Coarse-to-Refine: Trajectory Self-Refinement in Single Autoregressive Pass for Driving VLA

Canyu Chen[1,3*] **Yuguang Yang**[2,3*] **Jianing Pang**[7] **Zhewen Tan**[7]
Cheng Chi[4] Chunyang Liu[5] Kehua Sheng[5] Linlin Yang[6]
Xiaoyan Luo[8] Bo Zhang[5†] Yan Wang[3†] Baochang Zhang[9†]

[1]National College for Excellent Engineers, Beihang University
[2]School of Electronic and Information Engineering, Beihang University
[3]Institute for AI Industry Research, Tsinghua University
[4]School of Information, Renmin University of China [5]DiDi
[6]State Key Laboratory of Media Convergence and Communication,
Communication University of China
{[7]School of Computer Science and Engineering, [8]School of Astronautics,
[9] Beihang University Hangzhou Innovation Research Institute }, Beihang University

Abstract

Vision-language-action (VLA) models have emerged as a promising paradigm for autonomous driving trajectory planning. While explicit test-time thinking has driven mainstream progress in Large Language Models (LLMs), how to effectively utilize reasoning in driving VLA remains unsettled. We revisit this gap through Coarse-to-Refine (C2R), a trajectory self-refinement framework for driving VLA that completes refinement in a single autoregressive pass. C2R first predicts a coarse trajectory, performs explicit reasoning over it, and then generates a refined trajectory autoregressively. We show that the refinement formulation itself drives Imitation Learning improvement while enabling Reinforcement Learning to utilize reasoning content for further optimization. C2R establishes VLA state-of-the-art performance across standard and challenging benchmarks, achieving 92.1 PDMS on NAVSIM v1, 90.4 EPDMS[†3] on NAVSIM v2, and 49.6 EPDMS on NavHard. Code is available at `https://mashiroln.github.io/coarse2refine/`.

1 Introduction

Vision-Language-Action (VLA) models formulate autonomous driving as a unified multimodal control problem that integrates perception, language understanding, and trajectory planning. They have shown strong potential for end-to-end driving by exploiting pretrained knowledge and language reasoning. Recent methods including Senna [22], DriveVLM [50], AutoVLA [65], and AdaThinkDrive [36] aim to harness the reasoning abilities of Large Language Models (LLMs) by inserting scene descriptions, risk assessments, or meta-action predictions ahead of the planned trajectory, thereby enhancing the quality of planning.

However, how reasoning should enter trajectory planning in driving VLA remains unsettled. Existing methods largely split into explicit and implicit designs. Explicit-reasoning methods [29, 34, 65] enrich reasoning content with auxiliary tasks and supervision signals, whereas implicit-reasoning methods such as ColaVLA [41] and NoRD [42] compress reasoning into latent representations or

*Equal contribution: `{chencanyu,guangbuaa}@buaa.edu.cn`
†Corresponding author: `zhangbo@didiglobal.com`, `wangyan@air.tsinghua.edu.cn`, `bczhang@buaa.edu.cn`
[3]EPDMS[†] denotes the bug-fixed EPDMS metric released by the official repository of NAVSIM v2.

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

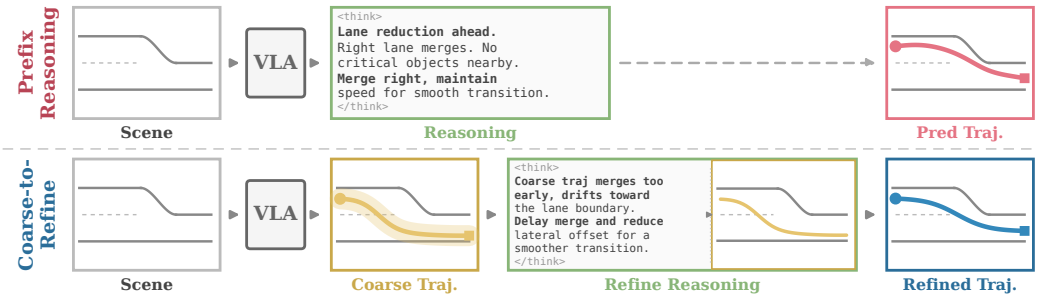


Figure 1: Trajectory planning formulations in autoregressive VLAs. In **Prefix Reasoning** (top), reasoning precedes the trajectory and describes the driving scene. In **Coarse-to-Refine** (bottom), a coarse trajectory is generated first, and reasoning guides its refinement into the final trajectory.

remove language reasoning altogether. Inspired by the pivotal role of explicit test-time thinking in advancing LLMs, our work re-examines explicit reasoning in driving VLA and seeks to unlock its potential for trajectory planning.

Most explicit-reasoning driving VLAs adopt **Prefix Reasoning**: the model generates reasoning before predicting the trajectory and uses it as prefix context. Prior work mainly enriches what the model reasons about, without changing how reasoning enters trajectory prediction. As a result, reasoning remains only loosely coupled with trajectory generation. If a trajectory draft is generated first, reasoning shifts to assessing a concrete plan, and each part of the response takes on a simpler, more direct role. To better utilize explicit reasoning in trajectory planning, we propose **Coarse-to-Refine (C2R)**, an autoregressive trajectory self-refinement framework for driving VLA. Specifically, C2R first drafts a coarse trajectory, uses explicit reasoning to assess and revise the draft, and then outputs the refined trajectory. Reasoning thus shifts from prefix context to an intermediate refinement step. Unlike ELF-VLA [37] and DriveFine [10], which rely on post-hoc feedback loops or additional planners for refinement, C2R completes refinement in a single autoregressive pass.

Training this formulation requires a dataset of samples that pair a coarse trajectory with reasoning and a revised trajectory, which serves as the *refined trajectory* in the C2R response. Standard trajectory annotations do not provide such samples. We therefore train C2R with an IL-to-RL (Imitation Learning to Reinforcement Learning) pipeline. During Imitation Learning, Guided Self-Refinement Synthesis (GSRS) and supervised fine-tuning are interleaved: we first train a Prefix Reasoning policy as warmup; GSRS then uses this policy on the training set to synthesize the refined trajectories through sampling, scoring, and selecting; a strong VLM annotates how each ground-truth (GT) trajectory should be revised toward the selected trajectory; and supervised fine-tuning trains C2R on the constructed dataset. Reinforcement Learning with Verified Reward (RLVR) then optimizes the final trajectory with segment-weighted SAPO.

Experimental results validate the proposed method: (i) it achieves state-of-the-art performance across standard and challenging benchmarks, attaining 92.1 PDMS on NAVSIM v1, 90.4 EPDMS† on NAVSIM v2, and 49.6 EPDMS on NavHard; (ii) controlled variants in Section 5.1 demonstrate that the C2R formulation drives IL improvement through refinement, independently of reasoning content; mechanistic analyses in Section 5.2 verify that reasoning is functionally utilized in trajectory prediction, providing the basis for RLVR to further optimize trajectory quality. Our contributions are summarized as follows:

- We propose Coarse-to-Refine (C2R), an autoregressive refinement formulation for driving VLA that moves explicit reasoning from prefix context to a trajectory-revision step.
- We develop a practical training pipeline that constructs the required training data, trains C2R with IL, and optimizes the final trajectory with RLVR.
- Through ablations and mechanistic analyses, we show that the C2R formulation drives IL improvement through refinement, while enabling RLVR to utilize reasoning for trajectory optimization. The final model achieves VLA state-of-the-art results on NAVSIM v1/v2 and NavHard.

2 Related Work

End-to-End Autonomous Driving. End-to-end autonomous driving unifies perception, prediction, and planning into a single learnable framework [9, 16, 17, 20, 21, 30, 32, 54]. Recent work improves trajectory modeling and optimization through residual learning [62], human-feedback RL [25], generalized scoring [1, 24, 31], diffusion-based refinement [10], and world model planning [28, 55, 61]. Recent driving VLA methods further introduce Vision-Language Models (VLMs) into end-to-end driving [2, 56]. Some methods condition a separate planning module on VLM representations or generated tokens [13, 26, 27, 29, 43]. Others directly decode trajectories within the VLM sequence, either as text waypoints [6, 8, 19, 44, 53, 64] or as discrete action tokens [51, 65]. C2R follows the *text-waypoint formulation* rather than introducing an extra planner.

Reasoning in Driving VLAs. A reasoning process in driving VLA denotes textual or latent intermediate steps that organize scene understanding and driving decisions before action generation. Existing work largely adopts Prefix Reasoning by default: reasoning precedes trajectory tokens and acts as context for action or waypoint generation. Explicit-Reasoning methods instantiate this form with scene descriptions, driving explanations [6, 22, 36, 47, 50, 58, 60, 65]. Implicit-Reasoning methods reduce or remove visible reasoning text: ColaVLA [41] compresses language reasoning into cognitive latent representations, NoRD [42] removes reasoning tokens, and related latent-reasoning variants are also considered [38, 48, 52]. Together, these methods mainly vary the content or visibility of the reasoning prefix. C2R instead defines a different causal role for reasoning: it refines a self-generated coarse trajectory, rather than serving as prefix content before trajectory prediction.

Trajectory-Centered Refinement for Planning. Trajectory-centered refinement treats the trajectory not only as the final output, but also as a representation to be modeled, scored or optimized [10, 24, 31, 37, 55, 61, 62]. LLMs have explored self-refinement through draft-feedback-correction loops [23, 39, 46], while embodied VLA has explored action-level refinement [4, 18, 63]. In driving, a trajectory is a natural refinement target: it already specifies the planned motion and can be directly assessed by planning metrics. C2R further internalizes refinement into the policy sequence itself, completing coarse-to-refined trajectory prediction in a single autoregressive pass.

3 Method

We present C2R in three parts: Section 3.1 formalizes the response formulation for text-waypoint generation (Figure 2(a)); Section 3.2 describes data construction and IL training; and Section 3.3 presents RLVR optimization and inference.

3.1 Coarse-to-Refine Formulation

Trajectory Planning Setup. For each driving scene, the policy observes a scene input $\mathcal{X}$, including camera/sensor observations, a navigation command, and current ego state, and predicts a future ego trajectory $\tau = \{w_k\}_{k=1}^{K}$. Each waypoint $w_k = (x_k, y_k, \psi_k)$ specifies an ego-frame pose, where x and y denote forward and left displacement, and ψ denotes heading. In our experiments, the trajectory spans four seconds at 2 Hz ($K = 8$). We serialize the waypoints into ordinary text tokens and train a VLA policy π_θ to generate the response autoregressively.

From Prefix Reasoning to C2R. Existing explicit-reasoning VLA systems [34, 65] commonly follow the Prefix Reasoning formulation: reasoning R is generated before the trajectory τ:

$$p_\theta(R, \tau \mid \mathcal{X}) = p_\theta(R \mid \mathcal{X})\, p_\theta(\tau \mid \mathcal{X}, R). \tag{1}$$

Under SFT teacher forcing, each token y_t is predicted with the ground-truth prefix $y_{<t}$ visible, so the response formulation determines which labeled fields each token can access during training. In Prefix Reasoning, trajectory tokens are trained with ground-truth reasoning visible, allowing shortcuts through this prefix. C2R follows `coarse trajectory` $\rightarrow$ `reasoning` $\rightarrow$ `refined trajectory`: the coarse trajectory τ^c is a scene-conditioned draft learned without reasoning dependency, and the refined trajectory τ^f is generated after both the coarse draft and reasoning, leveraging the maximum preceding context in a single autoregressive pass:

$$p_\theta(\tau^c, R^{\mathrm{ref}}, \tau^f \mid \mathcal{X}) = p_\theta(\tau^c \mid \mathcal{X})\, p_\theta(R^{\mathrm{ref}} \mid \mathcal{X}, \tau^c)\, p_\theta(\tau^f \mid \mathcal{X}, \tau^c, R^{\mathrm{ref}}). \tag{2}$$

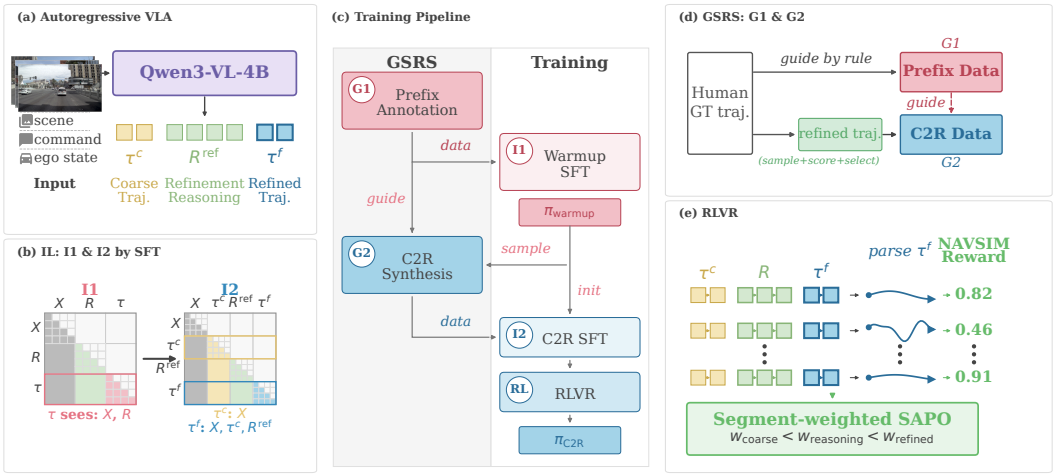


Figure 2: Overview of C2R. (a) The autoregressive VLA generates coarse trajectory τ^c, refinement reasoning R^{ref}, and refined trajectory τ^f in a single pass. The training pipeline (c) interleaves GSRS data construction (G1, G2) with IL training (I1, I2). G1 annotates Prefix reasoning for I1, which produces the warmup policy π_{warmup}; π_{warmup} then both supplies trajectory candidates for G2 to synthesize C2R data and initializes I2 for C2R training. (d) shows how G1 and G2 construct the training data, and the causal masks in (b) illustrate the context visible to each field under I1 and I2. After IL, RLVR (e) optimizes τ^f with Segment-Weighted SAPO.

This formulation gives each field a simpler role. The coarse trajectory τ^c captures scene input directly without relying on reasoning; reasoning R^{ref} assesses a concrete draft rather than planning before any trajectory exists; and the refined trajectory τ^f applies this assessment as a correction to the draft. In C2R, τ^c appears first in the response, predicted from $\mathcal{X}$ alone with no ground-truth labels visible in its context, unlike Prefix Reasoning where trajectory tokens see labeled reasoning (Figure 2(b)).

