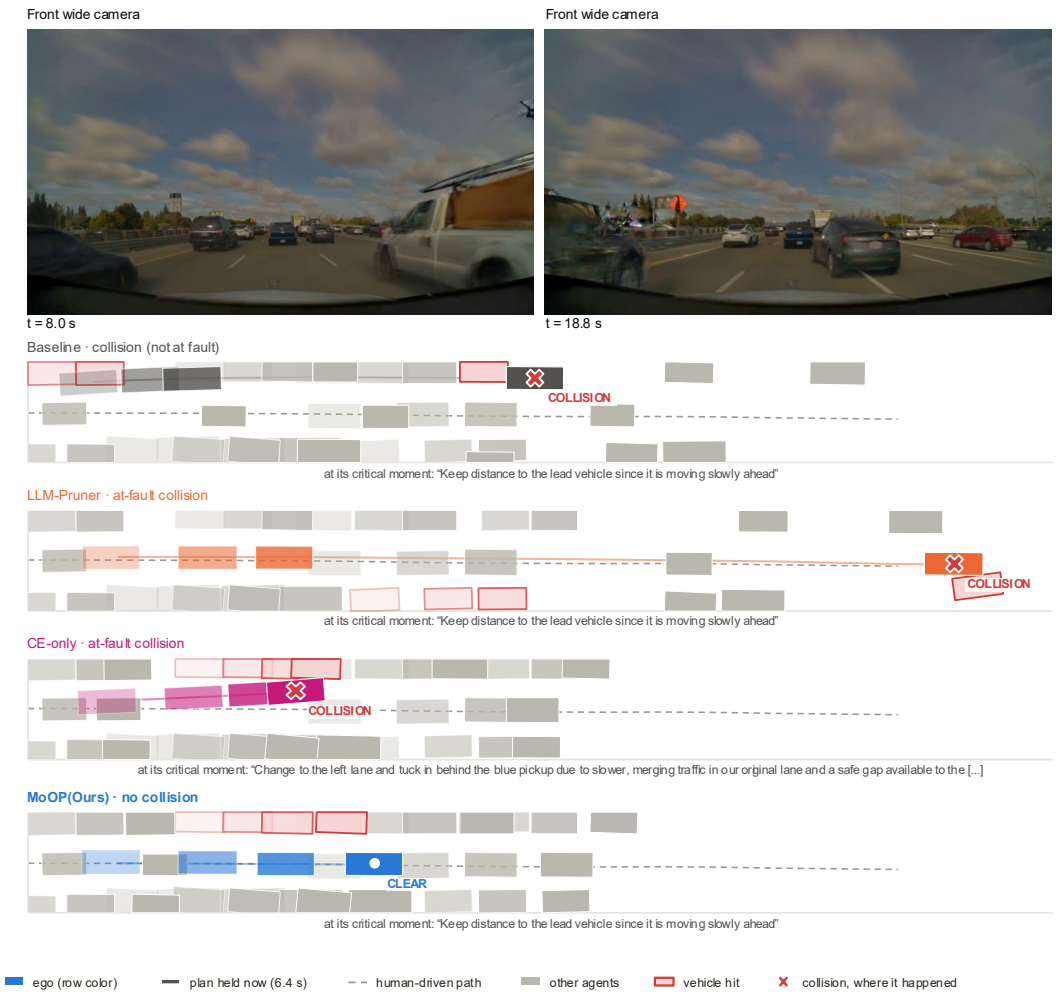


Figure 13: **Closed loop: cut-in.** The Baseline is hit (not at fault), and LLM-Pruner and CE-only have at-fault collisions. LLM-Pruner states the same intent as MoOP, to keep its distance to the slow lead vehicle. CE-only instead proposes a lane change to the left. MoOP keeps its distance and has no collision.

