SOMETIMES YOU GOTTA RUN BEFORE YOU CAN WALK: RUN-THEN-WALK SCHEDULING STRATEGY FOR VLM AUTONOMOUS DRIVING

Anonymous authors
Paper under double-blind review

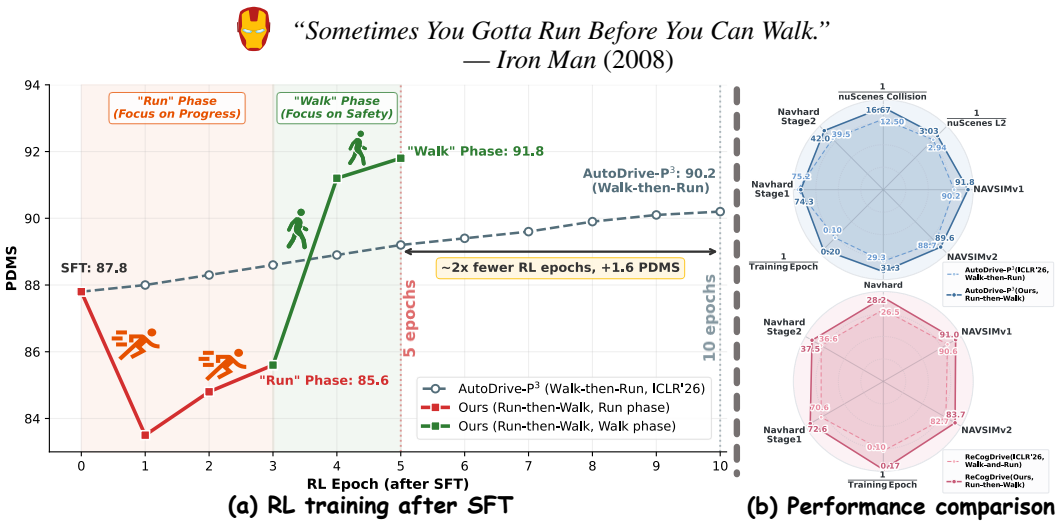


Figure 1: **RL training and performance comparison.** (a) Our *Run-then-Walk* strategy reaches a higher PDMS of 91.8 in only 5 RL epochs with AutoDrive-P^3 planner (Ye et al., 2026), outperforming 90.2 PDMS achieved by the 10-epoch baseline while substantially accelerating convergence. (b) Radar-chart comparisons of various methods on different benchmarks. With *Run-then-Walk*, the evaluated planner families improve their aggregate scores on the reported datasets while training converges 40–50% faster.

ABSTRACT

Recent VLM-based autonomous driving planners adopt GRPO-style reinforcement learning to optimize driving performance. However, existing GRPO recipes either optimize driving efficiency, risking progress-seeking but unsafe behavior, or enforce early safety constraints, leading to overly conservative behavior; both require lengthy training. To solve these problems, we first reveal two distinct RL regimes: a progress regime (Run-GRPO) that aggressively explores high progress, and a safety regime (Walk-GRPO) that restores safety under stable progress. Based on this finding, we propose *Run-then-Walk*, a simple yet effective two-stage reward scheduling strategy for GRPO, achieving both better performance and faster convergence. Unlike one-stage RL, which may focus on progress, safety, or a mixture of both within a single training phase, this schedule explicitly separates progress discovery from safety repair. In the *Run* phase, we focus on progress, allowing the policy to escape the conservative bias and discover high-progress modes. In the subsequent *Walk* phase, we introduce endpoint and safety strategy to repair unsafe behaviors from the Run phase. This reversed schedule overcomes the conservatism of Walk-first methods and the unsafe progress-seeking of joint optimization. We validate it with various VLM-based planners on multiple benchmarks: NAVSIMv1, NAVSIMv2, Navhard, and nuScenes. Extensive experiments demonstrate improved driving performance while requiring 40–50% fewer RL training epochs than the baselines.

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

1 INTRODUCTION

End-to-end autonomous driving has evolved from compact architectures (Hu et al., 2023; 2022; Jiang et al., 2023; Liao et al., 2025; Li et al., 2025b; 2024a) to Vision-Language Model (VLM) based systems (Tian et al., 2024; Hwang et al., 2024; Xu et al., 2024; Xing et al., 2025a), as they leverage large-scale pre-training with world knowledge. While VLMs achieve strong results via SFT, recent VLM planners adopt GRPO-style RL (Shao et al., 2024; Guo et al., 2025) to achieve better performance, especially in closed-loop driving quality (Dauner et al., 2024; Cao et al., 2025). Composite closed-loop metrics entangle progress and safety, and balancing these conflicting objectives remains the central challenge of RL for autonomous driving.

To address this issue, more structured training strategies have been explored (Zhou et al., 2025b; Jiang et al., 2025; Li et al., 2025c; Ye et al., 2026). However, as illustrated in Fig. 2, existing RL strategies still suffer from three major limitations. **1) Walk-and-Run: High collision.** ReCogDrive (Li et al., 2025c) uses Walk-and-Run style objectives, but its training remains biased toward progress. The resulting policy achieves progress with a high collision rate and other safety violations. **2) Walk-then-Run: Over-conservatism.** AutoDrive-P^3 (Ye et al., 2026) starts from strict safety constraints and only gradually relaxes them, making the model overly conservative and hesitant to move forward (Fig. 2(b)). **3) Lengthy training.** Both baselines require 10 RL epochs, which is extremely time-consuming and consumes substantial GPU resources. This naturally raises a question: *can we find a faster and more effective RL training paradigm for VLM-based driving?*

To answer this question, we first conduct a controlled toy experiment using AutoDrive-P^3 on NAVSIM as a concrete training instance. Starting from the same SFT checkpoint, we compare direct Run-GRPO (progress-oriented) and direct Walk-GRPO (safety-constrained) as the first RL objective. As shown in Fig. 3, direct Run-GRPO first rapidly optimizes ego progress (EP), and only after EP saturates does the policy begin to repair safety metrics; however, without explicit safety constraints, improvement within this aggressive regime is limited. Direct Walk-GRPO exhibits the opposite pattern: the policy explores only within the safe region around the SFT solution, preserving safety at the cost of slow progress improvement. ***Our key finding is that RL exhibits two distinct regimes: a progress regime (Run-GRPO) that aggressively explores high progress and a safety regime (Walk-GRPO) that keeps safe under low progress.***

Motivated by this diagnosis, we propose *Run-then-Walk*, a simple yet effective two-stage scheduling strategy for GRPO. Unlike one-stage RL strategy, which may focus on progress, safety, or a mixture of both within a single training phase, our strategy explicitly separates progress in an ordered manner: the *Run* phase first encourages progress exploration under relaxed constraints, and the *Walk* phase then introduces endpoint and safety objectives to repair unsafe behaviors. As illustrated in Fig. 2(c), this schedule differs both from one-stage RL and from generic easy-to-hard curricula, while addressing the unsafe behavior of *Walk-and-Run* and the over-conservatism of *Walk-then-Run*.