3.2 Data Construction and Imitation Learning

Equation 2 formulates C2R as autoregressive generation of $(\tau^c, R^{\text{ref}}, \tau^f)$, while standard datasets provide only τ. Thus, training requires the missing refined trajectory τ^f and refinement reasoning R^{ref}. We obtain these components through GSRS and learn C2R through IL implemented by SFT. The pipeline interleaves GSRS data construction and IL training (Figure 2(c)): **G1: Prefix Annotation** precedes **I1: Warmup SFT**, which enables **G2: C2R Synthesis** before **I2: C2R SFT**.

Chain-of-Thought Schema. Following prior driving chain-of-thought designs such as Poutine [44], reasoning contains three fields. In Prefix Reasoning, reasoning follows `critical objects` → `meta behavior` → `action explanation`; in C2R, it follows `critical objects` → `meta behavior` → `refinement explanation`, where the explanation describes how to adjust the coarse trajectory rather than the planned action.

Guided Self-Refinement Synthesis (GSRS). GSRS constructs the C2R supervision on the training split through G1 and G2 (Figure 2(d)). **G1: Prefix Annotation.** Rule proposals for `critical objects` and `meta behavior` guide a strong VLM (Qwen3-VL-235B) to verify these proposals and produce Prefix reasoning annotations alongside the human GT trajectory τ. **G2: C2R Synthesis.** After Warmup SFT trains π_{warmup}, GSRS samples candidate trajectories from it for each training scene, scores these trajectories with the standard NAVSIM scorer, and selects the best candidate as the refined trajectory τ^f. The human GT trajectory is used as the coarse trajectory τ^c, where "coarse" denotes its position as the first trajectory in the C2R response, and the VLM labels refinement reasoning R^{ref} from τ^c to τ^f. G1 Prefix annotations guide `critical objects` and `meta behavior`, while the VLM may revise them when τ^f implies a different refinement behavior. More details are in Appendix B.

Imitation Learning via Supervised Fine-Tuning. In autoregressive VLA, IL is implemented as SFT over the response format used in each training pass. We train C2R with two SFT passes

(Figure 2(b)) with the standard assistant-only next-token objective:

$$\mathcal{L}_{\mathrm{SFT}} = -\sum_t m_t \log p_\theta(y_t^* \mid \mathcal{X}, y_{<t}^*), \tag{3}$$

where m_t masks non-assistant tokens. **I1: Warmup SFT** (Figure 2(b)) trains a Prefix Reasoning policy on the Prefix annotations from G1, teaching the base VLA to generate reasoning and text-waypoint trajectories; this pass produces π_{warmup}, which is used both for candidate sampling above and as the initialization for I2. **I2: C2R SFT** (Figure 2(b)) trains on the C2R response format $y_{\mathrm{C2R}} = (\tau^c, R^{\mathrm{ref}}, \tau^f)$, so the policy learns the generation process defined in Eq. 2.

3.3 RLVR and Inference

After IL, RLVR further optimizes the final trajectory τ^f using the NAVSIM score as a verified reward (Figure 2(e)). Because the response contains multiple segments, we weight token updates by segment while assigning reward only from τ^f.

RLVR Reward. Following GRPO [45], the policy samples a group of G responses per input $\mathcal{X}$, and the group-relative advantage A_i is computed by normalizing rewards within each group. In each response, the policy generates its own coarse trajectory τ^c, followed by refinement reasoning R^{ref} and the refined trajectory τ^f. A reward parser extracts τ^f from each response and assigns $\mathcal{R}(\mathcal{X}, y) = \mathcal{S}_{\mathrm{nav}}(\tau^f; \mathcal{X})$ if the response is parseable, and 0 otherwise. Rewarding only τ^f avoids reward-hacking, while earlier segments remain autoregressive context for refinement.

Segment-Weighted SAPO (SWSAPO). We adopt SAPO [14] which uses a soft sigmoid gate $g_\alpha(r_t) = \frac{4}{\alpha}\sigma(\alpha(r_t - 1))$ to replace GRPO's hard importance-ratio clipping, continuously attenuating off-policy token gradients, where t indexes tokens in the response, r_t is the token-level likelihood ratio, and α is a sharpness hyperparameter. In C2R, each response contains three segments (τ^c, R^{ref}, and τ^f) with different roles relative to the reward. We assign per-segment weights to the SAPO loss:

$$\mathcal{L}_{\mathrm{SWSAPO}} = -\frac{1}{\sum_t m_t}\sum_t m_t\, A\, g_\alpha(r_t)\, w_{s(t)}, \quad w_{\mathrm{coarse}} < w_{\mathrm{reasoning}} < w_{\mathrm{refined}}, \tag{4}$$

aligning token-level gradient magnitude with proximity to the rewarded trajectory. This improves RLVR training stability while keeping the reward computed only from τ^f.

Inference. At inference time, C2R uses the same formulation: given only $\mathcal{X}$, the policy autoregressively generates τ^c, R^{ref}, and τ^f in a single pass. The coarse trajectory is generated by the policy itself, not prefilled by an external planner, and no post-process refinement loop is used. Only τ^f is used as the final planning output, realizing trajectory self-refinement in a single autoregressive pass.

4 Experiments

4.1 Experimental Setup

NAVSIM. We evaluate on NAVSIM [11], a non-reactive closed-loop planning benchmark on nuPlan [3] that scores planned trajectories. NAVSIM v1 uses PDM Score (PDMS); v2 [5] extends it to EPDMS. Sub-metric definitions are in Appendix C. We report both the original EPDMS and the bug-fixed EPDMS†, and use EPDMS† in all ablations (Section 5).

NavHard. NavHard [5] is a two-stage pseudo-closed-loop benchmark for safety-critical scenarios. Stage 1 evaluates planning on real observations while background vehicles react via the Intelligent Driver Model (IDM); Stage 2 perturbs the ego pose and renders synthetic observations, testing robustness under distribution shift. The final score is the product of per-stage EPDMS.

Implementation. C2R is built on Qwen3-VL-4B with a single front-view camera. We follow the IL-to-RL training pipeline described in Sections 3.2–3.3, and adopt Adaptive Diversity-Aware Sampling (ADAS) [6] to stabilize RL exploration. Models are trained on $\sim$103K `navtrain` scenes by default; hyperparameters and training schedules are provided in Appendix D.

Table 1: Comparison on NAVSIM v1 NavTest. C and L denote camera views and LiDAR, respectively. ⋆ denotes concurrent work. The best and second-best results are **bolded** and underlined, respectively.

	Base Model	Ref	Sensors	NC↑	DAC↑	EP↑	TTC↑	C↑	PDMS↑
Classical E2E									
UniAD [17]	–	CVPR'23	6×C	97.7	92.8	79.2	92.8	100.0	84.0
DiffusionDrive [32]	–	CVPR'25	3×C+L	98.2	96.2	82.2	94.7	100.0	88.1
WoTE [28]	–	ICCV'25	3×C+L	98.5	96.8	81.9	94.4	99.9	88.3
DiffusionDriveV2 [66]	–	arXiv'25	3×C+L	98.3	97.9	87.5	94.8	99.9	91.2
Planner-Based VLA									
ImagiDrive-S [26]	InternVL2.5-4B	ICRA'26	1×C	98.6	96.2	80.5	94.5	100.0	87.4
DriveVLA-W0 [27]	Emu3-8B	ICLR'26	1×C	98.7	**99.1**	83.3	95.3	99.3	90.2
ReCogDrive [29]	InternVL3-8B	ICLR'26	3×C	97.9	97.3	87.3	94.9	100.0	90.8
DriveFine⋆ [10]	LLaDA-8B	arXiv'26	1×C	98.8	98.6	86.9	96.2	100.0	91.8
Explicit Reasoning VLA									
AutoVLA [65]	Qwen2.5-VL-3B	NeurIPS'25	3×C	98.4	95.6	81.9	**98.0**	99.9	89.1
AutoDrive-R^2 [58]	Qwen2.5-VL-7B	ICLR'26	1×C	98.5	95.9	82.7	95.4	100.0	89.1
AdaThinkDrive [36]	InternVL3-8B	ICRA'26	1×C	98.4	97.8	84.4	95.2	100.0	90.3
Curious-VLA⋆ [6]	Qwen2.5-VL-3B	CVPR'26	1×C	98.4	96.9	88.5	97.9	98.1	90.3
ELF-VLA⋆ [37]	InternVL3-8B	CVPR'26	1×C	**98.9**	98.1	85.3	96.0	100.0	91.0
Implicit Reasoning VLA									
NoRD [42]	Qwen2.5-VL-3B	CVPR'26	1×C	97.6	94.9	79.3	93.5	100.0	85.6
OneDrive⋆ [59]	InternVL3-2B	arXiv'26	6×C	98.4	95.2	81.1	94.9	100.0	86.8
LaST-VLA⋆ [38]	InternVL3-8B	arXiv'26	1×C	98.7	97.9	86.8	95.6	100.0	91.3
FLARE [52]	Qwen3-VL-4B	arXiv'26	1×C	98.5	98.4	86.0	96.0	100.0	91.4
Ours									
Prefix Reasoning(baseline)	Qwen3-VL-4B	–	1×C	97.8	95.1	80.4	94.5	100.0	86.5
Coarse-to-Refine (C2R)	Qwen3-VL-4B	–	1×C	98.8	97.9	**89.0**	95.4	100.0	**92.1**

Table 2: Comparison on NAVSIM v2 NavTest. † denotes scores evaluated with the bug-fixed EPDMS metric released by the NAVSIM v2 authors. ⋆ denotes concurrent work. The best and second-best results are **bolded** and underlined, respectively.

	NC↑	DAC↑	DDC↑	TLC↑	EP↑	TTC↑	LK↑	C↑	EC↑	EPDMS↑
Human GT [11]	100.0	100.0	99.8	100.0	87.4	100.0	100.0	98.1	100.0	90.3 (†95.5)
End-to-End										
TransFuser [9]	96.9	89.9	97.8	99.7	87.1	95.4	92.7	98.3	87.2	76.7
DriveSuprim [57]	97.5	96.5	99.4	99.6	88.4	96.6	95.5	98.3	77.0	83.1
DiffusionDrive [32]	98.2	95.9	99.4	99.8	87.5	97.3	96.8	98.3	87.7	84.5
DiffusionDriveV2 [66]	97.7	96.6	99.2	99.8	88.9	97.2	96.0	97.8	**91.0**	85.5
VLA										
ReCogDrive [29]	98.3	95.2	**99.5**	99.8	87.1	97.5	96.6	98.3	86.5	83.6
Curious-VLA⋆ [6]	98.4	96.9	99.2	99.8	88.5	97.9	96.9	98.1	81.5	85.3
DriveVLA-W0 [27]	**99.0**	**98.4**	99.3	**99.9**	87.0	97.8	93.2	97.9	58.9	86.5
FLARE [52]	98.8	96.6	**99.5**	**99.9**	87.9	98.2	95.8	**98.4**	87.5	86.3
DriveFine⋆ [10]	98.7	97.3	98.8	99.8	88.2	97.8	**97.7**	**98.4**	84.7	87.1 (†89.7)
LaST-VLA⋆ [38]	98.7	97.9	99.2	99.7	90.3	98.2	96.6	98.3	86.3	87.1
Ours										
Coarse-to-Refine (C2R)	98.9	97.9	99.0	**99.9**	**91.1**	**98.3**	96.4	98.3	83.2	**87.2** (†**90.4**)

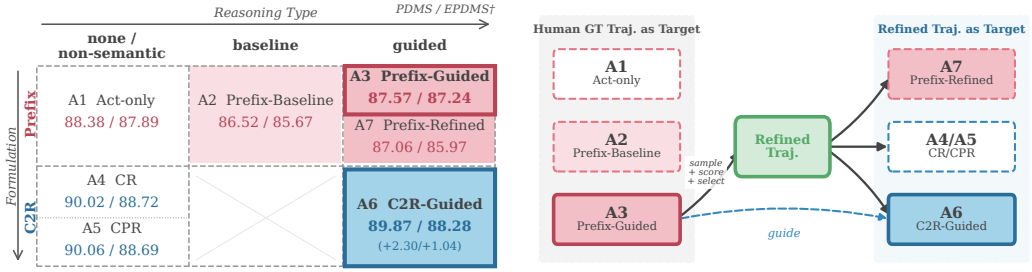
4.2 Main Results

NAVSIM. Tables 1 and 2 compare C2R against recent end-to-end planners and driving VLAs. C2R achieves the best results on both protocols: **92.1 PDMS** on v1 and **90.4 EPDMS**† (87.2 EPDMS) on v2. The PDMS-trained policy also leads under EPDMS and EPDMS† (Appendix C). Notably, C2R reaches these scores with a 4B backbone and a single camera, while all other top-performing methods [10, 27, 37, 38] rely on 8B-class models and/or multi-camera inputs. On the same Qwen3-VL backbone, Prefix Reasoning with direct LLM annotations reaches only 86.5 PDMS (Table 1). The gain is consistent across sub-metrics, with the main improvement on EP.

NavHard. In Table 3, scorer-based methods are reported separately for fair comparison. C2R sets a new state of the art among non-scorer methods: *(i)* **Among VLAs.** Without extra data, C2R reaches **34.0 EPDMS** on `navtrain` alone, outperforming all other VLAs including Curious-VLA and DriveFine. *(ii)* **Among all non-scorer methods.** With SimScale [49] augmentation, C2R reaches

Table 3: Results on NavHard [5]. Gray rows denote scorer-based methods. ⋆ denotes concurrent work. ‡ denotes VLA-based methods. Full sub-metric breakdown is in Appendix.