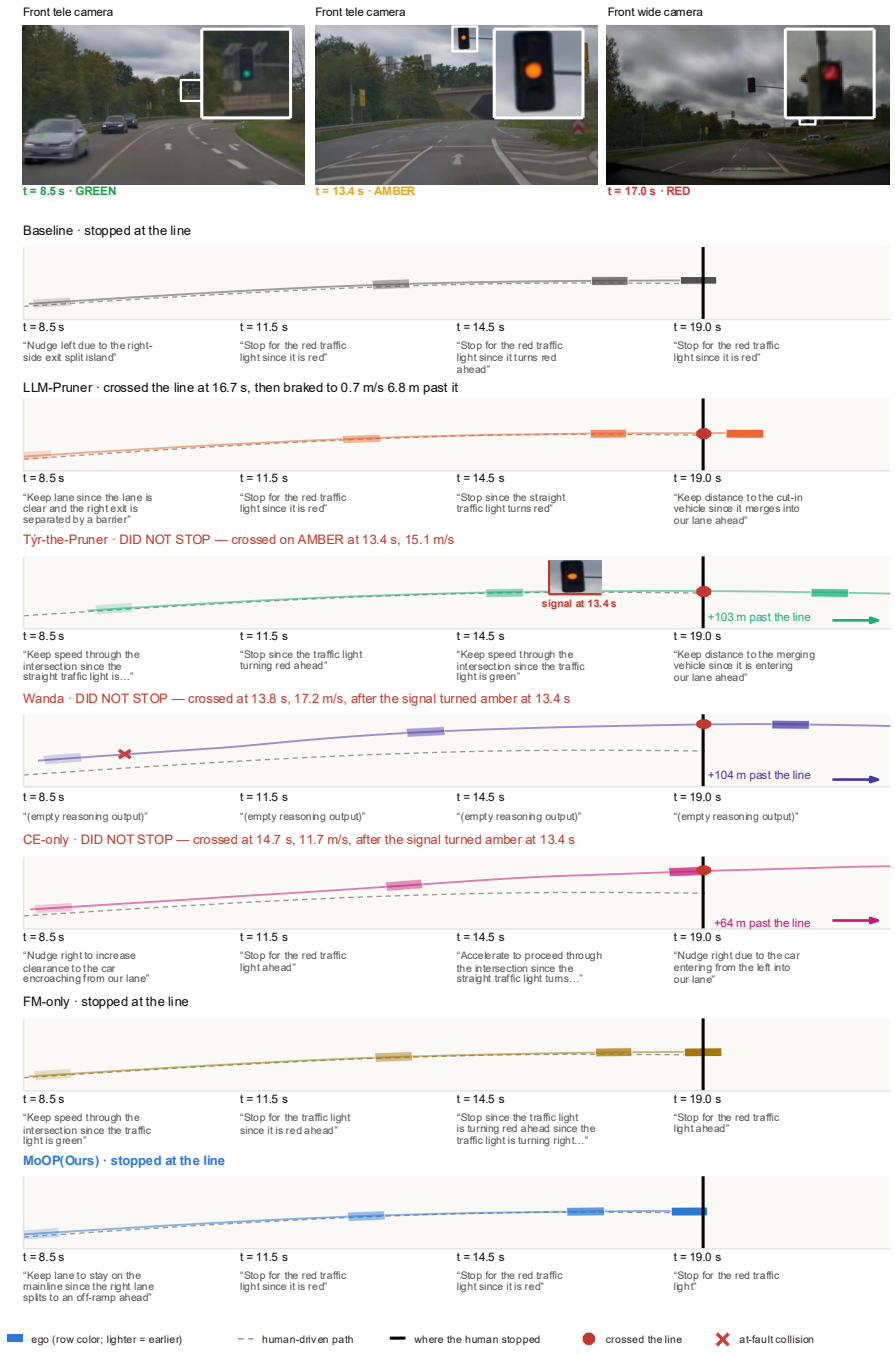


Figure 14: **Closed loop: signal changing green→amber→red.** Top: three camera frames at green, amber, and red, each with a magnified inset of the signal. Below: one strip per method along the approach, with the point where the human driver stopped (black line), the point where the method crossed that line (red dot), and the method's reasoning at four times. The Baseline, FM-only, and MoOP stop at the line. LLM-Pruner crosses at 16.7 s and brakes to 0.7 m/s, 6.8 m past the line. Týr-the-Pruner crosses on amber at 13.4 s at 15.1 m/s, and Wanda and CE-only cross after the signal turned amber, at 17.2 and 11.7 m/s. None of these three stops, and they end 64–104 m past the line. Wanda also has an at-fault collision at 9.0 s, before reaching the intersection.