As shown in Fig. 1, this algorithm yields both faster convergence and higher final performance. Across different VLM-based planners and multiple benchmarks, it improves driving performance while requiring only 5 epochs for AutoDrive-P^3 and 6 epochs for ReCogDrive, fewer than the 10 epochs used by the original RL policies. The final policies retain high EP and restore collision, drivable-area, and time-to-collision safety metrics. We hope that this progress-first, safety-refinement paradigm, simple and effective, can extend beyond autonomous driving to broader VLA decision-making problems. The main contributions of this paper are summarized as follows:

- We reveal that the RL optimization direction for VLM-based driving critically affects the policy, and summarize it into two main regimes: a progress regime (Run-GRPO) that aggressively explores high progress at the cost of safety, and a safety regime (Walk-GRPO) that preserves safety but remains conservative with slow progress improvement.
- We introduce *Run-then-Walk*, a simple and effective two-stage RL training strategy that first expands the progress distribution under relaxed constraints and then repairs safety with endpoint and safety-related objectives, which differs both from one-stage mixed-objective RL and from generic easy-to-hard curricula.
- We validate the same algorithm on VLM-based autoregressive and diffusion planners across NAVSIMv1, NAVSIMv2, Navhard, and nuScenes. The results show higher driving performance, and 40–50% fewer RL training steps.

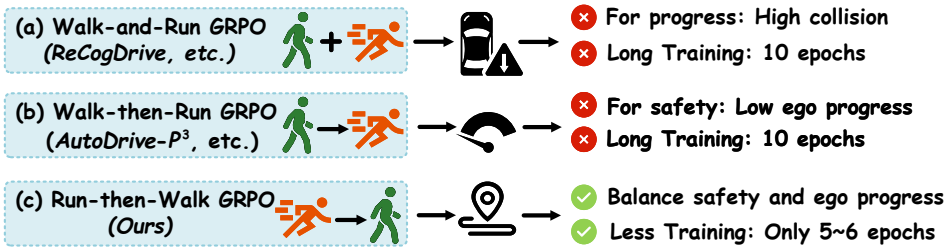


Figure 2: **Comparison of GRPO training paradigms.** (a) The *Walk-and-Run* paradigm, like ReCogDrive (Li et al., 2025c), remains biased toward Run, resulting in high collision. (b) The *Walk-then-Run* paradigm, seen in AutoDrive-P^3 (Ye et al., 2026), prioritizes safety at the cost of low ego progress. (c) Our proposed *Run-then-Walk* paradigm first explores freely and then refines for safety, achieving a balance between safety and progress while saving 40–50% of the training steps.

2 Related Work

VLM-based end-to-end driving. End-to-end driving has evolved from modular pipelines such as UniAD (Hu et al., 2023), VAD (Jiang et al., 2023), Law (Li et al., 2024a), Wote (Li et al., 2025b), and DiffusionDrive (Liao et al., 2025) to VLM-based systems including DriveVLM (Tian et al., 2024), EMMA (Hwang et al., 2024), VLM-AD (Xu et al., 2024), and OpenEMMA (Xing et al., 2025a). These VLM planners either use VLM features to support a downstream trajectory generator, as in ReCogDrive (Li et al., 2025c), or directly emit trajectories as language tokens, like OpenDriveVLA (Zhou et al., 2025a), AutoVLA (Zhou et al., 2025b), and AutoDrive-P^3 (Ye et al., 2026). To validate the robustness of our method, we evaluate it on two representative VLM planners: the autoregressive AutoDrive-P^3 and the diffusion-based ReCogDrive.

Group Relative Policy Optimization. Group Relative Policy Optimization (GRPO) (Shao et al., 2024; Guo et al., 2025; Shao et al., 2025), originally proposed by DeepSeek, has emerged as an effective reinforcement learning algorithm for enhancing the reasoning capabilities of large language models. Its success has since been extended to vision-language models, with works such as Vision-R1 (Huang et al., 2025) demonstrating that GRPO can substantially improve VLM reasoning. In the autonomous driving domain, several recent efforts have adopted GRPO to fine-tune driving VLMs with closed-loop metrics as rewards. However, existing approaches lie between two extremes: mixed Walk-and-Run schedules that remain biased toward progress (Li et al., 2025c), which may preserve progress-seeking but unsafe behaviors, and safety-first schedules that relax constraints only gradually (Ye et al., 2026), which may suppress progress and require a long training horizon. These limitations indicate that how different RL training regimes shape the progress-safety trade-off in VLA-based planners remains poorly understood. Systematically characterizing these regimes is therefore essential for designing RL strategies with better performance and faster convergence.

3 Toy Experiment: Run-GRPO vs. Walk-GRPO from SFT

We conduct a toy experiment to understand how the first RL objective shapes policy learning after SFT. Using the AutoDrive-P^3 planner in NAVSIMv1 benchmark as an illustrative example, we start from its SFT checkpoint, which already exhibits low-ADE yet relatively low-progress behavior, and compare two direct RL fine-tuning strategies. In direct Run-GRPO, we use a PDMS metric dominated by ego progress as the planning reward. Although PDMS is a composite reward, EP accounts for $5/12$ of its total weight, i.e., nearly half, and EP is also a relatively easier objective to optimize, so the policy tends to learn high-progress behaviors. In direct Walk-GRPO, we instead use a constrained planning reward consisting of the endpoint reward together with the safety terms DAC and NC from PDMS. All other settings are kept the same, and Fig. 3 reports the resulting EP, DAC, NC, PDMS, and ADE trends during training.