		Stage 1					Stage 2					EPDMS↑		
	Extra Data	NC	DAC	EP	TTC	HC	NC	DAC	EP	TTC	HC	Stg.1	Stg.2	Final
PDM-Closed	–	94.4	98.8	100.0	93.5	97.7	88.1	90.6	100.0	83.1	91.5	–	–	51.3
Scorer-Based														
GTRS-D [31]	–	98.9	94.9	76.1	98.4	94.9	89.9	90.5	77.6	88.5	92.0	–	–	41.9
GuideFlow-Scorer [33]	–	98.8	95.5	76.0	99.1	97.5	88.8	89.4	80.3	86.7	96.9	–	–	43.0
DriveSuprim [57]	–	98.7	98.0	75.9	98.7	97.6	89.5	89.6	78.9	86.4	96.5	–	–	44.7
GTRS-D [31]	+SimScale	99.1	98.2	71.9	99.3	93.8	92.3	93.8	75.4	90.7	95.9	–	–	47.2
DrivoR [24]	–	98.8	95.1	72.6	98.7	97.6	90.2	88.4	70.0	88.0	98.5	–	–	48.3
DrivoR [24]	+SimScale	99.1	98.2	75.4	98.7	97.6	92.3	91.6	75.7	90.6	98.4	–	–	**54.6**
Coarse-to-Refine‡ (Scorer)	+SimScale	98.4	96.9	83.2	98.4	97.8	93.1	86.7	80.8	90.1	95.6	84.5	58.8	49.6
w/o Extra Data														
DiffusionDrive [32]	–	96.8	86.0	84.0	95.8	97.6	80.1	72.8	85.9	76.6	96.3	66.7	40.5	–
GuideFlow [33]	–	96.6	80.5	82.3	94.9	97.7	**87.3**	**76.7**	84.3	**85.1**	93.1	54.9	**47.9**	27.1
ReCogDrive‡ [29]	–	97.1	80.2	83.6	95.3	97.6	78.2	69.3	86.6	73.9	96.4	68.9	37.8	–
DriveFine‡⋆ [10]	–	**97.6**	90.0	84.9	**96.7**	97.6	82.1	71.3	88.1	74.3	**96.8**	74.4	41.0	–
Curious-VLA‡⋆ [6]	–	95.6	86.4	85.3	94.2	**97.8**	79.5	71.0	89.5	76.2	95.7	67.8	39.4	27.6
Coarse-to-Refine‡	–	97.4	**90.9**	**87.6**	96.0	**97.8**	79.7	71.7	**92.5**	75.5	94.8	**80.1**	41.6	**34.0**
w/ Extra Data														
LTF [9]	+SimScale	98.0	84.4	83.6	**97.8**	**97.8**	83.0	69.4	**91.4**	78.1	**95.2**	67.5	42.4	30.2
DiffusionDrive [32]	+SimScale	97.1	90.9	83.8	96.0	97.6	81.6	69.3	90.3	76.9	94.9	71.5	43.5	32.8
RAP-DINO [12]	+500k raster	97.1	**94.4**	83.9	96.9	96.4	83.2	**83.9**	86.9	80.4	**95.2**	–	–	39.6
Coarse-to-Refine‡	+SimScale	**98.2**	92.7	**85.2**	97.3	**97.8**	**90.7**	81.9	86.9	**87.3**	**95.2**	**80.9**	**53.3**	**42.8**



(a) Output formulation and reasoning type.

(b) Supervision target source.

Figure 3: SFT ablations on output formulation, reasoning type, and target source. (a) The grid reports PDMS (NAVSIM v1) / EPDMS† (NAVSIM v2). Act-only directly uses the trajectory target without reasoning. Prefix-Baseline uses direct LLM labeling without rule guidance. Prefix-Guided uses the guided labeling step of GSRS under Prefix Reasoning. C2R-Guided uses the full C2R supervision: coarse trajectory, refinement reasoning, and refined trajectory target. CR removes reasoning from C2R, and CPR replaces reasoning with pause tokens. Prefix-Refined keeps Prefix Reasoning but regenerates guided annotations for the refined trajectory target. (b) The target-source panel extends Prefix-Guided supervision to refined targets, yielding Prefix-Refined, CR/CPR, and C2R-Guided.

42.8 EPDMS, the first non-scorer result above 40 and surpassing RAP-DINO [12] despite using far less training data. This also suggests that C2R benefits from synthetic data scaling. *(iii)* **With scorer integration.** Paired with DrivoR [24] in a training-free setting, C2R yields **49.6 EPDMS**, showing that its trajectories retain quality under different evaluation.

5 Understanding Coarse-to-Refine

C2R achieves VLA state-of-the-art performance with a simple refinement formulation. We analyze this result through two questions. *(i) What drives C2R's IL improvement?* Section 5.1 shows that the coarse-to-refine formulation itself drives the IL gain, even with semantically empty pause tokens; the gain is consistent across base models and not explained by target source alone. This creates an apparent tension. *(ii) If explicit reasoning seems unnecessary for IL, does C2R still utilize it for final prediction?* Section 5.2 answers this through activation intervention, image reliance, and attention analyses, and shows RLVR needs explicit reasoning. More refinement analyses are in Appendix I.

Table 4: Coarse-trajectory source. (a) In G2, we replace human GT with greedy trajectories from π_{warmup} as τ^c, keep the same refined trajectories τ^f, and re-annotate refinement reasoning R^{ref} before repeating I2. (b) We evaluate the same final RL model on NavTest under standard inference and GT prefill. In GT prefill, the human-GT trajectory is supplied as τ^c in the response prefix; the model then generates R^{ref} and τ^f. Δ is refined PDMS minus coarse PDMS. Details are in Appendix I.

(a) SFT

Coarse source	PDMS ↑	EPDMS† ↑
Human GT	89.87	88.28
Generated by π_{warmup}	89.83	88.07

(b) Inference with GT prefill

Setting	Coarse PDMS ↑	Refined PDMS ↑	Δ
Standard inference	89.8	92.1	+2.3
GT prefill	94.4	94.8	+0.4

Table 5: Cross-backbone and RLVR ablations on NAVSIM v2. **(a)** Cross-backbone SFT comparison between C2R and Prefix Reasoning. **(b)** RLVR ablations start from the corresponding SFT model; — denotes collapse to repeated pause tokens with no valid trajectory. Both panels report EPDMS†; sub-metrics are in Appendix E.

(a) Cross-Backbone (SFT)

Backbone	Prefix	C2R
Qwen3-VL-4B	87.24	**88.28**
Qwen3-VL-8B	86.19	**87.51**
InternVL3.5-8B	85.04	**87.70**

(b) RLVR Ablation

Formulation	EPDMS†
C2R (Init after SFT)	88.28
Coarse-Pause-Refine	—
C2R + GRPO	89.60
C2R + SAPO	89.69
C2R + SWSAPO	**90.4**

5.1 Formulation-Driven Improvement

Output Formulation, Reasoning Type, and Target Source. At the IL stage, the ablation in Figure 3 analyses three distinct dimensions: output formulation, reasoning type, and target source. Across matched settings, C2R consistently outperforms Prefix Reasoning. To isolate explicit reasoning, we compare C2R with Coarse-Refine (CR), which removes reasoning, and Coarse-Pause-Refine (CPR), which keeps the refinement formulation but replaces reasoning with length-matched pause tokens [15]. Their comparable scores (88.72, 88.69, and 88.28 EPDMS†) show that most IL gain comes from the coarse-to-refine formulation, not from reasoning semantics alone. Figure 3(b) further shows that, with the same synthesized refined targets, C2R improves over Prefix Reasoning from 85.97 to 88.28 EPDMS†, so the gain is not simply due to better targets (Appendix E).

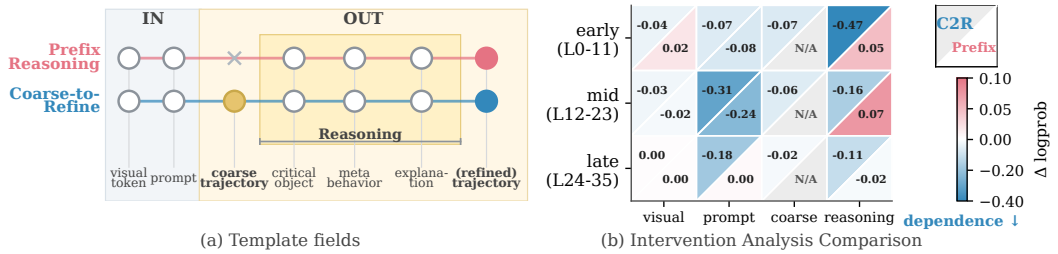
Coarse-Trajectory Source. Replacing human GT with trajectories generated by π_{warmup} as the coarse trajectories during SFT yields similar performance (Table 4(a)). At inference, C2R improves its own coarse trajectories while preserving the quality of human-GT coarse trajectories provided through GT prefill (Table 4(b)).

Cross-Backbone Consistency. C2R's improvement is consistent across backbones: it exceeds Prefix Reasoning by +1.04, +1.32, and +2.66 EPDMS† on Qwen3-VL-4B, Qwen3-VL-8B, and InternVL3.5-8B, respectively (Table 5(a)).

5.2 Functional Utilization of Reasoning

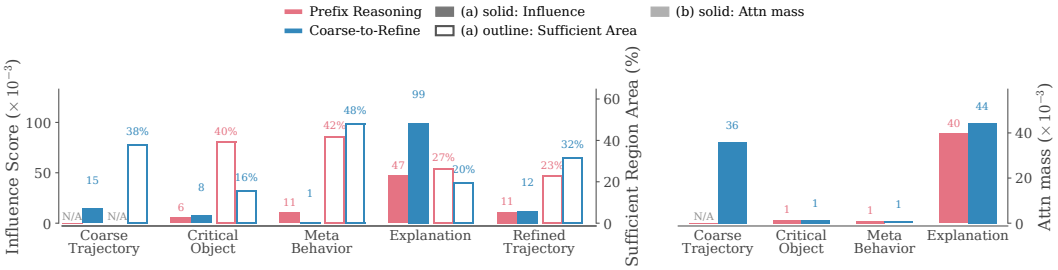
The IL ablations reveal that explicit reasoning contributes marginally to performance (Figure 3), raising a natural question: does C2R utilize explicit reasoning for final trajectory prediction?

Activation Intervention. Leveraging causal tracing [40], we intervene by zeroing hidden states of earlier fields and measuring how final-trajectory prediction changes (Figure 4). For C2R, intervening on refinement reasoning tokens causes clear drops across early, middle, and late layers (by -0.47, -0.16, and -0.11, respectively). The same intervention in Prefix Reasoning gives positive deltas in early and middle layers ($+0.05$ and $+0.07$) and remains near zero in late layers (-0.02). This



(a) Template fields (b) Intervention Analysis Comparison

Figure 4: Activation intervention analysis. (a) Field order before the final predicted trajectory. Prefix Reasoning generates reasoning before the trajectory; C2R inserts a coarse trajectory before reasoning and then predicts the refined trajectory. The reasoning field groups critical objects, meta behavior, and explanation. (b) Intervention results. For each field and decoder-layer range, we zero hidden states under teacher forcing and report the change in the final trajectory's mean log-probability. Qwen3-VL-4B's 36 decoder layers are grouped into early L0–11, middle L12–23, and late L24–35. C2R is shown in the upper-left triangle and Prefix Reasoning in the lower-right triangle; more negative values indicate stronger dependence on the intervened field.



(a) Influence Score and Sufficient Region Area (b) Prior-field Attention Distribution

Figure 5: Field-level image reliance and prior-field attention. (a) Image reliance of generated fields. Solid bars report the *influence score*: the log-probability gap between the original image and a blank image. Larger values indicate stronger visual reliance. Outlines report *sufficient region area*: the smallest visible area on a regular 4×4 grid that recovers 90% of the full-image score under an insight-score-guided search. (b) Attention to earlier generated fields. Bars report absolute decoder attention mass from final-trajectory tokens to earlier generated fields, averaged over late decoder layers and samples. Values are shown in 10^{-3} units and are not normalized across prior fields. In (a), C2R has a higher influence score on Explanation and a smaller sufficient-region area on Critical Object. In (b), C2R allocates visible attention mass to both Coarse Trajectory and Explanation.

indicates that C2R's trajectory prediction depends on refinement reasoning, whereas Prefix Reasoning doesn't show the same dependence.

Field-Level Image Reliance. C2R's refinement explanation is more visually grounded. We apply EAGLE-style image perturbations to each generated field (Figure 5(a)) [7]. The *influence score* compares log-probability under the original and blank image (*"What They Rely On"*). C2R's refinement explanation reaches roughly twice Prefix Reasoning's score (0.099 vs. 0.047). The *sufficient region area* finds the smallest visible region that recovers 90% of the full-image score (*"Where MLLMs Attend"*). For critical-object identification, C2R needs only 16% visible area, compared with 40% for Prefix Reasoning. Matched visualizations in Figure 6 show the same pattern.

Attention Allocation. We additionally investigate which earlier fields receive attention during final-trajectory next-token prediction (Figure 5(b)). For each earlier field, we aggregate decoder attention from final-trajectory tokens to its tokens, and average over late layers and samples. C2R nearly doubles the total attention to earlier fields relative to Prefix Reasoning (0.083 vs. 0.042). The increase is due to a newly introduced attention path to the coarse trajectory (0.036), while attention to the explanation remains slightly higher than in Prefix Reasoning (0.044 vs. 0.040). Thus, C2R incorporates both the coarse draft and explanation when predicting the final trajectory.

RLVR Relies on Explicit Reasoning. Finally, Table 5(b) investigates whether RLVR continues to yield improvements when reasoning is replaced by semantically empty pause tokens. RLVR on CPR

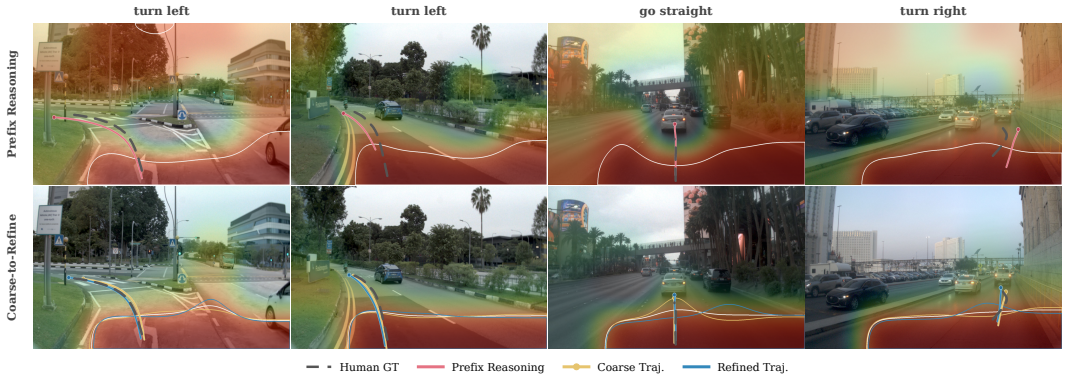


Figure 6: Sample visualizations for field-level image reliance. Each column shows a matched scene. Rows compare Prefix Reasoning and C2R. Trajectory overlays follow the legend below the figure. Heatmaps show field-level image-reliance maps. Additional samples are in Appendix J.

collapses to repeated pause tokens and produces no valid trajectory, showing that pause tokens do not provide usable tokens for RLVR optimization. C2R instead improves across RLVR variants and reaches 90.4 EPDMS† with SWSAPO. Together with the IL ablations, this separates the two roles: the refinement formulation accounts for most IL gain, whereas RLVR needs usable reasoning tokens.