Figure 15: **Closed loop: bus stopped in the right part of the ego lane.** Týr-the-Pruner, Wanda, CE-only, and FM-only have at-fault collisions. The Baseline keeps its distance behind the bus, and MoOP reasons that the bus blocks the right side of the lane and nudges left. Neither collides.

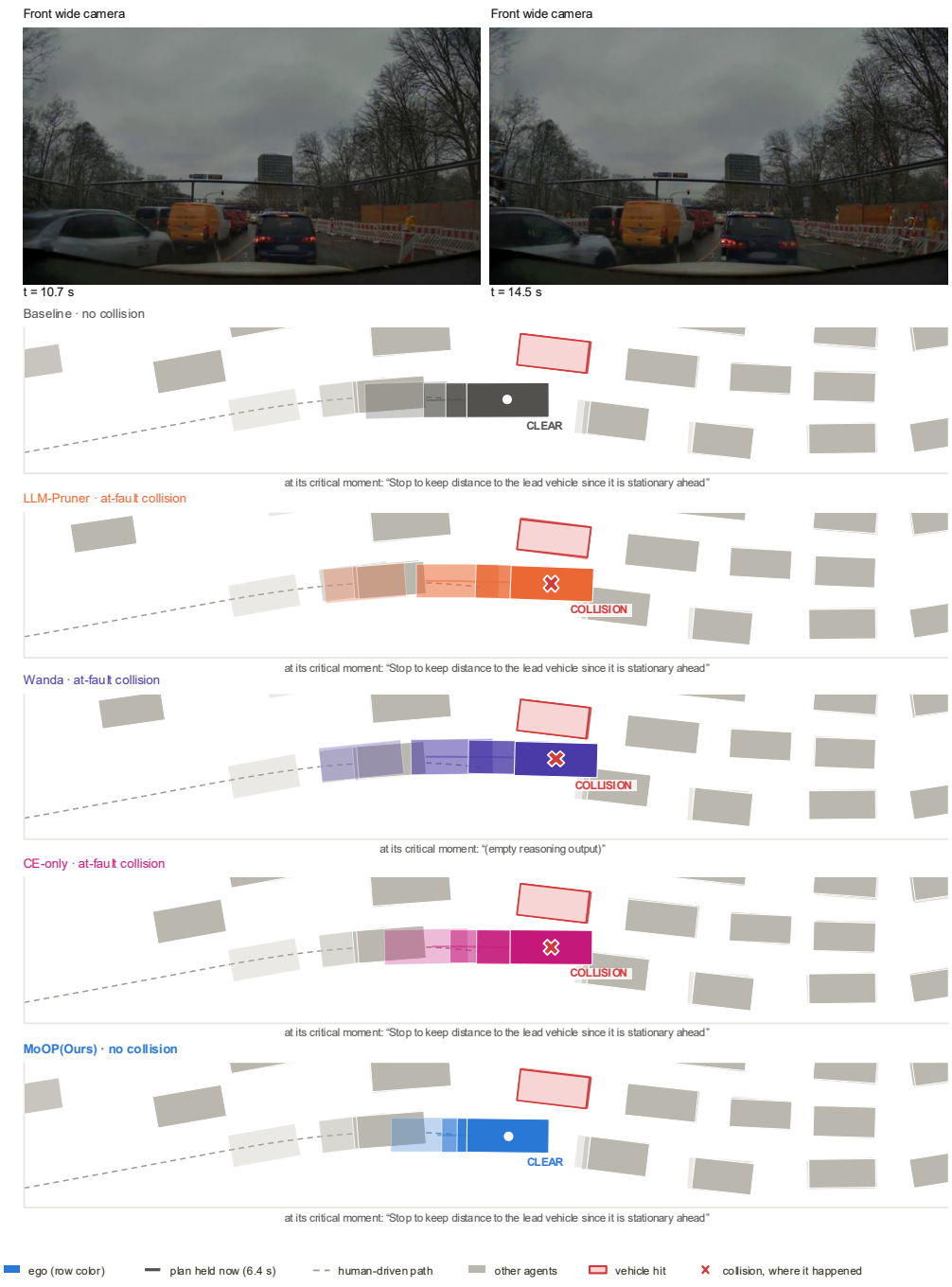


Figure 16: **Closed loop: stopped queue ahead.** LLM-Pruner, Wanda, and CE-only have at-fault collisions. LLM-Pruner and CE-only state the same intent as the Baseline and MoOP, to stop behind the stationary lead vehicle. The Baseline and MoOP stop without contact.