Fig. 3 shows that direct Run-GRPO and direct Walk-GRPO fail in opposite ways. With direct Run-GRPO, the policy quickly discovers that ego progress (EP) is the easiest part of PDMS to optimize from the SFT initialization. It therefore learns aggressive forward-driving behaviors first, which raises EP but also increases collisions and drivable-area violations. As a consequence, the overall PDMS drops below the SFT baseline even though progress improves. Only after the policy

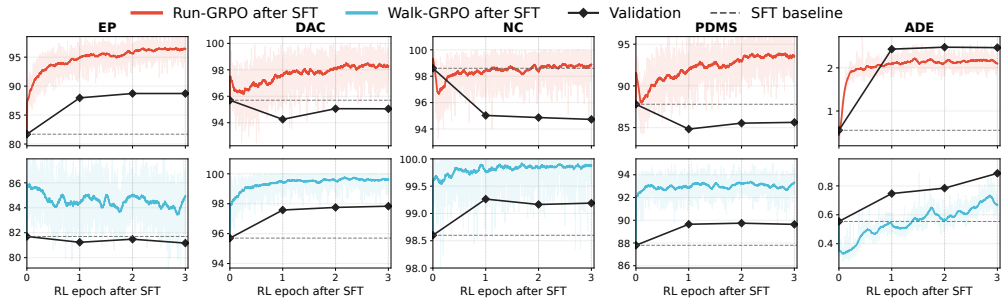


Figure 3: **Toy experiment from the same SFT checkpoint.** Starting from the same SFT model, we compare direct Run GRPO and direct constrained Walk GRPO for three epochs. Direct Run quickly discovers aggressive high-progress behaviors, but then has to improve safety within this overly aggressive regime, causing PDMS to fall well below the SFT baseline. Direct Walk exhibits the opposite pattern: it explores only within a conservative safety regime, so PDMS improves slowly.

has already entered this aggressive regime does optimization begin to repair safety. At that point, however, the model can only search for trajectories that are as safe as possible under aggressive behavior, making it difficult to recover a truly balanced policy.

Direct Walk-GRPO exhibits the opposite pattern. Because its planning reward is dominated from the beginning by endpoint consistency and the safety terms DAC and NC, the policy is encouraged to explore only within a constrained safe region around the SFT solution. This preserves and improves safety-related behavior, but it also suppresses forward exploration and keeps the policy conservative. As a result, PDMS improves only slowly, since the model learns to drive safely under constraints rather than to discover stronger progress modes.

This contrast directly motivates our Run-then-Walk strategy. A good first-stage objective after SFT should be able to break the conservative bias and uncover progress-oriented behaviors, while a good second-stage objective should repair the safety deficits introduced by that exploration. Run-then-Walk follows exactly this logic: it first uses Run to expand the progress distribution, and then uses Walk to refine the policy into safe and stable closed-loop driving.

4 Method

4.1 Run-then-Walk Planning Strategy Design

Let π_θ be a VLM driving policy, for a driving query q, a complete policy sample x_i can be an autoregressive response or a continuous sample from a diffusion trajectory head; a decoder maps either form to a trajectory τ_i. Run-then-Walk preserves the planner's architecture and native auxiliary objectives and changes only temporal schedule and the planning reward on the decoded trajectory distribution rather than on a particular output representation.

In the Run stage, we adopt an aggressive planning reward that prioritizes ego progress,

$$R^{\text{run}}_{\text{plan}}(\tau_i) = PDMS(\tau_i), \tag{1}$$

where $PDMS(\tau_i)$ denotes an aggressive reward function. This stage intentionally relaxes safety repair and encourages the policy to move beyond the conservative SFT solution by discovering higher-progress trajectory modes.

In the Walk stage, we keep the other reward terms unchanged and replace the aggressive planning reward with a new planning objective composed of endpoint and safety-related reward terms. We design the endpoint distance d_E as the L_1 distance between the final predicted point p_T and the expert endpoint g_T:

$$d_E = \|p_T - g_T\|_1. \tag{2}$$

We introduce a step size hyperparameter $\Delta > 0$ and define the endpoint reward as

$$k(d_E) = \min(K,\ \max(0, \lfloor (d_E - 2\Delta)/\Delta \rfloor + 1))\,, \qquad R_{\text{end}}(d_E) = 1 - \eta\, k(d_E), \tag{3}$$

where $\eta \in (0, 1)$ is the reward decrement and $K = \lfloor 1/\eta \rfloor$ is the maximum number of decay steps. This yields a piecewise endpoint reward that decreases by a fixed amount every Δ units of endpoint

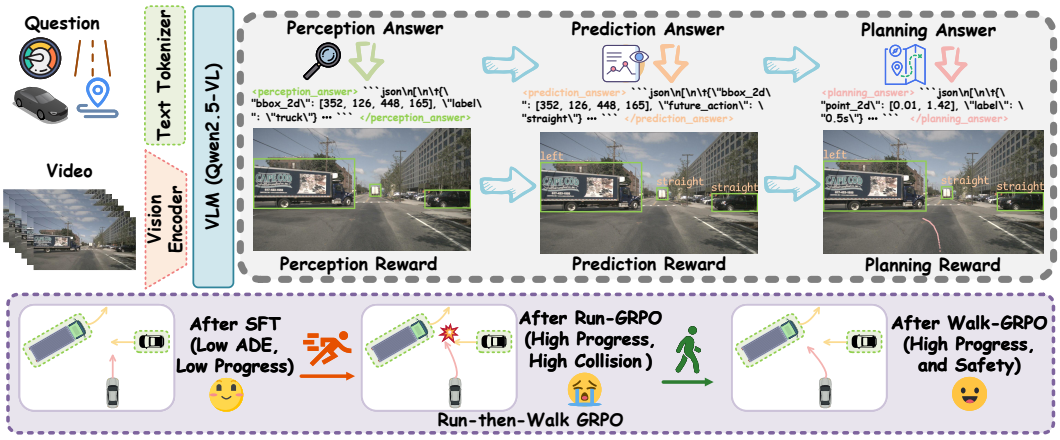


Figure 4: **Illustrative Run-then-Walk scheduling strategy for AutoDrive-P^3 planner.** It performs RL in two stages: a *Run* phase that discovers high-progress actions under relaxed constraints, followed by a *Walk* phase that adds endpoint and safety rewards to produce a safe, high-progress policy.

error and eventually becomes zero. A smaller Δ enforces a tighter endpoint constraint, making the policy more conservative, while a larger Δ yields a looser Walk stage, making it more aggressive. Additionally, a larger η causes the endpoint reward to decay more aggressively. With the endpoint reward, the Walk planning reward is

$$R_{\text{plan}}^{\text{walk}}(\tau_i, \tau_i^\star) = \mathbb{I}_{\text{safe}}(\tau_i)\left(R_{\text{safe}}(\tau_i) + R_{\text{end}}(d_E)\right), \tag{4}$$

where $\mathbb{I}_{\text{safe}}(\tau_i) = 1$ only if the rollout satisfies all safety constraints, and is 0 otherwise. Here $\tau_i^\star$ denotes the expert trajectory, and $R_{\text{safe}}(\cdot)$ denotes a safety-related reward term. Therefore, once a rollout becomes unsafe, its planning reward is set to zero; otherwise, the Walk reward combines endpoint consistency with safety-related terms. In our implementation, we instantiate $R_{\text{safe}}(\tau_i)$ as $\text{NC}(\tau_i) + \text{DAC}(\tau_i)$, and define $\mathbb{I}_{\text{safe}}(\tau_i) = 1$ only if $\text{NC}(\tau_i) > 0$, and $\text{DAC}(\tau_i) > 0$.