6 Conclusion and Limitations

We have presented Coarse-to-Refine (C2R), an autoregressive self-refinement formulation for driving VLA that first drafts a coarse trajectory, uses explicit reasoning to assess the draft, and then outputs the refined trajectory. Controlled variants confirm that the C2R formulation drives IL improvement through refinement, mechanistic analyses verify that reasoning is functionally utilized in refined trajectory prediction, and explicit reasoning tokens are necessary for RLVR to improve trajectory quality. C2R achieves VLA state-of-the-art performance: 92.1 PDMS on NAVSIM v1, 90.4 EPDMS† on NAVSIM v2, and 49.6 EPDMS on NavHard. The current data construction employs rule-based scoring to synthesize refined trajectory targets, while incorporating learnable scorers for this step is left for future work.

Acknowledge

This work is funded by the National Natural Science Foundation of China under Grant No. 62576018 and No. 62550184, the Beijing Natural Science Foundation under Grant No. L244043 and L2607011, the Zhejiang Provincial Natural Science Foundation of China under Grant No. LD24F020007, DiDi and Wuxi Research Institute of Applied Technologies, Tsinghua University under Grant 20242001120.

References

[1] Sining Ang, Yuguang Yang, Canyu Chen, and Yan Wang. Clover: Closed-loop value estimation and ranking for end-to-end autonomous driving planning, 2026. URL `https://arxiv.org/abs/2605.15120`.

[2] Sining Ang, Yuguang Yang, Chenxu Dang, Canyu Chen, Cheng Chi, Haiyan Liu, Xuanyao Mao, Jason Bao, Xuliang, Bingchuan Sun, and Yan Wang. From representational complementarity to dual systems: Synergizing vlm and vision-only backbones for end-to-end driving, 2026. URL `https://arxiv.org/abs/2602.10719`.

[3] Holger Caesar, Juraj Kabzan, Kok Seang Tan, Whye Kit Fong, Eric Wolff, Alex Lang, Luke Fletcher, Oscar Beijbom, and Sammy Omari. Nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles. In *CVPR ADP3 workshop*, 2021.

[4] Jiajun Cao, Xiaoan Zhang, Xiaobao Wei, Liyuqiu Huang, Wang Zijian, Hanzhen Zhang, Zhengyu Jia, Wei Mao, Hao Wang, Xianming Liu, Shuchang Zhou, Yang Wang, and Shang-

hang Zhang. Evodrivevla: Evolving autonomous driving vision-language-action model via collaborative perception-planning distillation. *arXiv preprint arXiv:2603.09465*, 2026.

[5] Wei Cao, Marcel Hallgarten, Tianyu Li, Daniel Dauner, Xunjiang Gu, Caojun Wang, Yakov Miron, Marco Aiello, Hongyang Li, Igor Gilitschenski, Boris Ivanovic, Marco Pavone, Andreas Geiger, and Kashyap Chitta. Pseudo-simulation for autonomous driving. In *Proceedings of The 9th Conference on Robot Learning*, pages 4709–4722, 2025.

[6] Canyu Chen, Yuguang Yang, Zhewen Tan, Yizhi Wang, Ruiyi Zhan, Haiyan Liu, Xuanyao Mao, Jason Bao, Xinyue Tang, Linlin Yang, Bingchuan Sun, Yan Wang, and Baochang Zhang. Devil is in narrow policy: Unleashing exploration in driving vla models, 2026. URL `https://arxiv.org/abs/2603.06049`.

[7] Ruoyu Chen, Xiaoqing Guo, Kangwei Liu, Siyuan Liang, Shiming Liu, Qunli Zhang, Laiyuan Wang, Hua Zhang, and Xiaochun Cao. Where mllms attend and what they rely on: Explaining autoregressive token generation, 2026. URL `https://arxiv.org/abs/2509.22496`.

[8] Haohan Chi, Huan-ang Gao, Ziming Liu, Jianing Liu, Chenyu Liu, Jinwei Li, Kaisen Yang, Yangcheng Yu, Zeda Wang, Wenyi Li, et al. Impromptu vla: Open weights and open data for driving vision-language-action models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

[9] Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *IEEE transactions on pattern analysis and machine intelligence*, 45(11):12878–12895, 2022.

[10] Chenxu Dang, Sining Ang, Yongkang Li, Haochen Tian, Jie Wang, Guang Li, Hangjun Ye, Jie Ma, Long Chen, and Yan Wang. Drivefine: Refining-augmented masked diffusion vla for precise and robust driving. *arXiv preprint arXiv:2602.14577*, 2026.

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

[12] Lan Feng, Yang Gao, Eloi Zablocki, Quanyi Li, Wuyang Li, Sichao Liu, Matthieu Cord, and Alexandre Alahi. Rap: 3d rasterization augmented end-to-end planning, 2026. URL `https://arxiv.org/abs/2510.04333`.

[13] 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. *arXiv preprint arXiv:2503.19755*, 2025.

[14] Chang Gao, Chujie Zheng, Xiong-Hui Chen, Kai Dang, Shixuan Liu, Bowen Yu, An Yang, Shuai Bai, Jingren Zhou, and Junyang Lin. Soft adaptive policy optimization, 2025. URL `https://arxiv.org/abs/2511.20347`.

[15] Sachin Goyal, Ziwei Ji, Ankit Singh Rawat, Aditya Krishna Menon, Sanjiv Kumar, and Vaishnavh Nagarajan. Think before you speak: Training language models with pause tokens. In *International Conference on Learning Representations (ICLR)*, 2024.

[16] Ke Guo, Haochen Liu, Xiaojun Wu, Jia Pan, and Chen Lv. ipad: Iterative proposal-centric end-to-end autonomous driving. *arXiv preprint arXiv:2505.15111*, 2025.

[17] Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, et al. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 17853–17862, 2023.

[18] Chi-Pin Huang, Yueh-Hua Wu, Min-Hung Chen, Yu-Chiang Frank Wang, and Fu-En Yang. Thinkact: Vision-language-action reasoning via reinforced visual latent planning. *arXiv preprint arXiv:2507.16815*, 2025.

[19] Jyh-Jing Hwang, Runsheng Xu, Hubert Lin, Wei-Chih Hung, Jingwei Ji, Kristy Choi, Di Huang, Tong He, Paul Covington, Benjamin Sapp, et al. Emma: End-to-end multimodal model for autonomous driving. *Transactions on Machine Learning Research*, 2025.

[20] Xiaosong Jia, Junqi You, Zhiyuan Zhang, and Junchi Yan. Drivetransformer: Unified transformer for scalable end-to-end autonomous driving. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=M42KR4W9P5.

[21] Bo Jiang, Shaoyu Chen, Qing Xu, Bencheng Liao, Jiajie Chen, Helong Zhou, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. Vad: Vectorized scene representation for efficient autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8340–8350, 2023.

[22] 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.

[23] Ryo Kamoi, Yusen Zhang, Nan Zhang, Jiawei Han, and Rui Zhang. When can llms actually correct their own mistakes? a critical survey of self-correction of llms. *Transactions of the Association for Computational Linguistics*, 12:1417–1440, 2024.

[24] 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. *arXiv preprint arXiv:2601.05083*, 2026.

[25] Derun Li, Jianwei Ren, Yue Wang, Xin Wen, Pengxiang Li, Leimeng Xu, Kun Zhan, Zhongpu Xia, Peng Jia, Xianpeng Lang, et al. Finetuning generative trajectory model with reinforcement learning from human feedback. *arXiv preprint arXiv:2503.10434*, 2025.

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

[27] Yingyan Li, Shuyao Shang, Weisong Liu, Bing Zhan, Haochen Wang, Yuqi Wang, Yuntao Chen, Xiaoman Wang, Yasong An, Chufeng Tang, et al. Drivevla-w0: World models amplify data scaling law in autonomous driving. *arXiv preprint arXiv:2510.12796*, 2025.

[28] Yingyan Li, Yuqi Wang, Yang Liu, Jiawei He, Lue Fan, and Zhaoxiang Zhang. End-to-end driving with online trajectory evaluation via bev world model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 27137–27146, 2025.

[29] Yongkang Li, Kaixin Xiong, Xiangyu Guo, Fang Li, Sixu Yan, Gangwei Xu, Lijun Zhou, Long Chen, Haiyang Sun, Bing Wang, et al. Recogdrive: A reinforced cognitive framework for end-to-end autonomous driving. In *The Fourteenth International Conference on Learning Representations*, 2026.

[30] Zhenxin Li, Kailin Li, Shihao Wang, Shiyi Lan, Zhiding Yu, Yishen Ji, Zhiqi Li, Ziyue Zhu, Jan Kautz, Zuxuan Wu, et al. Hydra-mdp: End-to-end multimodal planning with multi-target hydra-distillation. *arXiv preprint arXiv:2406.06978*, 2024.

[31] Zhenxin Li, Wenhao Yao, Zi Wang, Xinglong Sun, Joshua Chen, Nadine Chang, Maying Shen, Zuxuan Wu, Shiyi Lan, and Jose M. Alvarez. Generalized trajectory scoring for end-to-end multimodal planning. *arXiv preprint arXiv:2506.06664*, 2025.

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

[33] Lin Liu, Caiyan Jia, Guanyi Yu, Ziying Song, JunQiao Li, Feiyang Jia, Peiliang Wu, Xiaoshuai Hao, and Yadan Luo. Guideflow: Constraint-guided flow matching for planning in end-to-end autonomous driving, 2026. URL https://arxiv.org/abs/2511.18729.

[34] Xueyi Liu, Zuodong Zhong, Qichao Zhang, Yuxin Guo, Yupeng Zheng, Junli Wang, Dongbin Zhao, Yun-Fu Liu, Zhiguo Su, Yinfeng Gao, Qiao Lin, and Chen Huiyong. Reasonplan: Unified scene prediction and decision reasoning for closed-loop autonomous driving. In *Proceedings of The 9th Conference on Robot Learning*, pages 3051–3068, 2025.

[35] Jinghui Lu, Jiayi Guan, Zhijian Huang, Jinlong Li, Guang Li, Lingdong Kong, Yingyan Li, Han Wang, Shaoqing Xu, Yuechen Luo, et al. Xiaomi OneVL: One-step latent reasoning and planning with vision-language explanation. *arXiv preprint arXiv:2604.18486*, 2026.

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

[37] Yuechen Luo, Qimao Chen, Fang Li, Shaoqing Xu, Jaxin Liu, Ziying Song, Zhi xin Yang, and Fuxi Wen. Unleashing vla potentials in autonomous driving via explicit learning from failures, 2026. URL `https://arxiv.org/abs/2603.01063`.

[38] Yuechen Luo, Fang Li, Shaoqing Xu, Yang Ji, Zehan Zhang, Bing Wang, Yuannan Shen, Jianwei Cui, Long Chen, Guang Chen, Hangjun Ye, Zhi-Xin Yang, and Fuxi Wen. Last-vla: Thinking in latent spatio-temporal space for vision-language-action in autonomous driving. *arXiv preprint arXiv:2603.01928*, 2026.

[39] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. In *Advances in Neural Information Processing Systems*, 2023.

[40] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372, 2022.

[41] Qihang Peng, Xuesong Chen, Chenye Yang, Shaoshuai Shi, and Hongsheng Li. Colavla: Leveraging cognitive latent reasoning for hierarchical parallel trajectory planning in autonomous driving. *arXiv preprint arXiv:2512.22939*, 2025.

[42] Ishaan Rawal, Shubh Gupta, Yihan Hu, and Wei Zhan. Nord: A data-efficient vision-language-action model that drives without reasoning. *arXiv preprint arXiv:2602.21172*, 2026.

[43] Katrin Renz, Long Chen, Ana-Maria Marcu, Jan Hünermann, Benoit Hanotte, Alice Karnsund, Jamie Shotton, Elahe Arani, and Oleg Sinavski. Carllava: Vision language models for camera-only closed-loop driving. *arXiv preprint arXiv:2406.10165*, 2024.

[44] Luke Rowe, Rodrigue de Schaetzen, Roger Girgis, Christopher Pal, and Liam Paull. Poutine: Vision-language-trajectory pre-training and reinforcement learning post-training enable robust end-to-end autonomous driving. *arXiv preprint arXiv:2506.11234*, 2025.

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

[46] Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023.

[47] Chonghao Sima, Katrin Renz, Kashyap Chitta, Li Chen, Hanxue Zhang, Chengen Xie, Jens Beißwenger, Ping Luo, Andreas Geiger, and Hongyang Li. Drivelm: Driving with graph visual question answering. In *European conference on computer vision*, pages 256–274. Springer, 2024.

[48] Shuhan Tan, Kashyap Chitta, Yuxiao Chen, Ran Tian, Yurong You, Yan Wang, Wenjie Luo, Yulong Cao, Philipp Krahenbuhl, Marco Pavone, and Boris Ivanovic. Latent chain-of-thought world modeling for end-to-end driving. 2026. URL `https://arxiv.org/abs/2512.10226`.

[49] Haochen Tian, Tianyu Li, Haochen Liu, Jiazhi Yang, Yihang Qiu, Guang Li, Junli Wang, Yinfeng Gao, Zhang Zhang, Liang Wang, Hangjun Ye, Tieniu Tan, Long Chen, and Hongyang Li. Simscale: Learning to drive via real-world simulation at scale, 2026. URL https://arxiv.org/abs/2511.23369.

[50] 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*, pages 4698–4726, 2025.

[51] Wei Wu, Xiaoxin Feng, Ziyan Gao, and Yuheng Kan. Smart: Scalable multi-agent real-time motion generation via next-token prediction. *Advances in Neural Information Processing Systems*, 37:114048–114071, 2024.

[52] Chengen Xie, Chonghao Sima, Tianyu Li, Bin Sun, Junjie Wu, Zhihui Hao, and Hongyang Li. Flare: Learning future-aware latent representations from vision-language models for autonomous driving. *arXiv preprint arXiv:2601.05611*, 2026.