Figure 17: **Closed loop: slow lead vehicle.** Wanda and CE-only have at-fault collisions although all four methods state the same intent, to keep their distance to the slowing lead vehicle. The Baseline and MoOP have no collision.



Figure 18: **Closed loop: vehicle merging from the left under low sun.** Wanda and CE-only have at-fault collisions. CE-only proposes to nudge left toward a traffic island. The Baseline and MoOP both reason about the car merging from the left, nudge right, and have no collision.



Figure 19: **Closed loop: slow queue ahead.** Wanda and CE-only have at-fault collisions. CE-only states the same intent as the Baseline and MoOP, to keep its distance to the slowing lead vehicle. The Baseline and MoOP keep their distance and have no collision.

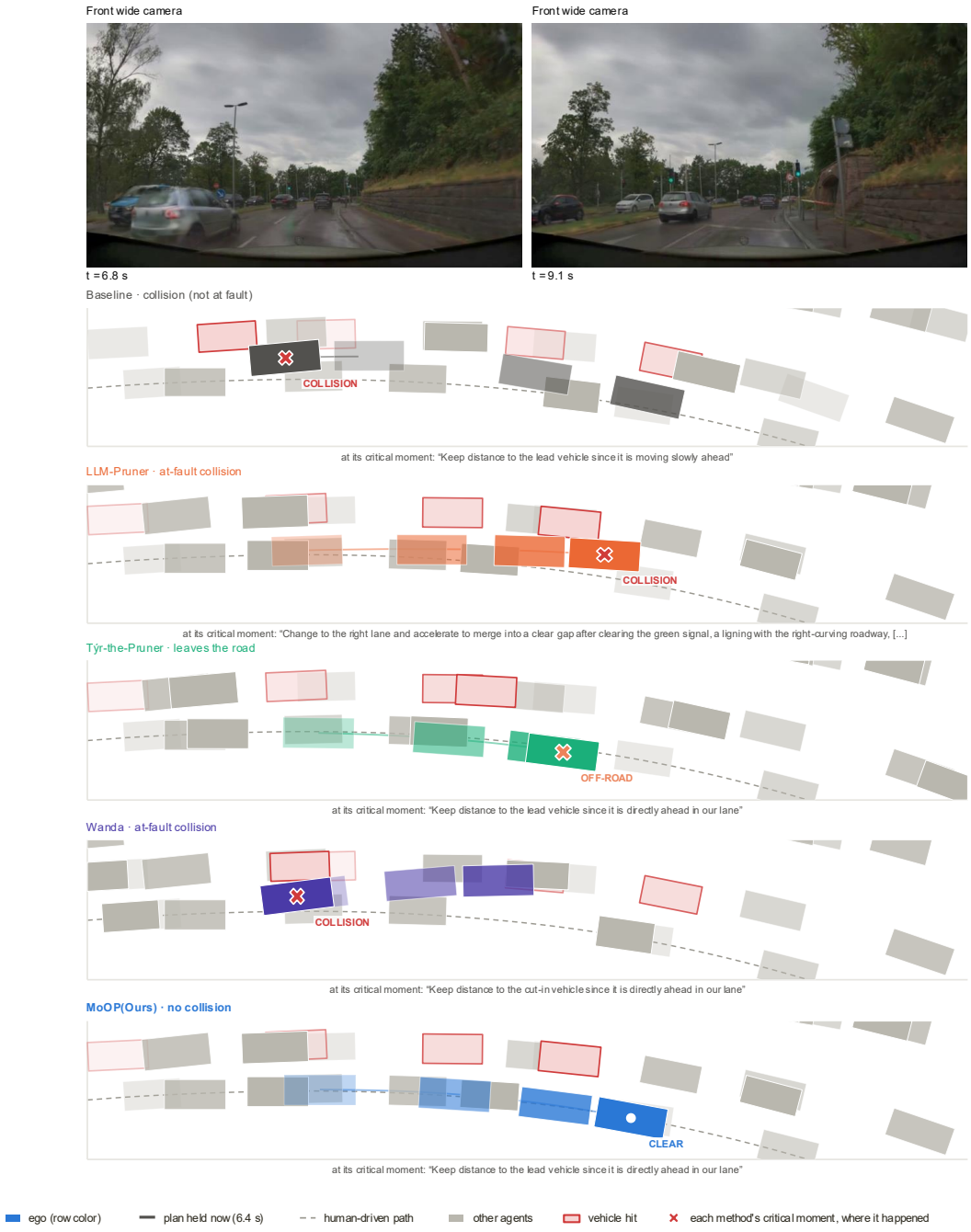


Figure 20: **Closed loop: dense traffic on a wet road.** The Baseline is hit (not at fault), LLM-Pruner and Wanda have at-fault collisions, and Týr-the-Pruner leaves the road while stating the same intent as MoOP. MoOP has no collision.

B.4 Multi-Answer Cases in Closed-Loop Evaluation

As the examples in §B.3 show, the closed-loop score reflects driving quality in most scenes. The exception we observed is the multi-answer case: a driving situation that admits more than one acceptable behavior, of which the protocol rewards only one. At an intersection, for example, both turning and going straight can be valid. Each closed-loop scene, however, has a single reference: the path the human driver took when the scene was recorded. The model is given no navigation command, so nothing tells it which of the valid options that path follows. AlpaSim (NVIDIA et al., 2025b) measures progress along the recorded path and stops scoring once the ego is more than $4\,\mathrm{m}$ from it. A model that chooses a valid option other than the recorded one therefore receives a lower score than its driving warrants.

Figs. 21–24 show scenes where the unpruned Baseline scores below MoOP for this reason. In three of them, the Baseline states a route choice that the scene supports (a clear side street, a right-turn signal it reads as green, cross-traffic that has cleared) and takes it, while the recorded driver goes the other way. In the fourth, the recorded driver bears right at a fork and the Baseline keeps its lane. Scoring stops where the two paths separate, so the Baseline is credited with nothing it drives afterwards, whereas MoOP follows the recorded route and scores 1.000 in all four. In these scenes, the lower score of the Baseline reflects which route it chose rather than how well it drove.

The ambiguity could be removed by conditioning the policy on a navigation command or by scoring against all drivable routes consistent with the scene. We leave this to future work.