Because both stages consume only a decoded trajectory and closed-loop feedback, the construction is planner-agnostic. For an autoregressive planner, the reward is assigned to the sampled token sequence that contains the trajectory; for a VLM-conditioned diffusion planner, it is assigned to the sampled continuous trajectory. In both cases, Run expands high-progress modes and Walk applies the same endpoint term and safety gate while preserving the planner's native auxiliary supervision.

4.2 Run-then-Walk Strategy

For each driving query q, we sample a group of G complete planner outputs $\{x_i\}_{i=1}^G$ from the current policy and compute the stage-wise reward

$$R_i^{(s)} = R_{\text{aux}}(x_i) + \lambda_{\text{plan}} R_{\text{plan}}^{(s)}(\tau_i), \tag{5}$$

where $s \in \{\text{run}, \text{walk}\}$ and R_{aux} collects any architecture-native rewards that remain fixed across stages. These can include format, perception, and prediction rewards for a structured autoregressive planner or the native auxiliary terms of a diffusion planner.

The sampled rewards are normalized into group-relative advantages:

$$A_i^{(s)} = \frac{R_i^{(s)} - \bar{R}^{(s)}}{\sigma_R^{(s)} + \epsilon_A}, \qquad \bar{R}^{(s)} = \frac{1}{G}\sum_{j=1}^{G} R_j^{(s)}, \qquad \sigma_R^{(s)} = \sqrt{\frac{1}{G}\sum_{j=1}^{G}\left(R_j^{(s)} - \bar{R}^{(s)}\right)^2}. \tag{6}$$

We set $r_i = \pi_\theta(x_i|q)/\pi_{\theta_{\text{old}}}(x_i|q)$, then optimize the policy with the GRPO objective:

$$\mathcal{J}^{(s)}(\theta) = \mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G}\min\left(r_i A_i^{(s)},\ \text{clip}(r_i, 1-\epsilon, 1+\epsilon)A_i^{(s)}\right)\right] - \beta_s\,\mathbb{E}\left[D_{\text{KL}}\left(\pi_\theta(\cdot|q)\,\|\,\pi_{\text{ref}}^{(s)}(\cdot|q)\right)\right]. \tag{7}$$

Table 1: **Performance comparison on NAVSIMv1 benchmark.**

Method	Image	Lidar	NC↑	DAC↑	EP↑	TTC↑	Comf↑	PDMS↑
Human	✗	✗	100.0	100.0	87.5	100.0	99.9	94.8
Constant Velocity	✗	✗	69.9	58.8	49.3	49.3	100.0	21.6
Ego Status MLP	✗	✗	93.0	77.3	62.8	83.6	100.0	65.6
VADv2 (Weng et al., 2024)	✔	✗	97.9	91.7	77.6	92.9	100.0	83.0
UniAD (Hu et al., 2023)	✔	✗	97.8	91.9	78.8	92.9	100.0	83.4
TransFuser (Prakash et al., 2021)	✔	✔	97.7	92.8	79.2	92.8	100.0	84.0
PARA-Drive (Weng et al., 2024)	✔	✗	97.9	92.4	79.3	93.0	99.8	84.0
Hydra-MDP (Li et al., 2024b)	✔	✔	98.3	96.0	78.7	94.6	100.0	86.5
DiffusionDrive (Liao et al., 2025)	✔	✔	98.2	96.2	82.2	94.7	100.0	88.1
WoTE (Li et al., 2025b)	✔	✔	98.5	96.8	81.9	94.9	99.9	88.3
DriveVLA-W0 (Li et al., 2025a)	✔	✗	98.7	99.1	83.3	95.3	99.3	90.2
DriveFuture (Hong et al., 2026)	✔	✗	99.1	97.2	84.5	96.0	100.0	90.7
DriveWorld-VLA (Liu et al., 2026)	✔	✗	99.1	98.2	85.9	96.1	100.0	91.3
ReCogDrive (Li et al., 2025c)	✔	✗	98.1	97.7	86.5	94.9	100.0	90.6
ReCogDrive (Run-then-Walk, ours)	✔	✗	**98.6**	**97.9**	85.9	**95.8**	100.0	**91.0**
AutoDrive-P^3 (Ye et al., 2026)	✔	✗	98.9	97.7	83.7	96.6	99.9	90.2
AutoDrive-P^3 (Run-then-Walk, ours)	✔	✗	98.6	97.3	**88.9**	95.5	**100.0**	**91.8**

Table 2: **Performance comparison on NAVSIMv2 benchmark (Navtest).**

Method	NC↑	DAC↑	DDC↑	TLC↑	EP↑	TTC↑	LK↑	HC↑	EC↑	EPDMS↑
Human	100.0	100.0	99.8	100.0	87.4	100.0	100.0	98.1	90.1	94.5
Ego Status MLP	93.1	77.9	92.7	99.6	86.0	91.5	89.4	98.3	85.4	64.0
Transfuser (Prakash et al., 2021)	96.9	89.9	97.8	99.7	87.1	95.4	92.7	98.3	87.2	84.0
DiffusionDrive (Liao et al., 2025)	98.2	96.2	99.5	99.8	87.4	97.3	96.9	98.4	87.7	88.2
WoTE (Li et al., 2025b)	98.5	96.8	98.8	99.8	86.1	97.9	95.5	98.3	82.9	87.7
DriveWorld-VLA (Liu et al., 2026)	98.6	99.1	99.6	99.8	87.4	97.9	97.0	97.8	78.6	86.8
DriveVLA-W0 (Li et al., 2025a)	98.5	99.1	98.0	99.7	86.4	98.1	93.2	97.9	58.9	86.1
Latent-WAM (Wang et al., 2026)	98.1	97.3	99.6	99.8	87.7	97.3	97.6	98.1	87.3	89.3
ReCogDrive (Li et al., 2025c)	98.2	97.7	98.4	100.0	89.9	97.4	90.8	97.7	26.8	82.7
ReCogDrive (Run-then-Walk, ours)	**98.5**	**97.9**	**98.8**	100.0	89.4	**97.8**	**92.0**	**98.2**	**29.8**	**83.7**
AutoDrive-P^3 (Ye et al., 2026)	98.9	97.6	98.9	99.8	86.8	98.5	95.4	98.3	80.6	88.7
AutoDrive-P^3 (Run-then-Walk, ours)	98.6	97.3	98.7	99.7	**92.8**	98.1	**96.1**	97.5	78.3	**89.6**

Starting from the SFT checkpoint π_{sft}, we first run GRPO with $R^{\text{run}}_{\text{plan}}$ and use π_{sft} as the reference policy. This Run stage expands the progress distribution and discovers high-progress modes. We then select a Run checkpoint π_{run} and continue GRPO with $R^{\text{walk}}_{\text{plan}}$, using π_{run} as the new reference policy. The Walk stage therefore repairs safety around the progress-oriented behaviors discovered in Run, rather than collapsing back to the original conservative SFT solution.