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

[54] Zebin Xing, Xingyu Zhang, Yang Hu, Bo Jiang, Tong He, Qian Zhang, Xiaoxiao Long, and Wei Yin. Goalflow: Goal-driven flow matching for multimodal trajectories generation in end-to-end autonomous driving. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1602–1611, 2025.

[55] Pengxuan Yang, Ben Lu, Zhongpu Xia, Chao Han, Yinfeng Gao, Teng Zhang, Kun Zhan, XianPeng Lang, Yupeng Zheng, and Qichao Zhang. Worldrft: Latent world model planning with reinforcement fine-tuning for autonomous driving. *arXiv preprint arXiv:2512.19133*, 2025.

[56] Yuguang Yang, Canyu Chen, Zhewen Tan, Yizhi Wang, Zichao Feng, Chunyang Liu, Kehua Sheng, Juan Zhang, Linlin Yang, Baochang Zhang, Yan Wang, Bo Zhang, and Xianbin Cao. Teaching vision-language-action models what to see and where to look, 2026. URL https://arxiv.org/abs/2607.01658.

[57] Wenhao Yao, Zhenxin Li, Shiyi Lan, Zi Wang, Xinglong Sun, Jose M. Alvarez, and Zuxuan Wu. Drivesuprim: Towards precise trajectory selection for end-to-end planning, 2025. URL https://arxiv.org/abs/2506.06659.

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

[59] Yiwei Zhang, Xuesong Chen, Jin Gao, Hanshi Wang, Fudong Ge, Weiming Hu, Shaoshuai Shi, and Zhipeng Zhang. Onedrive: Unified multi-paradigm driving with vision-language-action models, 2026. URL https://arxiv.org/abs/2604.17915.

[60] Rui Zhao, Qirui Yuan, Jinyu Li, Haofeng Hu, Yun Li, Zhenhai Gao, and Fei Gao. Sce2drivex: A generalized mllm framework for scene-to-drive learning. *IEEE Robotics and Automation Letters*, 2025.

[61] Yupeng Zheng, Pengxuan Yang, Zebin Xing, Qichao Zhang, Yuhang Zheng, Yinfeng Gao, Pengfei Li, Teng Zhang, Zhongpu Xia, Peng Jia, and Dongbin Zhao. World4drive: End-to-end autonomous driving via intention-aware physical latent world model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

[62] Zhiyu Zheng, Shaoyu Chen, Haoran Yin, Xinbang Zhang, Jialv Zou, Xinggang Wang, Qian Zhang, and Lefei Zhang. Resad: Normalized residual trajectory modeling for end-to-end autonomous driving. *arXiv preprint arXiv:2510.08562*, 2025.

[63] Linqing Zhong, Yi Liu, Yifei Wei, Ziyu Xiong, Maoqing Yao, Si Liu, and Guanghui Ren. Acot-vla: Action chain-of-thought for vision-language-action models. *arXiv preprint arXiv:2601.11404*, 2026.

[64] Xingcheng Zhou, Xuyuan Han, Feng Yang, Yunpu Ma, Volker Tresp, and Alois C. Knoll. Opendrivevla: Towards end-to-end autonomous driving with large vision language action model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026.

[65] 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.

[66] Jialv Zou, Shaoyu Chen, Bencheng Liao, Zhiyu Zheng, Yuehao Song, Lefei Zhang, Qian Zhang, Wenyu Liu, and Xinggang Wang. Diffusiondrivev2: Reinforcement learning-constrained truncated diffusion modeling in end-to-end a utonomous driving, 2025. URL `https://arxiv.org/abs/2512.07745`.

A LLM Usage Declaration

Qwen3-VL-235B is used as a core component of the Guided Self-Refinement Synthesis (GSRS) pipeline (Section 3.2). It annotates chain-of-thought reasoning fields — critical objects, meta behaviour, action explanation, and refinement explanation — for constructing supervised fine-tuning data on the training split. This annotation is performed offline on training samples only; no large language model is invoked at inference time. Beyond data synthesis, LLMs were used to assist with paper writing and editing.

B Guided Self-Refinement Synthesis Details

GSRS builds the C2R supervision that is not directly available in standard trajectory annotations. As summarized in Figure 7, GSRS has two data-construction stages: **G1: Prefix Annotation** builds Prefix Data from human trajectory annotations, and **G2: C2R Synthesis** turns warmup-policy candidates into $(\tau^c, R^{\text{ref}}, \tau^f)$ supervision. The two SFT passes use these data products: Warmup SFT learns π_{warmup} from Prefix Data, and C2R SFT trains the final C2R response format.

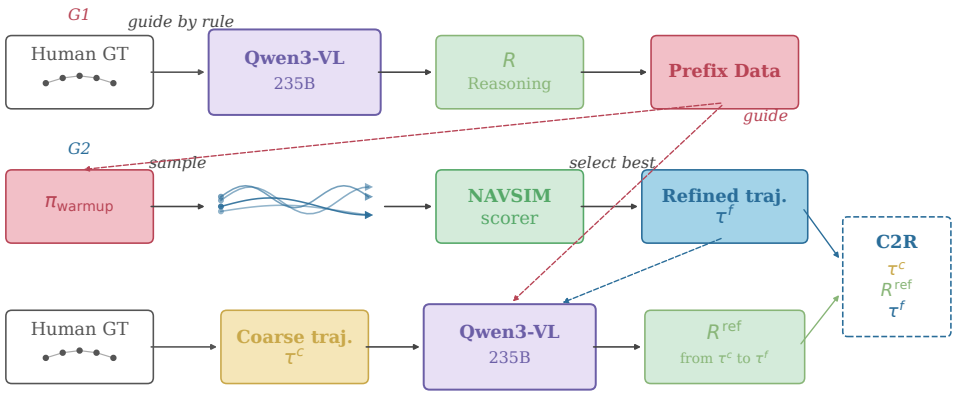


Figure 7: Detailed GSRS data construction. In G1, rule guidance is provided to Qwen3-VL-235B, which annotates Prefix reasoning from human GT trajectories to form Prefix Data. In G2, the warmup policy π_{warmup} samples trajectory candidates, the NAVSIM scorer selects the refined trajectory τ^f, and the human GT trajectory is used as the coarse trajectory τ^c. Prefix Data guides Qwen3-VL-235B again to label refinement reasoning R^{ref} from τ^c to τ^f, producing C2R supervision $(\tau^c, R^{\text{ref}}, \tau^f)$.

B.1 GSRS Procedure and Label Fields

G1: Prefix Annotation. For each `navtrain` scene, G1 starts from rule-based proposals and asks Qwen3-VL-235B to correct them using the front-view image, high-level command, 1.5-second ego history, and human trajectory target. The rules are intentionally simple: speed is estimated from trajectory velocity change, command is mapped from the navigation command, and critical-object labels are treated as visual proposals to be verified by the VLM. Specifically, the speed proposal uses $v_{\text{end}} - v_{\text{start}}$ from the human trajectory, and the command proposal is mapped from NAVSIM's ground-truth `ego_status.driving_command`. The rules are used only for offline annotation and are fixed before training, without tuning on validation or test results.

Under the same Prefix formulation and human-trajectory target, rule-guided labeling improves both PDMS and EPDMS† over direct labeling (Table 12). The VLM revises rather than directly copies the proposals: the keep-speed share changes from 64.64% in rule proposals to 48.86% in G1 outputs and 48.14% in the final C2R data (Table 7).

The resulting Prefix Reasoning annotation contains three fields: `critical_objects`, `meta_behaviour`, and `explanation`. The `critical_objects` field audits twelve object or condition classes, including vehicles, pedestrians, cyclists, construction, traffic elements, road hazards, and special cases such as emergency or door-opening vehicles. The `meta_behaviour` field assigns one speed label (`keep`, `accelerate`, or `decelerate`) and one command label (`straight`, `yield`, `left_turn`, `right_turn`, `lane_follow`, `lane_change_left`, `lane_change_right`, or `reverse`). The explanation then gives the driving rationale behind the trajectory target.

G2: C2R Synthesis. After Warmup SFT trains π_{warmup} on Prefix Data, G2 samples trajectory candidates from π_{warmup} on the training split, scores them with the NAVSIM scorer, and selects the best candidate as the refined trajectory τ^f. The human trajectory is used as the coarse trajectory τ^c. Here "coarse" denotes its position as the first trajectory in the C2R response. Using the human trajectory keeps the coarse field tied to the same trajectory target used in warmup training, rather than using a sampled candidate as τ^c.

Qwen3-VL-235B then labels the refinement reasoning R^{ref} from τ^c to τ^f. This annotation is also guided: the Prefix Data from G1 provide proposals for `critical_objects` and `meta_behaviour`, and the VLM checks or revises them according to the image and the coarse-to-refined trajectory change.

Refined-Trajectory Quality. Table 6 compares the PDMS of human GT and the refined trajectories τ^f selected in G2 on the same scenes from the GSRS training split. The refined trajectories have higher mean and median PDMS, with a larger proportion scoring at least 95.

Table 6: PDMS distributions of human GT and the refined trajectories τ^f selected in G2. Trajectories are paired by scene token on the GSRS training split. PDMS is reported on a 0–100 scale; the last column gives the percentage of trajectories with PDMS at least 95.

Trajectory	Mean PDMS	Median PDMS	10th percentile	PDMS ≥ 95 (%)
Human GT	94.10	95.94	85.71	54.58
Refined trajectory τ^f	96.41	98.47	91.34	87.99

B.2 C2R Labeling Prompt

Figure 8 reports the C2R refinement labeling prompt. The prompt, image, trajectory pair, reference proposals, and example output are kept in one box because the annotation task is to explain a specific trajectory change using the visible scene.

Prompt and Example for C2R Refinement Labeling

Question.
You are an expert autonomous driver. Given a front-view image, high-level intent, past ego trajectory, a **Coarse Trajectory**, a **Refined Trajectory**, and reference proposals, generate the reasoning that bridges the coarse trajectory to the refined trajectory.

Labeling instructions.

1. Identify critical objects or conditions that justify the trajectory change.
2. Label the behavior of the refined trajectory with one speed category and one command category.
3. Explain the change by contrasting the coarse motion, the visual evidence, and the refined motion.

Reference proposals.
The provided `critical_objects` and `meta_behaviour` fields are proposals from Prefix Reasoning. Check and correct them; do not copy them blindly.

Output format.
Return strict JSON with exactly three fields: `critical_objects`, `meta_behaviour`, and `explanation`.



Example input snippet.
Intent: go straight.
Coarse endpoint: $(7.81, 0.05, 0.00)$.
Refined endpoint: $(8.49, 0.06, 0.01)$.
Full trajectory pair: eight raw-scale ego-frame waypoints.

Example output snippet.
`critical_objects`: nearby vehicle = yes, traffic element = yes.
`meta_behaviour`: speed = accelerate, command = straight.
`explanation`: the coarse plan is conservative, while the visible vehicle ahead and green traffic signal support smooth forward acceleration within the lane.

Figure 8: C2R refinement labeling prompt and example. The example uses raw-scale ego-frame coordinates from the same scene used in Figure 9.

B.3 Data Sources and Statistics

Table 7 summarizes the data variants used by GSRS and its ablations. The final C2R training data are `C2R-Guided-by-Prefix`, whose refined trajectory targets are selected from warmup-policy candidates and whose reasoning fields are guided by G1 Prefix Data. In this set, the most frequent positive critical-object labels are `traffic_element` (86.06%), `nearby_vehicle` (82.73%), and `conflicting_vehicle` (65.29%). The main command labels are `straight` (46.92%), `left_turn` (20.53%), `lane_follow` (20.27%), and `right_turn` (11.76%).

Table 7: GSRS data variants. Target is `human` for human trajectory targets and `refined` for scorer-selected refined trajectory targets. Speed labels are reported in separate columns.

Dataset	Role	Target	Keep	Accel.	Decel.
Prefix-Baseline	direct labels	human	15.11	41.58	43.31
Rule proposals	heuristic seed	human	64.64	16.91	18.46
Prefix-Guided	G1 output	human	48.86	30.27	20.87
C2R-Guided-by-Rule	ablation	refined	63.29	17.35	19.36
C2R-Guided-by-Prefix	**final C2R**	**refined**	**48.14**	**30.29**	**21.57**

B.4 Response-Format Examples

Figure 9 shows raw-scale response snippets from one training scene. Prefix Reasoning emits reasoning fields before the future trajectory. C2R emits the coarse trajectory, then the refinement reasoning, and finally the refined trajectory. The trajectory values are copied from raw-scale sample files and shown as plain field values.

Raw-Scale Response Examples on the Same Scene



Prefix Reasoning response.
Response fields: critical objects → meta behaviour → explanation → future trajectory.
Snippet: `critical_objects`: nearby vehicle = yes, traffic element = yes.
`meta_behaviour`: speed = accelerate, command = straight.
`explanation`: the ego vehicle accelerates straight ahead because nearby vehicles maintain lane position and the traffic signal permits forward movement.

$$\tau = [(0.00, 0.00, 0.00), (0.03, 0.00, 0.00), (0.30, 0.00, 0.00), (1.01, 0.00, 0.00), (2.15, 0.00, 0.00), (3.67, 0.02, 0.00), (5.57, 0.03, 0.00), (7.81, 0.05, 0.00)].$$

C2R response.
Response fields: coarse trajectory → critical objects → meta behaviour → explanation → refined trajectory.
Snippet:

$$\tau^c = [(0.00, 0.00, 0.00), (0.03, 0.00, 0.00), (0.30, 0.00, 0.00), (1.01, 0.00, 0.00), (2.15, 0.00, 0.00), (3.67, 0.02, 0.00), (5.57, 0.03, 0.00), (7.81, 0.05, 0.00)].$$

`critical_objects`: nearby vehicle = yes, traffic element = yes.
`meta_behaviour`: speed = accelerate, command = straight.
`explanation`: the coarse plan is conservative; the refined plan moves farther forward while staying in lane, matching the visible vehicle and traffic signal.

$$\tau^f = [(0.00, 0.00, 0.00), (0.13, 0.00, 0.00), (0.50, 0.00, 0.00), (1.24, 0.01, 0.00), (2.43, 0.01, 0.00), (3.99, 0.01, 0.01), (6.03, 0.02, 0.01), (8.49, 0.06, 0.01)].$$

Figure 9: Raw-scale examples of Prefix Reasoning and C2R response formats. The two snippets use the same scene and are shortened only in the natural-language fields; all eight displayed waypoints are kept.