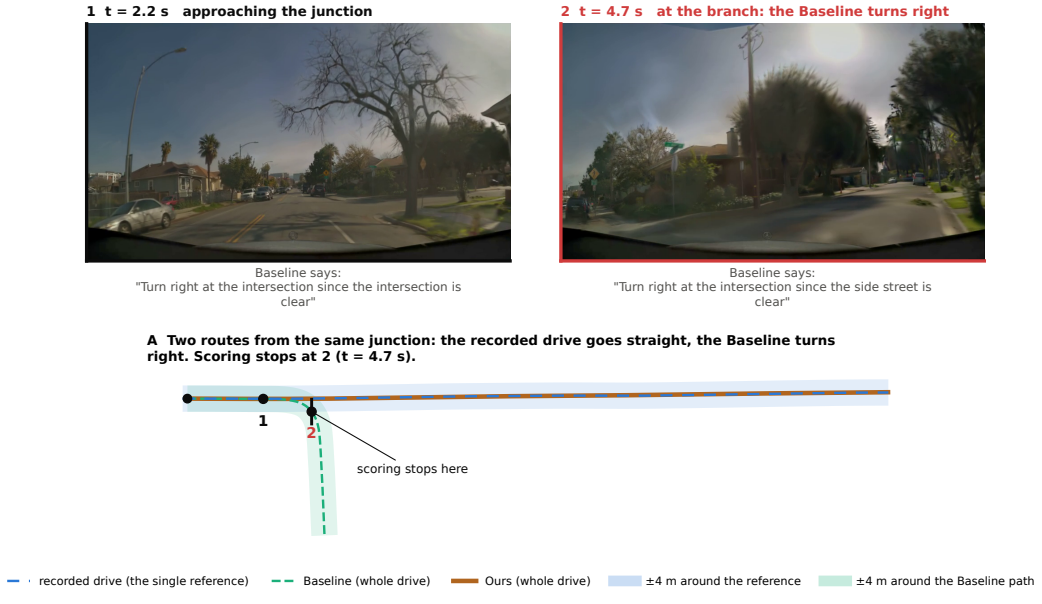


Figure 21: **Multi-answer: turning right at a residential crossroads.** The recorded driver goes straight, and the Baseline turns right after reasoning that the side street is clear. Scoring stops at the branch (2), and the rollout scores 0.221 (0.223 over the scene) although it continues to drive; MoOP goes straight and scores 1.000.

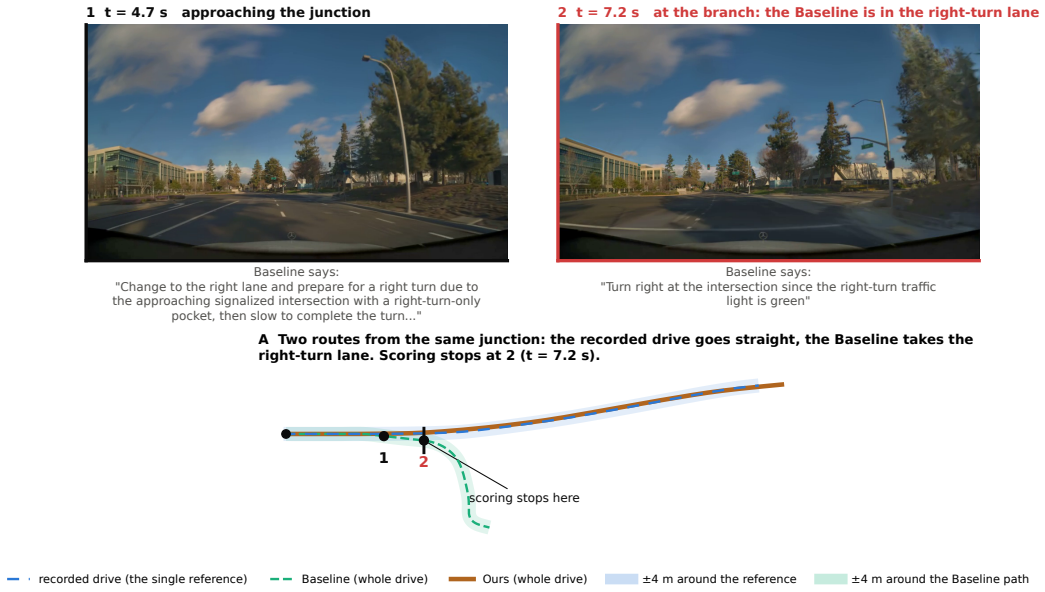


Figure 22: **Multi-answer: a signalled right turn.** The recorded driver goes straight, and the Baseline moves into the right-turn pocket and turns after reasoning that the right-turn light is green. Scoring stops at the branch (2), and the rollout scores 0.360 (0.680 over the scene); MoOP goes straight and scores 1.000.