4.3 Idealized Theoretical Justification

The following analysis is idealized and serves as a reward-direction explanation rather than a convergence guarantee. Run-then-Walk separates progress discovery from safety repair. On a fixed query distribution $q \sim \rho$, let $\pi(x \mid q)$ denote a complete planner-sample distribution and define

$$J(\pi) = \mathbb{E}_{\rho,\pi}[SQ], \qquad Q = \alpha P + (1-\alpha)B, \qquad 0 \le S, P, B \le 1, \tag{8}$$

where P is progress, S contains multiplicative safety/compliance factors, and B contains the remaining quality terms. This covers PDMS ($S = \text{NC}\,\text{DAC}$, $\alpha = 5/12$) and EPDMS ($S = \text{NC}\,\text{DAC}\,\text{DDC}\,\text{TLC}$, $\alpha = 5/16$). Since $J \le \min\{\mathbb{E}[S], \mathbb{E}[Q]\}$, progress alone cannot compensate for poor safety.

In an idealized population update anchored at $\pi_R = \pi_{\text{run}}$, maximizing $\mathbb{E}[U] - \beta D_{\text{KL}}(\pi \| \pi_R)$ reweights samples by $e^{U/\beta}$. Walk sets $W = R^{\text{walk}}_{\text{plan}}$ to zero when the safety gate fails and gives $W \ge 3/2$ when it passes. Thus a positive total-reward gap increases the probability of gate-passing samples. If the range of unchanged auxiliary rewards is at most ω, a sufficient gap is $(3/2)\lambda_{\text{plan}} - \omega > 0$. This condition is stated per implementation and makes the argument independent of any particular planner architecture.

Table 3: **Performance comparison on Navhard benchmark.**

Methods	EPDMS↑	Stage	NC↑	DAC↑	DDC↑	TLC↑	EP↑	TTC↑	LK↑	HC↑	EC↑
LTF (Chitta et al., 2022)	23.1	1	96.2	79.5	99.1	99.5	84.1	95.1	94.2	97.5	79.1
		2	77.7	70.2	84.2	98.0	85.1	75.6	45.4	95.7	75.9
DiffusionDrive (Liao et al., 2025)	28.9	1	96.8	88.2	99.3	99.3	84.5	94.6	95.5	97.5	79.1
		2	80.3	74.4	86.1	98.4	87.4	76.9	50.4	95.5	69.4
GoalFlow (Xing et al., 2025b)	28.7	1	96.0	92.6	99.3	99.3	84.0	95.7	97.1	97.5	40.4
		2	79.4	78.2	86.4	97.7	86.5	76.0	45.5	94.4	40.4
ReCogDrive (Li et al., 2025c)	26.5	1	95.4	90.0	96.0	99.6	**86.3**	94.4	90.9	97.1	28.9
		2	76.8	72.6	80.4	98.3	**90.5**	73.8	42.2	95.6	34.7
ReCogDrive (Run-then-Walk, ours)	**28.2**	1	**95.6**	**91.3**	**96.8**	**99.8**	85.9	**95.3**	**92.2**	**97.8**	**32.9**
		2	**78.3**	**73.3**	**81.0**	**98.8**	89.2	**75.0**	**43.0**	**95.8**	**36.7**
AutoDrive-P^3 (Ye et al., 2026)	29.3	1	96.8	**88.7**	**97.1**	**99.6**	84.0	95.1	93.1	97.6	**68.4**
		2	81.2	68.3	**82.7**	**98.4**	86.4	**77.8**	42.3	**95.4**	**62.4**
AutoDrive-P^3 (Run-then-Walk, ours)	**31.3**	1	**97.1**	86.9	96.9	99.3	**89.9**	**95.8**	**96.9**	**97.6**	61.8
		2	**81.6**	**72.1**	81.5	98.0	**91.5**	76.3	**50.1**	95.0	52.0

Table 4: **Performance comparison on nuScenes Benchmark.**

Method	L2 (m) ↓				Collision (%) ↓				VLM
	1s	2s	3s	Avg.	1s	2s	3s	Avg.	
ST-P3 (Hu et al., 2022)	1.33	2.11	2.90	2.11	0.23	0.62	1.27	0.71	-
VAD (Jiang et al., 2023)	0.17	0.34	0.60	0.37	0.07	0.10	0.24	0.14	-
Ego-MLP (Li et al., 2024c)	0.46	0.76	1.12	0.78	0.21	0.35	0.58	0.38	-
UniAD (Hu et al., 2023)	0.44	0.67	0.96	0.69	0.04	0.08	0.23	0.12	-
InsightDrive (Song et al., 2025)	0.23	0.41	0.68	0.44	0.09	0.10	0.27	0.15	-
GPT-Driver (Mao et al., 2023)	0.20	0.40	0.70	0.44	0.04	0.12	0.36	0.17	GPT-3.5
DriveVLM (Tian et al., 2024)	0.18	0.34	0.68	0.40	0.10	0.22	0.45	0.27	Qwen2-VL-7B
OpenEMMA (Xing et al., 2025a)	1.45	3.21	3.76	2.81	–	–	–	–	Qwen2-VL-7B
RDA-Driver (Huang et al., 2024)	0.17	0.37	0.69	0.40	0.01	0.05	0.26	0.10	LLaVA-7B
OmniDrive (Wang et al., 2024)	0.14	0.29	0.55	0.33	0.01	0.04	0.27	0.11	LLaVA-7B
OpenDriveVLA (Zhou et al., 2025a)	0.14	0.30	0.55	0.33	0.02	0.07	0.22	0.10	Qwen2.5-VL-3B
AutoVLA (Zhou et al., 2025b)	0.25	0.46	0.73	0.48	0.07	0.07	0.26	0.13	Qwen2.5-VL-3B
AutoDrive-R^2 (Yuan et al., 2025)	0.35	0.49	0.62	0.49	–	–	–	–	Qwen2.5-VL-3B
AutoDrive-P^3 (Ye et al., 2026)	0.16	0.31	0.56	0.34	**0.00**	0.04	0.20	0.08	Qwen2.5-VL-3B
AutoDrive-P^3 (Run-then-Walk, ours)	**0.15**	**0.30**	**0.55**	**0.33**	0.01	**0.03**	**0.16**	**0.06**	Qwen2.5-VL-3B