C Metric Definitions

We evaluate on the NAVSIM benchmark [11] using the PDM Score (PDMS) for v1 and the Extended PDM Score (EPDMS) for v2 [5].

PDMS (PDM Score). PDMS is the composite planning metric of NAVSIM v1, consisting of five sub-metrics divided into penalty terms and weighted-average terms. No at-fault Collision (NC) and Drivable Area Compliance (DAC) are penalty terms that enter multiplicatively; Time-to-Collision (TTC), Ego Progress (EP), and Comfort (C) form the weighted average:

$$\text{PDMS} = \text{NC} \cdot \text{DAC} \cdot \frac{5 \cdot \text{TTC} + 5 \cdot \text{EP} + 2 \cdot \text{C}}{12}. \tag{5}$$

EPDMS (Extended PDM Score). EPDMS is introduced in NAVSIM v2 [5], extending PDMS from 5 to 9 sub-metrics. It adds two penalty terms — Driving Direction Compliance (DDC) and Traffic Light Compliance (TLC) — and extends the weighted-average component with Lane Keeping (LK), History Comfort (HC), and Ego Comfort (EC), while replacing the original Comfort:

$$\text{EPDMS} = \text{NC} \cdot \text{DAC} \cdot \text{DDC} \cdot \text{TLC} \cdot \frac{5 \cdot \text{EP} + 5 \cdot \text{TTC} + 2 \cdot \text{LK} + 2 \cdot \text{HC} + 2 \cdot \text{EC}}{16}. \tag{6}$$

All sub-metrics are false-positive-filtered: a violation is not penalized when the human driver also violates the same rule in the logged scenario.

EPDMS also uses a different normalization for ego progress, while History Comfort evaluates smoothness across the history–prediction trajectory.

EPDMS† (Bug-fixed Metric). After the initial release of NAVSIM v2, the authors identified and fixed a bug in the downstream aggregation used for EPDMS reporting. EPDMS† denotes the bug-fixed EPDMS metric released by the NAVSIM v2 authors. We report both the original EPDMS and the bug-fixed EPDMS† in the main comparison (Table 2), and use EPDMS† exclusively in all ablation studies (Section 5).

NavHard. NavHard [5] is a two-stage pseudo-closed-loop benchmark for safety-critical scenarios. Stage 1 evaluates planning on real observations while background vehicles react via the Intelligent Driver Model (IDM), producing an EPDMS score s_1. Stage 2 perturbs the ego pose and renders synthetic observations, testing robustness under distribution shift. Each initial scene yields multiple follow-up evaluations; their EPDMS scores are aggregated into s_2 via Gaussian-weighted average:

$$s_2 = \sum_i \hat{w}^{\,i}\, s_2^i, \qquad \hat{w}^{\,i} = \frac{w^i}{\sum_j w^j}, \qquad w^i = \exp\Big(-\frac{\|x^i - \hat{x}\|^2}{2\sigma^2}\Big), \tag{7}$$

where x^i is the perturbed ego state, $\hat{x}$ the original logged state, and $\sigma^2 = 0.1$ by default. The combined score for each initial scene is $s_{\text{combined}} = s_1 \cdot s_2$; the final NavHard metric averages s_{combined} over all initial scenes in the evaluation set.

D Implementation Details

We summarize the implementation settings used for the reported C2R experiments. C2R is built on Qwen3-VL-4B and uses a single front-view camera image, the high-level navigation command, and ego-state context as input. The future trajectory covers four seconds at 2 Hz, giving eight ego-frame waypoints. During training and inference, waypoints are serialized as ordinary text tokens, and NAVSIM scores are computed from the parsed final trajectory.

Table 8: Model and input configuration.

Setting	Value
Backbone	Qwen3-VL-4B
Camera input	single front-view camera
Driving context	navigation command, ego state, and 1.5 s ego history
Trajectory horizon	4 s, 2 Hz, 8 waypoints
Waypoint format	text-serialized ego-frame waypoints
Trajectory to evaluate	refined trajectory τ^f only

Supervised Fine-tuning. The IL implementation follows the G1/I1/G2/I2 pipeline described in Section 3.2. **G1: Prefix Annotation** produces Prefix Data for **I1: Warmup SFT**. **G2: C2R Synthesis** then produces $(\tau^c, R^{\text{ref}}, \tau^f)$ supervision for **I2: C2R SFT**, which initializes from the I1 warmup checkpoint. Both SFT passes update all model modules in the reported setting. Table 9 reports the shared and pass-specific hyperparameters.

Table 9: SFT hyperparameters.

Setting	I1: Warmup SFT	I2: C2R SFT
Supervision source	G1 Prefix Data, ∼103K navtrain	G2 C2R supervision, ∼103K navtrain
Response format	R, τ	$\tau^c, R^{\text{ref}}, \tau^f$
Initialization	Qwen3-VL-4B	I1 warmup checkpoint
Trainable modules	all modules	all modules
Epochs	3	3
Effective global batch size	32	32
Learning rate	4.0×10^{-5}	4.0×10^{-5}
Precision	bfloat16	bfloat16

RLVR and SWSAPO. RLVR initializes from the I2 C2R SFT checkpoint and optimizes only the parsed refined trajectory τ^f with the NAVSIM verified reward. Invalid or unparsable responses receive zero reward. We use group-relative advantage estimation, SAPO-style soft gating, and segment weights so that tokens closer to the rewarded final trajectory receive larger update weights.

Table 10: RLVR/SWSAPO hyperparameters for the reported C2R model.

Setting	Value
Initial policy	I2 C2R SFT checkpoint
Reward target	parsed refined trajectory τ^f
Reward metric	NAVSIM PDMS
Actor learning rate	1.0×10^{-6}
Segment weights (w_c, w_r, w_f)	0.25 / 0.70 / 1.60
Precision	bfloat16

ADAS Exploration. Adaptive Diversity-Aware Sampling is used only to choose informative RL training scenes. For each ADAS round, we sample repeated trajectories from the current policy on training scenes, score the rollouts with the NAVSIM scorer, compute per-scene reward statistics, and retain scenes whose rollout group has non-trivial score variance and mixed outcomes. These filtered scene identifiers are passed to the RL dataloader; ADAS does not introduce new labels or change the reward definition, and it does not add an external planner.

Inference and Compute. At inference time, the model generates a single C2R response and the planner uses only the parsed τ^f. No external planner and no multi-turn refinement loop are used. Repeated rollouts are used for training-scene filtering or separate analysis, not as an additional refinement loop in the C2R policy itself. We report latency and throughput comparisons separately in the efficiency analysis.

Table 11: Compute settings.

Setting	Value
Training hardware	one 8×H100 GPU node
Training compute	SFT: about 62 H100-hours; RLVR: about 144 H100-hours

E Formulation Sub-metric Results

Tables 12–14 report the NAVSIM v2 breakdown for the formulation ablations summarized in the main formulation figure and table. The sub-metric columns follow the NAVSIM v2 definitions, and EPDMS† uses the bug-fixed EPDMS metric that corrects the downstream aggregation used for EPDMS reporting. For the SFT formulation variants, the rightmost column additionally reports the final PDMS from a NAVSIM v1 evaluation.

The dash in the pause-token RLVR row denotes the collapsed run that produces no valid trajectory.

Table 12: SFT formulation breakdown on NAVSIM v2. NC: No at-fault Collision, DAC: Drivable Area Compliance, DDC: Driving Direction Compliance, TLC: Traffic Light Compliance, EP: Ego Progress, TTC: Time-to-Collision, LK: Lane Keeping, HC: History Comfort, and EC: Ego Comfort. NC through EC are NAVSIM v2 sub-metrics. EPDMS† is the bug-fixed EPDMS metric that corrects the downstream aggregation used for EPDMS reporting. The rightmost PDMS column reports the final score from an additional NAVSIM v1 evaluation. All values are percentages rounded to one decimal place.

Backbone	Variant	NC↑	DAC↑	DDC↑	TLC↑	EP↑	TTC↑	LK↑	HC↑	EC↑	EPDMS†↑	PDMS↑
Qwen3-VL-4B	A1 Act-only	98.3	96.6	99.5	99.8	87.1	97.7	97.7	98.3	79.3	87.9	88.4
Qwen3-VL-4B	A2 Prefix-Baseline	97.8	95.1	99.3	99.8	87.0	97.3	97.1	98.3	76.0	85.7	86.5
Qwen3-VL-4B	A3 Prefix-Guided	98.3	95.7	99.4	99.9	87.1	97.5	97.0	98.3	83.2	87.2	87.6
Qwen3-VL-4B	A4 Coarse-Refine (CR)	97.4	96.8	99.1	99.7	93.0	96.6	97.3	98.1	80.6	88.7	90.0
Qwen3-VL-4B	A5 Coarse-Pause-Refine (CPR)	97.2	96.8	99.1	99.7	93.2	96.5	97.3	98.1	81.3	88.7	90.1
Qwen3-VL-4B	A6 C2R-Guided	97.3	96.2	99.0	99.8	93.1	96.6	96.7	98.1	79.3	88.3	89.9
Qwen3-VL-4B	A7 Prefix-Refined	96.9	95.1	98.7	99.7	92.9	96.0	96.3	98.1	75.4	86.0	87.1

Table 13: Cross-backbone SFT breakdown on NAVSIM v2. NC through EC are NAVSIM v2 sub-metrics. EPDMS† is the bug-fixed EPDMS metric that corrects the downstream aggregation used for EPDMS reporting. All values are percentages rounded to one decimal place.

Backbone	Variant	NC↑	DAC↑	DDC↑	TLC↑	EP↑	TTC↑	LK↑	HC↑	EC↑	EPDMS†↑
Qwen3-VL-4B	Prefix Reasoning	98.3	95.7	99.4	99.9	87.1	97.5	97.0	98.3	83.2	87.2
Qwen3-VL-4B	C2R	97.3	96.2	99.0	99.8	93.1	96.6	96.7	98.1	79.3	88.3
Qwen3-VL-8B	Prefix Reasoning	97.9	95.2	99.3	99.9	87.1	97.2	96.9	98.3	81.4	86.2
Qwen3-VL-8B	C2R	96.8	96.2	99.0	99.7	93.1	96.0	96.9	98.1	79.2	87.5
InternVL3.5-8B	Prefix Reasoning	98.2	94.1	99.0	99.9	85.9	97.6	96.2	98.4	80.6	85.0
InternVL3.5-8B	C2R	96.9	96.3	99.0	99.7	93.3	96.1	97.0	98.1	80.6	87.7

Table 14: RLVR breakdown on NAVSIM v2. NC through EC are NAVSIM v2 sub-metrics. EPDMS† is the bug-fixed EPDMS metric that corrects the downstream aggregation used for EPDMS reporting. — denotes the collapsed pause-token run with no valid trajectory. All values are percentages rounded to one decimal place.

Backbone	Variant	Training	NC↑	DAC↑	DDC↑	TLC↑	EP↑	TTC↑	LK↑	HC↑	EC↑	EPDMS†↑
Qwen3-VL-4B	C2R	Init after SFT	97.3	96.2	99.0	99.8	93.1	96.6	96.7	98.1	79.3	88.3
Qwen3-VL-4B	Coarse-Pause-Refine (CPR)	RLVR	—	—	—	—	—	—	—	—	—	—
Qwen3-VL-4B	C2R + GRPO	RLVR	98.6	97.2	99.0	99.8	91.5	97.9	96.8	98.1	80.3	89.6
Qwen3-VL-4B	C2R + SAPO	RLVR	98.3	97.4	98.9	99.8	92.5	97.5	96.7	98.1	80.7	89.7
Qwen3-VL-4B	C2R + SWSAPO	RLVR	98.9	97.9	99.0	99.9	91.1	98.3	96.4	98.1	80.2	90.4

SFT and RLVR. Table 15 reports Prefix-Guided (A3), Prefix-Refined (A7), and C2R-Guided (A6), following the settings in Table 12. For Prefix-Refined, Prefix reasoning is re-annotated for the refined trajectory τ^f. RLVR starts from the corresponding SFT checkpoint for A7 and A6, using the same NAVSIM reward and training recipe. C2R-Guided achieves higher scores than Prefix-Refined after both SFT and RLVR.

Table 15: SFT and RLVR results for A3, A7, and A6. A3 uses human GT as the SFT trajectory target; A7 and A6 use the same refined trajectories τ^f selected in G2.

		SFT		RLVR	
Setting	SFT trajectory target	PDMS	EPDMS†	PDMS	EPDMS†
A3 Prefix-Guided	Human GT	87.57	87.24	—	—
A7 Prefix-Refined	Refined trajectory τ^f	87.06	85.97	90.70	88.25
A6 C2R-Guided	Refined trajectory τ^f	89.87	88.28	92.10	90.40

Table 16: Full sub-metric breakdown on NavHard. Gray rows denote scorer-based methods. ⋆ concurrent work; ‡ VLA-based. NC: No at-fault Collision, DAC: Drivable Area Compliance, DDC: Driving Direction Compliance, TLC: Traffic Light Compliance, EP: Ego Progress, TTC: Time-to-Collision, LK: Lane Keeping, HC: History Comfort, EC: Ego Comfort, S.: stage EPDMS.