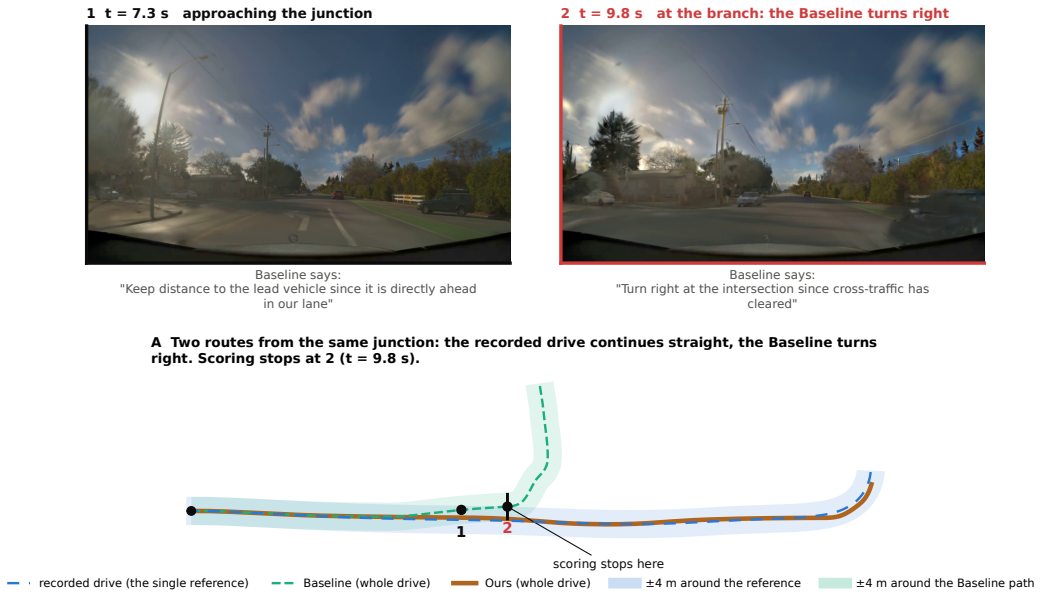


Figure 23: **Multi-answer: turning right once cross-traffic clears.** The recorded driver continues straight, and the Baseline turns right after reasoning that cross-traffic has cleared. Scoring stops at the branch (2), and the rollout scores 0.560 (0.780 over the scene); MoOP continues straight and scores 1.000.

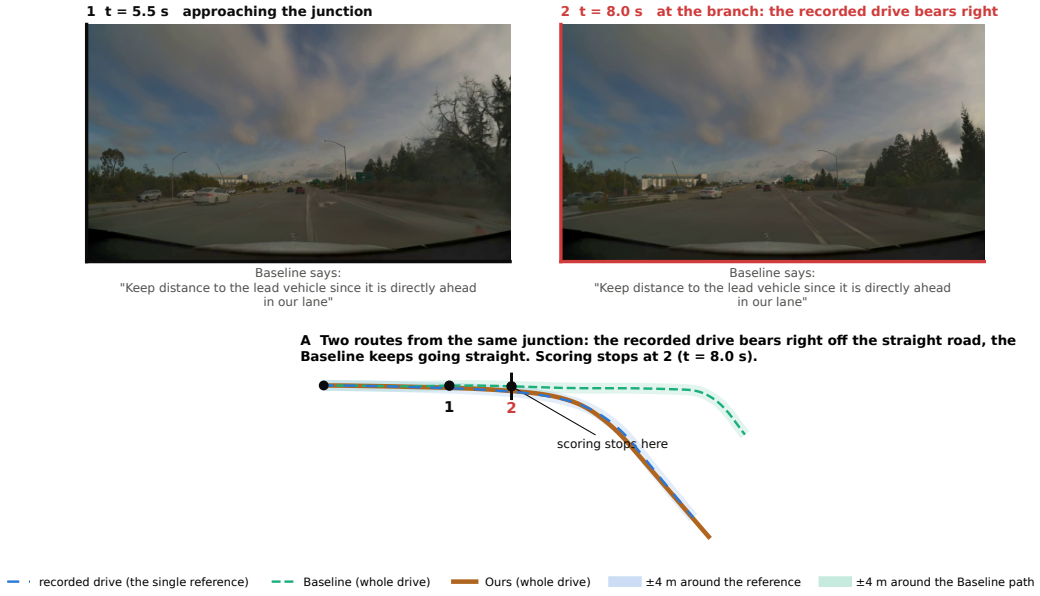


Figure 24: **Multi-answer: a fork on a multi-lane road.** Here the recorded driver leaves the straight road and bears right, while the Baseline keeps its lane behind the lead vehicle. Scoring stops at the branch (2), and the rollout scores 0.563 (0.782 over the scene); MoOP follows the recorded fork and scores 1.000.