If the achieved full-sample KL to π_R is at most κ, Pinsker's inequality gives $\varepsilon_\kappa = \min\{1, \sqrt{\kappa/2}\}$ and, for $a_R = \Pr_R(P \geq p_0)$ and $f_W = \Pr_W(S < s_0)$,

$$\mathbb{E}_W[P] \geq \mathbb{E}_R[P] - \varepsilon_\kappa, \qquad J(\pi_W) \geq \alpha p_0 s_0 [a_R - \varepsilon_\kappa - f_W]_+. \tag{9}$$

Run must therefore create high-progress mass, Walk must reduce safety failures, and the reference must retain enough of that mass. Writing $s_t = \mathbb{E}_t[S]$ and $c_t = \mathbb{E}_t[SQ]/s_t$, a quality-loss bound $c_W \geq c_R - \ell$ yields

$$J(\pi_W) - J(\pi_R) \geq (s_W - s_R)c_R - s_W \ell, \tag{10}$$

so safety improvement must exceed quality lost among safety-weighted trajectories.

These statements require positive Run support for useful safe-progress samples, adequate group coverage, a fixed evaluation distribution, and a measured rather than nominal KL bound. A zero KL forbids repair, and our experiments further show that removing the KL term leads to training collapse; excessive KL weakens progress retention. The Walk gate is not identical to EPDMS safety, and endpoint proximity is neither a path-safety nor a progress certificate. Full derivations, the idealized probability-ratio bound, and finite-sampling conditions are provided in Appendix B.

5 Experiments

Benchmarks and baselines. We evaluate Run-then-Walk on four benchmarks: NAVSIMv1 (Dauner et al., 2024), NAVSIMv2 (Cao et al., 2025), Navhard (Cao et al., 2025), and nuScenes (Caesar et al., 2020). To validate the robustness of our method, we evaluate it on two representative VLM planners: the autoregressive AutoDrive-P^3 (fast-mode) and the diffusion-based ReCogDrive. NAVSIMv1 uses the Predictive Driver Model Score (PDMS), whereas NAVSIMv2 and Navhard use the Extended PDMS (EPDMS). nuScenes provides trajectory-error and collision evaluation. Since only

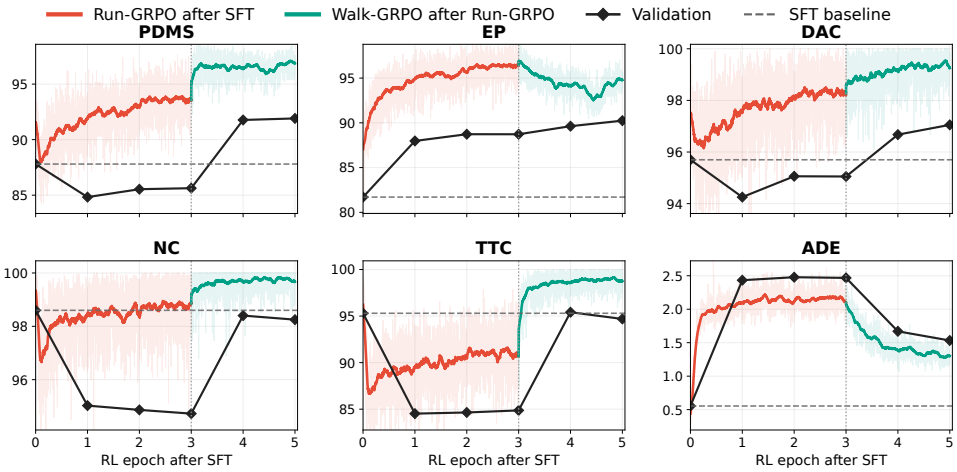


Figure 5: **PDMS sub-metric evolution during AutoDrive-P^3 Run-then-Walk strategy.** Red curves denote the 3-epoch Run-GRPO after SFT, green curves denote the following 2-epoch Walk-GRPO after Run-GRPO. Run-GRPO increases progress but hurts safety; Walk-GRPO then restores safety, lifts PDMS above the SFT baseline, and yields a safe, high-progress policy in only 5 epochs.

Table 5: **Run/Walk recipe comparison.**

Different RL recipes	NC↑	DAC↑	EP↑	TTC↑	Comf↑	PDMS↑
Only Run	94.8	95.1	88.8	84.8	**100.0**	85.7
Only Walk	**99.1**	**97.7**	81.5	**97.3**	**100.0**	89.7
Walk-and-Run	98.0	94.2	85.4	94.0	**100.0**	85.9
Walk-then-Run	98.9	**97.7**	83.7	96.6	99.9	90.2
Ours (Run-then-Walk)	**98.6**	**97.3**	**88.9**	95.5	**100.0**	**91.8**

Table 6: **Effect of safety metrics in *Walk*.**

Reward Signals			Evaluation Metrics					
DAC	NC	TTC	NC↑	DAC↑	EP↑	TTC↑	Comf↑	PDMS↑
✔	✗	✗	98.5	97.1	88.0	94.9	100.0	91.5
✔	✔	✗	98.6	97.3	**88.9**	95.5	100.0	**91.8**
✔	✔	✔	**98.7**	**97.4**	86.9	**96.0**	100.0	91.2

AutoDrive-P^3 has been trained on nuScenes, we restrict the nuScenes comparison to this planner. We set R_{agg} to PDMS, and for the endpoint reward, we set $\Delta = 10$ and $\eta = 0.2$ in all experiments.

Main results across planners and benchmarks. Tables 1 and 2 show consistent improvements for both planner families rather than a gain tied to one base model. On NAVSIMv1, Run-then-Walk raises the paired diffusion-planner PDMS from 90.6 to 91.0 and the paired autoregressive-planner PDMS from 90.2 to 91.8. The ReCogDrive comparison improves all safety-related metrics, only with a small decrease in EP, whereas the AutoDrive-P^3 comparison highlights increased progress while retaining strong safety. For AutoDrive-P^3, the schedule uses 3 Run epochs and 2 Walk epochs, for 5 RL epochs in total. For ReCogDrive, we use 5 Run epochs and 1 Walk epoch, for 6 RL epochs in total. Thus the method saves 40–50% of the 10-epoch baseline training steps, converges faster, and improves performance for both planners. On NAVSIMv2 (Table 2), paired EPDMS rises from 82.7 to 83.7 for ReCogDrive and from 88.7 to 89.6 for AutoDrive-P^3 with EP 92.8. On the two-stage Navhard protocol, as shown in Table 3, paired aggregates improve from 26.5 to 28.2 and from 29.3 to 31.3, respectively. On nuScenes, as shown in Table 4, AutoDrive-P^3 planner reduces average trajectory error and collision rate. Across different benchmarks and planner methods, our Run-then-Walk strategy not only better balances safety and progress, thereby improving driving performance, but also substantially saves training time.