Method	Extra Data	Stage	NC↑	DAC↑	DDC↑	TLC↑	EP↑	TTC↑	LK↑	HC↑	EC↑	S.↑	EPDMS↑
PDM-Closed	–	S1	94.4	98.8	100.0	99.5	100.0	93.5	99.3	97.7	36.0	–	51.3
		S2	88.1	90.6	96.3	98.5	100.0	83.1	73.7	91.5	25.4	–	
Scorer-Based													
GTRS-D [31]	–	S1	98.9	94.9	99.1	100.0	76.1	98.4	93.8	94.9	37.8	–	41.9
		S2	89.9	90.5	94.1	99.3	77.6	88.5	56.0	92.0	30.2	–	
GuideFlow-S [33]	–	S1	98.8	95.5	99.1	99.5	76.0	99.1	94.4	97.5	52.4	–	43.0
		S2	88.8	89.4	89.4	98.8	80.3	86.7	52.9	96.9	56.9	–	
DriveSuprim [57]	–	S1	98.7	98.0	99.1	99.8	75.9	98.7	94.7	97.6	49.8	–	44.7
		S2	89.5	89.6	92.9	98.5	78.9	86.4	55.3	96.5	52.7	–	
GTRS-D [31]	+SimScale	S1	99.1	98.2	99.8	100.0	71.9	99.3	95.6	93.8	28.0	–	47.2
		S2	92.3	93.8	94.9	99.3	75.4	90.7	60.3	95.9	37.4	–	
DrivoR [24]	–	S1	98.8	95.1	98.9	100.0	72.6	98.7	94.0	97.6	73.3	–	48.3
		S2	90.2	88.4	91.9	98.6	70.0	88.0	50.1	98.5	76.2	–	
DrivoR [24]	+SimScale	S1	99.1	98.2	99.3	99.8	75.4	98.7	94.9	97.6	70.2	–	54.6
		S2	92.3	91.6	97.3	99.1	75.7	90.6	56.1	98.4	44.7	–	
C2R‡ (+Scorer)	+SimScale	S1	98.4	96.9	99.4	99.3	83.2	98.4	95.3	97.8	68.9	84.5	49.6
		S2	93.1	86.7	94.6	98.8	80.8	90.1	54.4	95.6	34.5	58.8	
w/o Extra Data													
DiffusionDrive [32]	–	S1	96.8	86.0	98.8	99.3	84.0	95.8	96.7	97.6	79.6	66.7	–
		S2	80.1	72.8	84.4	98.4	85.9	76.6	46.4	96.3	72.8	40.5	
GuideFlow [33]	–	S1	96.6	80.5	96.3	99.3	82.3	94.9	91.5	97.7	67.8	54.9	27.1
		S2	87.3	76.7	88.8	99.2	84.3	85.1	49.7	93.1	44.5	47.9	
ReCogDrive‡ [29]	–	S1	97.1	80.2	98.4	100.0	83.6	95.3	93.6	97.6	76.0	68.9	–
		S2	78.2	69.3	83.9	98.2	86.6	73.9	44.1	96.4	72.0	37.8	
DriveFine‡⋆ [10]	–	S1	97.6	90.0	99.1	99.3	84.9	96.7	97.3	97.6	72.0	74.4	–
		S2	82.1	71.3	84.8	98.4	88.1	74.3	47.2	96.8	72.8	41.0	
Curious-VLA‡⋆ [6]	–	S1	95.6	86.4	98.0	99.3	85.3	94.2	94.9	97.8	71.6	67.8	27.6
		S2	79.5	71.0	80.7	98.7	89.5	76.2	45.1	95.7	64.0	39.4	
C2R‡	–	S1	97.4	90.9	98.3	99.8	87.6	96.0	96.7	97.8	68.0	**80.1**	**34.0**
		S2	79.7	71.7	83.1	98.6	92.5	75.5	47.3	94.8	64.2	41.6	
w/ Extra Data													
LTF [9]	+SimScale	S1	98.0	84.4	98.9	99.6	83.6	97.8	96.2	97.8	80.9	67.5	30.2
		S2	83.0	69.4	90.8	98.7	91.4	78.1	59.4	95.2	43.1	42.4	
DiffusionDrive [32]	+SimScale	S1	97.1	90.9	99.0	99.3	83.8	96.0	96.7	97.6	80.4	71.5	32.8
		S2	81.6	69.3	92.9	98.2	90.3	76.9	62.8	94.9	38.0	43.5	
RAP-DINO [12]	+500k raster	S1	97.1	94.4	98.8	99.8	83.9	96.9	94.7	96.4	66.2	–	39.6
		S2	83.2	83.9	87.4	98.0	86.9	80.4	52.3	95.2	52.4	–	
C2R‡	+SimScale	S1	98.2	92.7	98.9	99.6	85.2	97.3	95.6	97.8	71.1	**80.9**	**42.8**
		S2	90.7	81.9	92.5	98.9	86.9	87.3	55.3	95.2	32.2	**53.3**	

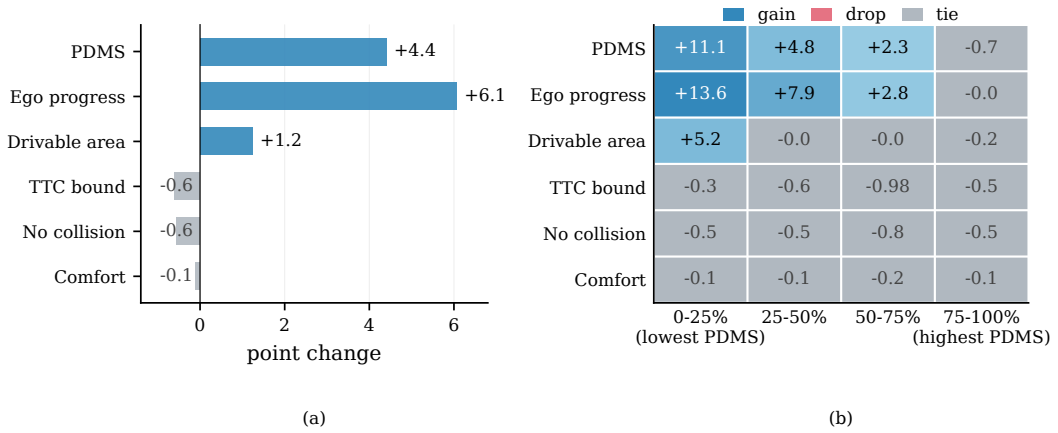
F NavHard Sub-metric Results

Table 16 provides the full per-stage sub-metric results for all methods evaluated on NavHard [5], complementing the compact comparison in Table 3. Each method occupies two rows (Stage 1 and Stage 2).

Among the sub-metrics, EC (Ego Comfort) exhibits the largest variance across methods and stages, frequently dominating the final EPDMS. Stage 2 scores are consistently lower than Stage 1, with DAC and LK degrading most under the perturbed-observation setting.

I Behavior of Refinement

Adaptive Refinement. Comparing coarse and refined trajectories within each C2R sample (Figure 10), refinement acts as targeted correction rather than uniform perturbation. When scenes are grouped by initial coarse-trajectory PDMS percentile, the lowest-PDMS group receives the largest score gain while the highest-PDMS group shows near-zero change, confirming that the model corrects weak coarse proposals and preserves adequate ones. The gain concentrates on ego progress; safety-related components (TTC, collision, comfort) remain nearly unchanged. This selective behavior



(a) (b)

Figure 10: Coarse-to-refine mechanism analysis. (a) Scene-level component deltas (refined − coarse) on a 0–100 point scale; gray bars denote nearly unchanged components ($|\Delta| < 1$). Components not included in the official PDMS computation (lane keeping, driving direction, traffic light) are excluded. (b) Component deltas grouped by initial coarse-trajectory PDMS percentile. The lowest-PDMS coarse trajectories (0–25%) receive the strongest correction, while the highest-PDMS ones (75–100%) are largely preserved.

confirms that the coarse trajectory carries informative structure that the refinement stage builds upon. This is an emergent behavior, not an explicit preservation mechanism: neither GSRS nor the reward includes a copy gate, preservation loss, or rule for retaining a near-optimal coarse trajectory.

Coarse Source during SFT. For Table 4(a), we generate τ^c from π_{warmup} using greedy decoding (temperature $= 0$, top-p $= 1$). We keep the refined trajectories τ^f selected in G2 and re-annotate R^{ref} for the new (τ^c, τ^f) pairs using the same VLM annotation pipeline, then repeat I2.

Inference with GT Prefill. For Table 4(b), we evaluate the same final RL checkpoint on NavTest. Standard inference generates the complete C2R response. GT prefill supplies the human-GT trajectory as τ^c in the response prefix, after which the model generates R^{ref} and τ^f. We evaluate both τ^c and τ^f with PDMS.

J Extended Visual Attribution Gallery

Figure 11 expands the sample visualizations in Figure 6 to eight matched driving scenes. The goal of this appendix is qualitative: it makes the field-level image-reliance evidence inspectable at the scene level, while the aggregate measurements are reported in Figure 5. Each matched scene compares Prefix Reasoning with C2R under the same front-view observation and high-level maneuver. The Prefix Reasoning row overlays its predicted trajectory and the image-reliance map for its explanation field, whereas the C2R row overlays the coarse trajectory, the refined trajectory, and the image-reliance map for the refinement explanation.

Across left turns, right turns, and straight driving, the gallery shows the same pattern as the main analysis. Prefix Reasoning often produces a plausible explanation field, but the visible reliance map can be broad or displaced from the part of the scene that determines the final trajectory. C2R instead evaluates a concrete coarse proposal before predicting the final trajectory, so its highlighted regions more often align with trajectory-relevant road structure, nearby vehicles, traffic signals, or turn geometry. The refined trajectory is therefore best read as a targeted correction conditioned on both the coarse draft and the refinement explanation, rather than as an independent second prediction or an external post-processing step.

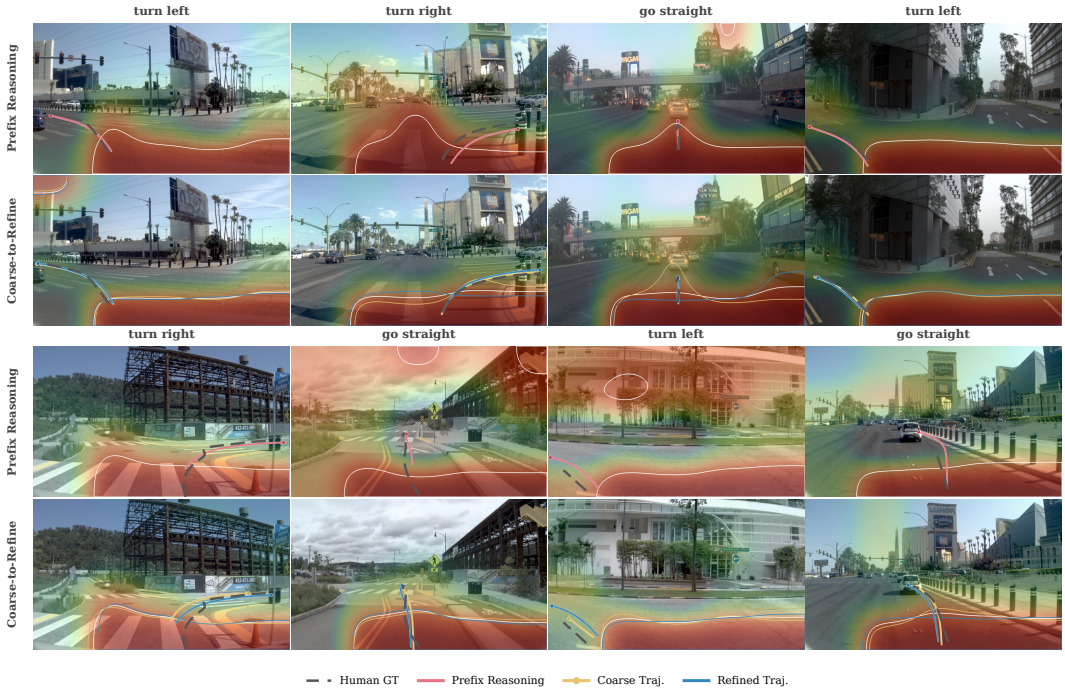


Figure 11: Extended visual attribution gallery. The figure shows eight matched scenes covering left turns, right turns, and straight driving. For each scene, the Prefix Reasoning row shows its generated trajectory and the image-reliance map of the explanation field; the C2R row shows the coarse trajectory, refined trajectory, and image-reliance map of the refinement explanation. Dashed gray denotes human GT, pink denotes Prefix Reasoning, yellow denotes the C2R coarse trajectory, and blue denotes the C2R refined trajectory. These samples qualitatively support the aggregate finding in Figure 5: C2R's refinement explanation is more visually grounded, and its final trajectory uses the coarse draft and explanation as trajectory-revision context.

K Compatibility with Scorer-Based Methods

C2R can be paired with scorer-based selection without changing the trained policy. For each sampled C2R response, we parse the final refined trajectory τ^f and expose only this trajectory to the external scorer. The coarse trajectory and refinement reasoning remain internal to the C2R response format, so the scorer ranks candidate trajectories rather than reasoning text.

Most ablations in this paper use NAVSIM v2 because it is the stricter protocol for internal design comparisons. For scorer-based selection and Best-of-N sampling, however, we follow the NAVSIM v1 PDMS reporting protocol used by the relevant public candidate-selection comparisons. This keeps the candidate counts and PDMS values directly comparable to published VLA planning tables, while the standard NAVSIM v1 sub-metrics are included for completeness.

Table 17 reports the corresponding NAVSIM v1 comparison. The C2R-only row uses the default single-response inference protocol. The scorer rows use candidate selection, with the candidate count shown explicitly. Under this setting, C2R paired with a learned scorer model reaches the same PDMS as the 64-candidate scorer-model baseline while evaluating fewer candidates per scene. This NAVSIM v1 compatibility table is complementary to the main-text NavHard table: there, scorer-based methods are reported in gray and C2R paired with DrivoR is evaluated under the NavHard two-stage EPDMS protocol. In both cases, the interface is unchanged: C2R exposes only the parsed final trajectory τ^f to the scorer, so scorer integration ranks trajectories rather than C2R's internal coarse trajectory or refinement reasoning.

Table 17: Compatibility with scorer-based selection on NAVSIM v1 NavTest. The columns follow the same sub-metric order as the main NAVSIM v1 table. Cand. is the number of candidate trajectories scored per scene.

Method	Cand.	NC↑	DAC↑	EP↑	TTC↑	C↑	PDMS↑
C2R	1	98.8	97.9	89.0	95.4	100.0	92.1
DrivoR [24]	64	99.0	98.9	90.0	96.7	100.0	93.7
C2R + DrivoR	16	99.1	98.9	90.5	96.3	100.0	93.7

L Best-of-N Sampling

Best-of-N is a direct scorer-selection protocol for autoregressive text-waypoint generation. Given the same scene input, we sample N stochastic complete responses, parse the final refined trajectory τ^f from each response, score the parsed trajectories with the NAVSIM v1 PDMS scorer, and return the highest-scored trajectory. Following the NAVSIM v1 Best-of-N setting used in prior VLA planning comparisons, all rows in Table 18 use $N = 6$ candidates per scene. Each candidate is still produced in one autoregressive C2R pass; Best-of-N adds selection over sampled candidates, not another refinement loop.

Best-of-N evaluates the quality of sampled candidates through oracle selection. It uses the benchmark scorer after candidates have been generated, analogous to how pass@k in language-model evaluation measures whether a correct solution appears among multiple samples. Therefore, Best-of-N reveals an upper-bound property of the sampled candidate set. By contrast, a learned scorer model can be used at inference time to rank candidate trajectories without access to the benchmark oracle.