6 Analysis and Visualization

Analysis of Run-then-Walk Training Strategy. Figure 5 reveals the behavioral evolution of AutoDrive-P^3 across the two training stages. During *Run*-GRPO, the EP rises substantially (81.7→88.8), but the safety metrics drop, indicating that the policy learns to maximize progress at the cost of frequent collisions and unsafe events, ultimately degrading PDMS from 87.8 to 85.7. In the subsequent *Walk*-GRPO, safety is strongly restored, while the high EP is preserved, and the final checkpoint reaches 91.8 PDMS. This AutoDrive-P^3 result uses only 5 RL epochs, and the same safety-repair principle applied to ReCogDrive reaches 91.0 PDMS in 6 epochs, together saving 40–50% of RL training steps relative to the 10-epoch baselines while improving performance. More detailed analysis, including ReCogDrive, is provided in the supplementary material.

Analysis of different RL recipes. Table 5 compares different orderings of Run and Walk objectives. *Only Run* yields high EP (88.8) but the lowest PDMS (85.7) due to poor safety; *Only Walk* achieves

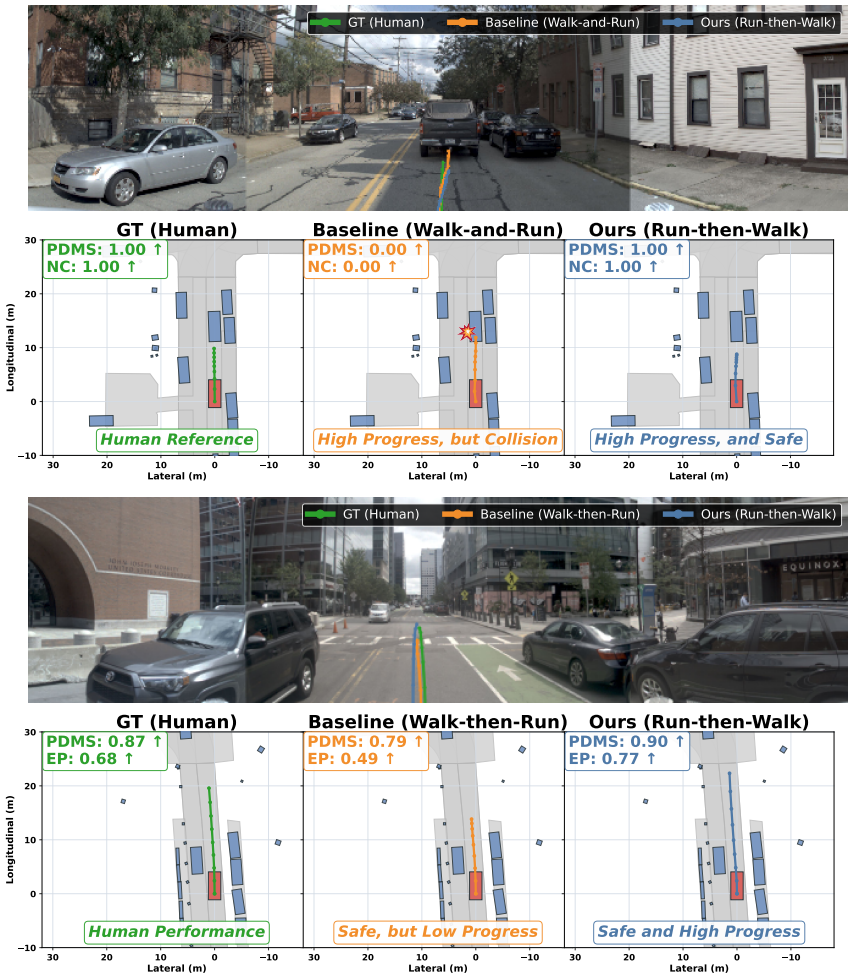


Figure 6: **Qualitative trajectory visualization.**

the best safety but limits progress (EP 81.5, PDMS 89.7). *Walk-then-Run* is suboptimal, whereas our *Run-then-Walk* achieves the highest PDMS by exploring progress first and restoring safety afterward.

Effect of safety signal composition. Table 6 ablates which safety signals are used in the Walk-phase planning reward $R_{\text{plan}}^{\text{walk}}$ (Eq. 4). Using only DAC yields 91.5 PDMS. Adding NC improves the balance to 91.8 PDMS and 88.9 EP. Further adding TTC raises safety scores but lowers EP to 86.9, reducing PDMS to 91.2. These results suggest that DAC+NC provides the best safety–progress trade-off, while TTC as a reward signal introduces over-conservatism.

Visualization. Figure 6 presents qualitative trajectory comparisons on two representative driving scenarios. Walk-and-Run achieves high progress but collides (NC $= 0$), and Walk-then-Run remains safe but advances less (EP $= 0.65$). Our method maintains a high EP while ensuring safety.

7 Conclusion

In this paper, we first reveal two RL training recipe regimes: a progress regime (Run-GRPO) that aggressively explores high progress and a safety regime (Walk-GRPO) that restores safety. Then, we present *Run-then-Walk*, a simple and effective two-stage RL training scheduling strategy for VLM-based autonomous driving. By first exploring progress under relaxed constraints and then repairing safety with endpoint rewards, our method escapes SFT conservatism while avoiding the unsafe driving. Across VLM-based autoregressive and diffusion planners on different benchmarks, experiments show improved driving performance and a 40–50% reduction in RL training steps. These findings demonstrate that a progress-first, safety-refinement curriculum is efficient and effective, and we hope this simple paradigm inspires future RL in broader embodied decision-making.

AI USE STATEMENT

In this work, we used generative AI tools for editing the manuscript and for assisting in literature search to identify relevant papers. We have not used generative AI tools for other tasks with required disclosure, and the rest of the required disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for language polishing. We have reviewed all AI-assisted work: the AI-suggested references were verified through a manual literature survey, and the AI-edited text was checked by the authors for correctness and clarity. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work adheres to the ICLR Code of Ethics and all authors have read and adhered to the Code of Ethics. In this study, no human subjects is involved. The use of all datasets, including nuScenes (Caesar et al., 2020) and NAVSIM (Dauner et al., 2024; Cao et al., 2025), follows the relevant usage guidelines and public licenses, ensuring no violation of privacy. We have been careful to avoid any biased or discriminatory results during our research process. No personally identifiable information is used, and no privacy or security concerns will be raised due to our experiments. We are committed to maintaining transparency and integrity throughout the research process.