Table 18 compares Best-of-N results under the NAVSIM v1 protocol, using the same sub-metrics as the main NAVSIM v1 table. Although the main ablations are generally reported on NAVSIM v2, we use NAVSIM v1 here because published Best-of-N rows for VLA planning are most commonly available under the NAVSIM v1 PDMS protocol. The baseline rows report Best-of-N results under the settings reported by the cited methods. Relative to the single-response C2R result in Table 17, oracle selection increases PDMS from 92.1 to 95.3.

Table 18: Best-of-N comparison on NAVSIM v1 NavTest. The N column is the number of sampled candidates per scene. The metric columns follow the same sub-metric order as the main NAVSIM v1 table.

Method	Selection	N	Base Model	Sensors	NC↑	DAC↑	EP↑	TTC↑	C↑	PDMS↑
DriveVLA-W0 [27]	BoN	6	Emu3-8B	1×C	99.3	97.4	88.3	97.0	99.9	93.0
AutoVLA [65]	BoN	6	Qwen2.5-VL-3B	3×C	99.1	98.8	87.9	97.2	100.0	92.1
AdaThinkDrive [36]	BoN	6	InternVL3-8B	1×C	99.1	98.8	87.9	95.2	100.0	93.0
C2R	BoN	6	Qwen3-VL-4B	1×C	99.2	98.8	93.8	96.9	100.0	95.3

M Computational Overhead

C2R introduces additional output tokens, but it keeps planning in a single autoregressive call. Each rollout emits the coarse trajectory, the refinement reasoning, and the refined trajectory together, after which the planner consumes only the parsed refined trajectory τ^f. The extra cost over Prefix Reasoning is therefore mainly longer response decoding, not a second image pass, an external planner, or multi-turn feedback.

Profiling Setup. Table 19 gives the settings for the Prefix–C2R comparison and C2R serving optimization. The Prefix checkpoint is Prefix-Guided. Each policy uses its corresponding training-format profiling subset. We report arithmetic means over measured requests. Client timing starts after image preprocessing and request-body creation. Prefill and decode use server measurements; the IO column is the remaining client time. Field timings use streamed chunk arrival boundaries. The C2R reasoning field includes object and behavior fields, explanation, field names, and the JSON syntax between the two trajectory blocks. The reported trajectory and reasoning fields exclude the response prefix and suffix, so their sum need not equal server decode time.

Table 19: Settings for Prefix–C2R profiling and C2R serving acceleration. Both BF16 profiles use the same server and generation settings. The optimized run retains the original C2R prompt and image/text order.

Setting	Value
Hardware / server	One NVIDIA H20; one vLLM 0.26.0 instance; tensor parallelism 1
Execution	Batch size 1; strictly serial requests
Sampling	3 warmup requests, then 100 measured requests per profile
Generation	Greedy; maximum 768 output tokens
BF16 serving	O3 compilation, FlashAttention 3, asynchronous scheduling; prefix cache disabled
Speculative decoding	GPU n-gram, 8 draft tokens, lookup range 2–5; enabled in both BF16 and optimized runs
Optimized serving	Static FP8 W8A8, explicit compiler fusions, full CUDA graphs, prefix cache enabled
FP8 scope	Language-model linear layers; visual tower and output head remain BF16
Calibration	32 existing C2R prompts for static activation scales
Timing	Client request wall time after preprocessing; server prefill/decode; streaming field boundaries
Summary	Arithmetic mean over measured requests

Prefix–C2R Inference Profiling. Table 21(b) reports output length and timing for both policies. The latency increment is concentrated in decoding; refinement reasoning is C2R's largest individual decoding field.

Serving Acceleration. Table 21(c) compares BF16 serving with the FP8 optimized configuration in Table 19. The optimized configuration accelerates the same response formulation.

We report latency under serial inputs, measured as wall-clock seconds per evaluated scene. Table 21 follows the standard text-waypoint VLA latency convention and compares autoregressive text-waypoint methods, followed by Prefix–C2R inference profiling and serving acceleration. C2R takes 1.98s per scene in this setting, while AutoVLA's text-waypoint dual-system path takes 9.31s. The comparison covers single-rollout inference with text waypoints. C2R uses one camera view and a single autoregressive response. Latency depends on the hardware, serving backend, batching, and decoding configuration of each method.

Best-of-N and scorer-based selection add candidate generation on top of this single-rollout path. For a fixed scene, the candidates share the same image, navigation command, ego state, and prompt prefix. Modern inference backends, such as vLLM-style serving systems, are designed to exploit exactly this structure through prefix caching and KV-cache reuse: the shared visual input and prompt prefix can be prefetched once, while multiple stochastic continuations branch from the cached state. Thus multi-candidate C2R does not require naively repeating the full image-and-prompt computation for every candidate. Its marginal cost is dominated by response-suffix decoding and trajectory scoring, which makes candidate selection much cheaper than independent full forward passes. In Appendix K, C2R with a learned scorer uses 16 scored candidates, compared with 64 candidates for the DrivoR scorer baseline. The reported multi-candidate settings are therefore candidate multipliers on the same planning interface: standard C2R uses one response per scene, Best-of-N samples six responses, and scorer integration uses 16 C2R candidates. With caching-aware serving, these multipliers are strongly amortized. Table 20 reports the p50 measurements for multi-candidate generation with shared scene prefixes. With the shared scene prefix cached, additional stochastic continuations mainly pay response-suffix decoding and scoring.

Table 20: Multi-candidate generation with shared scene prefixes and cache reuse. Total latency is p50; overhead and throughput are relative to the single-rollout measurement in this experiment. This is a separate measurement from the serial-request means in Table 21.

Rollouts	Total (s)	Overhead	Throughput	Seconds / rollout
6	2.21	+11.6%	5.4×	0.37
16	2.37	+19.7%	13.4×	0.15

Table 21: Serial inference latency at batch size 1. (a) Autoregressive text-waypoint methods under their respective inference configurations. (b) Prefix-Guided–C2R inference profiling with the same Qwen3-VL-4B backbone. (c) C2R serving acceleration with settings in Table 19. Times are seconds per request; (b) and (c) use the timing definitions above; token counts are per-response averages.

(a) Autoregressive text-waypoint comparison

Method	Backbone	Cameras	Latency (s) ↓	PDMS ↑
OneVL AR, answer only [35]	Qwen3-VL-4B	1	4.49	87.5
OneVL AR, CoT + answer [35]	Qwen3-VL-4B	1	6.58	88.3
Curious-VLA [6]	Qwen2.5-VL-3B	1	1.57	90.3
AutoVLA [65]	Qwen2.5-VL-3B	3	9.31	89.1
Prefix-Guided	Qwen3-VL-4B	1	1.50	87.57
C2R	Qwen3-VL-4B	1	1.98	92.1

(b) Inference profiling

Formulation	Tokens	Coarse	Reasoning	Final	Prefill	Decode	IO	Total
Prefix-Guided	365.56	—	0.887	0.477	0.094	1.384	0.022	1.500
C2R	576.30	0.462	0.978	0.369	0.091	1.868	0.023	1.982

(c) Serving acceleration

Configuration	Tokens	Prefill	Decode	IO	Total	Speedup
BF16	576.30	0.091	1.868	0.023	1.982	1.000×
FP8 optimized	577.52	0.058	1.465	0.016	1.539	1.288×

N Prefix vs. Coarse-to-Refine: Additional Analysis

Figure 12 provides a difficulty-conditioned diagnostic of the final trajectories produced by Prefix Reasoning and C2R. This analysis uses a randomly sampled validation subset. We sort validation examples by the FDE of the Prefix Reasoning trajectory to human GT and split them into four equal-size bins. The left panel reports, for each bin, the mean FDE to human GT and the mean PDMS of the final Prefix trajectory and the final C2R refined trajectory. The right panel expands the hardest Prefix bin into per-example FDE outcomes.

The comparison highlights a difference between direct prediction and trajectory revision. Prefix Reasoning can be close to the human endpoint in the easiest bins, but its FDE and PDMS degrade as the same model's Prefix-FDE difficulty increases. C2R is less sensitive to this Prefix-defined difficulty split: its PDMS remains nearly flat across bins (98.1–98.4), and its mean FDE stays in a narrower range (1.78–2.73 m). In the hardest Prefix bin, C2R has lower mean FDE than Prefix Reasoning (2.73 m vs. 3.65 m) and is better or tied on 82% of examples. This pattern is consistent with the C2R formulation: the final trajectory is generated after the model has committed to a coarse proposal and a refinement explanation, making the prediction a concrete revision rather than a direct trajectory emitted after prefix reasoning alone.

FDE measures endpoint proximity to a human trajectory and is useful for describing geometric mismatch, but it is not the only or necessarily the best criterion for driving trajectory quality. NAVSIM PDMS evaluates the planning trajectory through the official score terms used in the benchmark, so the stable C2R PDMS across Prefix-defined difficulty bins is the stronger evidence that C2R preserves planning quality even when exact endpoint matching varies.

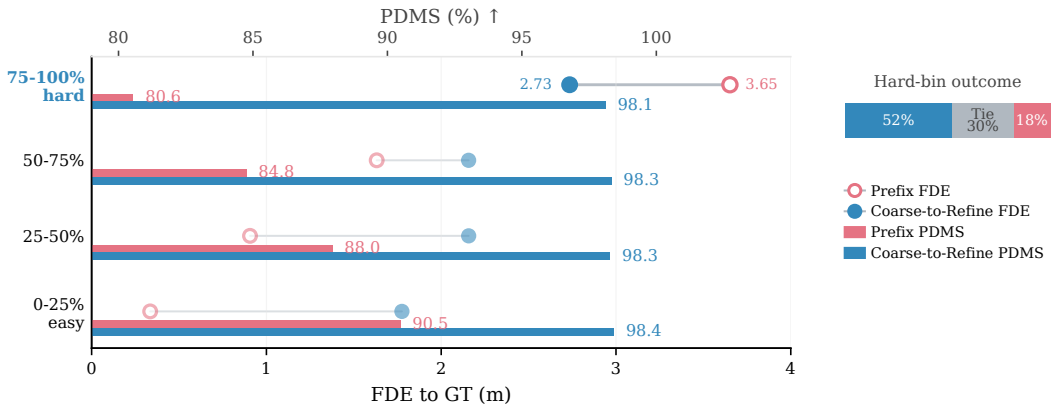


Figure 12: Difficulty-conditioned final-output comparison between Prefix Reasoning and C2R on a randomly sampled validation subset. Examples are sorted by Prefix Reasoning FDE to human GT and split into four equal-size bins from easiest (0–25%) to hardest (75–100%). Circles show mean FDE to human GT; bars show mean PDMS for the same final trajectories. C2R keeps nearly constant PDMS across the Prefix-defined difficulty bins (98.1–98.4) and a comparatively stable FDE range (1.78–2.73 m), while Prefix Reasoning drops from 90.5 to 80.6 PDMS and its FDE increases from 0.34 m to 3.65 m. The hardest-bin outcome strip reports per-example FDE wins, ties, and losses for C2R relative to Prefix Reasoning. FDE measures endpoint proximity to human GT, while PDMS evaluates planning quality.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: The contributions, including C2R framework, the two-stage training pipeline and SOTA results are clearly stated in the Abstract and Introduction. The experimental results in Section 4 and 5 directly support these claims.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: We discuss the limitations in Section 6, specifically noting that the current data construction relies on rule-based scoring and highlighting potential future work on learnable scorers.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

Answer: [N/A]

Justification: This is an empirical study focused on model architecture and training strategies, which does not contain formal mathematical theorems or proofs.

Guidelines:

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

4. **Experimental result reproducibility**

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

Answer: [Yes]

Justification: We detail the model architecture in Section 3 and the experimental setup in Section 4. And the comprehensive hyperparameter settings and training schedules are provided in Appendix D. The data construction pipeline (GSRS) is extensively detailed in Appendix B.

Guidelines:

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

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

5. **Open access to data and code**

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

   Answer: [Yes]

   Justification: We utilize publicly available benchmarks (NAVSIM and NavHard). The code and the data construction pipeline will be released following the review phase.

   Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: Training details, including the use of SFT and RLVR, are described in Section 3.2 and 3.3. Evaluation metrics are defined in Section 4.1 and Appendix C. Further technical configurations are detailed in Appendix D.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [No]

   Justification: Due to the extreme computational cost of training 4B-class VLA models and performing extensive closed-loop evaluations, we report results from single representative runs, consistent with standard practice in the field. However, the performance improvements

are validated through comprehensive experiments across multiple benchmarks (NAVSIM v1, v2, and NavHard).

Guidelines:

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

8. **Experiments compute resources**

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

   Answer: [Yes]

   Justification: Information regarding GPU types, total training hours, and software environment is provided in Appendix D.

   Guidelines:

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

9. **Code of ethics**

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

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Ethics Guidelines and confirm that our research, which focuses on enhancing autonomous driving safety using public datasets, complies with all standards.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.

- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

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

    Answer: [Yes]

    Justification: C2R improves trajectory planning quality, contributing to safer autonomous driving. However, all experiments are conducted in simulation (the NAVSIM closed-loop benchmark); real-world deployment requires additional safety validation and regulatory compliance beyond the scope of this work.

    Guidelines:

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

11. **Safeguards**

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

    Answer: [N/A]

    Justification: This research focuses on trajectory planning within a simulated environment using public datasets. The model is not deployed on real-world vehicles, and the study does not involve the release of sensitive personal or biological data.

    Guidelines:

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

12. **Licenses for existing assets**

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

Answer: [Yes]

Justification: All used datasets (NAVSIM, nuPlan) and base models (Qwen-VL) are properly cited. License details for these assets are provided in Appendix D.

Guidelines:

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

13. **New assets**

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

Answer: [Yes]

Justification: We introduce the GSRS data synthesis pipeline in Section 3.2, which is documented with details in Appendix B. This asset will be made public alongside our code release.

Guidelines:

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

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

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

Answer: [N/A]

Justification: This research does not involve human subjects, crowdsourcing, or personal data collection.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

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

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

    Answer: [N/A]

    Justification: This work does not involve any human subjects.

    Guidelines:

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

16. **Declaration of LLM usage**

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

    Answer: [Yes]

    Justification: Qwen3-VL-235B is used in the GSRS data synthesis pipeline for annotating chain-of-thought reasoning (Appendix A). LLMs were also used for writing assistance.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.