REPRODUCIBILITY STATEMENT

The training objective and reward definitions appear in Section 4 and Appendix A; the algorithm and hyperparameters are in Appendix C.

REFERENCES

Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11621–11631, 2020.

Wei Cao, Marcel Hallgarten, Tianyu Li, Daniel Dauner, Xunjiang Gu, Caojun Wang, Yakov Miron, Marco Aiello, Hongyang Li, Igor Gilitschenski, et al. Pseudo-simulation for autonomous driving. *arXiv preprint arXiv:2506.04218*, 2025.

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.

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.

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

Yufeng Hong, Xiaotian Zhou, Yingyan Li, Xiangpo Zhou, Lin Liu, Yadan Luo, Shaoqing Xu, Lei Yang, and Ziying Song. Drivefuture: Future-aware latent world models for autonomous driving. *arXiv preprint arXiv:2605.09701*, 2026.

Shengchao Hu, Li Chen, Penghao Wu, Hongyang Li, Junchi Yan, and Dacheng Tao. St-p3: End-to-end vision-based autonomous driving via spatial-temporal feature learning. In *European Conference on Computer Vision*, pp. 533–549. Springer, 2022.

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*, pp. 17853–17862, 2023.

Wenxuan Huang, Bohan Jia, Zijie Zhai, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Yao Hu, and Shaohui Lin. Vision-r1: Incentivizing reasoning capability in multimodal large language models. *arXiv preprint arXiv:2503.06749*, 2025.

Zhijian Huang, Tao Tang, Shaoxiang Chen, Sihao Lin, Zequn Jie, Lin Ma, Guangrun Wang, and Xiaodan Liang. Making large language models better planners with reasoning-decision alignment. In *European Conference on Computer Vision*, pp. 73–90. Springer, 2024.

Jyh-Jing Hwang, Runsheng Xu, Hubert Lin, Wei-Chih Hung, Jingwei Ji, Kristy Choi, Di Huang, Tong He, Paul Covington, Benjamin Sapp, et al. Emma: End-to-end multimodal model for autonomous driving. *arXiv preprint arXiv:2410.23262*, 2024.

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

Bo Jiang, Shaoyu Chen, Qian Zhang, Wenyu Liu, and Xinggang Wang. Alphadrive: Unleashing the power of vlms in autonomous driving via reinforcement learning and reasoning. *arXiv preprint arXiv:2503.07608*, 2025.

Yingyan Li, Lue Fan, Jiawei He, Yuqi Wang, Yuntao Chen, Zhaoxiang Zhang, and Tieniu Tan. Enhancing end-to-end autonomous driving with latent world model. *arXiv preprint arXiv:2406.08481*, 2024a.

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*, 2025a.

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. *arXiv preprint arXiv:2504.01941*, 2025b.

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

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*, 2024b.

Zhiqi Li, Zhiding Yu, Shiyi Lan, Jiahan Li, Jan Kautz, Tong Lu, and Jose M Alvarez. Is ego status all you need for open-loop end-to-end autonomous driving? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14864–14873, 2024c.

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*, pp. 12037–12047, 2025.

Lin Liu, Ziying Song, Caiyan Jia, Hangjun Ye, Xiaoshuai Hao, Long Chen, et al. Driveworld-vla: Unified latent-space world modeling with vision-language-action for autonomous driving. *arXiv preprint arXiv:2602.06521*, 2026.

Jiageng Mao, Yuxi Qian, Junjie Ye, Hang Zhao, and Yue Wang. Gpt-driver: Learning to drive with gpt. *arXiv preprint arXiv:2310.01415*, 2023.

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

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.

Zhihong Shao, Yuxiang Luo, Chengda Lu, ZZ Ren, Jiewen Hu, Tian Ye, Zhibin Gou, Shirong Ma, and Xiaokang Zhang. Deepseekmath-v2: Towards self-verifiable mathematical reasoning. *arXiv preprint arXiv:2511.22570*, 2025.

Ruiqi Song, Xianda Guo, Hangbin Wu, Qinggong Wei, and Long Chen. Insightdrive: Insight scene representation for end-to-end autonomous driving. *arXiv preprint arXiv:2503.13047*, 2025.

Xiaoyu Tian, Junru Gu, Bailin Li, Yicheng Liu, Yang Wang, Zhiyong Zhao, Kun Zhan, Peng Jia, Xianpeng Lang, and Hang Zhao. Drivevlm: The convergence of autonomous driving and large vision-language models. *arXiv preprint arXiv:2402.12289*, 2024.

Linbo Wang, Yupeng Zheng, Qiang Chen, Shiwei Li, Yichen Zhang, Zebin Xing, Qichao Zhang, Xiang Li, Deheng Qian, Pengxuan Yang, et al. Latent-wam: Latent world action modeling for end-to-end autonomous driving. *arXiv preprint arXiv:2603.24581*, 2026.

Shihao Wang, Zhiding Yu, Xiaohui Jiang, Shiyi Lan, Min Shi, Nadine Chang, Jan Kautz, Ying Li, and Jose M Alvarez. Omnidrive: A holistic llm-agent framework for autonomous driving with 3d perception, reasoning and planning. *CoRR*, 2024.

Xinshuo Weng, Boris Ivanovic, Yan Wang, Yue Wang, and Marco Pavone. Para-drive: Parallelized architecture for real-time autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15449–15458, 2024.

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*, pp. 1001–1009, 2025a.

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 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1602–1611, June 2025b.

Yi Xu, Yuxin Hu, Zaiwei Zhang, Gregory P Meyer, Siva Karthik Mustikovela, Siddhartha Srinivasa, Eric M Wolff, and Xin Huang. Vlm-ad: End-to-end autonomous driving through vision-language model supervision. *arXiv preprint arXiv:2412.14446*, 2024.

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

Zhenlong Yuan, Jing Tang, Jinguo Luo, Rui Chen, Chengxuan Qian, Lei Sun, Xiangxiang Chu, Yujun Cai, Dapeng Zhang, and Shuo Li. Autodrive-r²: Incentivizing reasoning and self-reflection capacity for vla model in autonomous driving. *arXiv preprint arXiv:2509.01944*, 2025.

Xingcheng Zhou, Xuyuan Han, Feng Yang, Yunpu Ma, and Alois C Knoll. Opendrivevla: Towards end-to-end autonomous driving with large vision language action model. *arXiv preprint arXiv:2503.23463*, 2025a.

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

A Reward Function Details

A.1 Planner-Agnostic Reward Interface

For each driving query q, we sample a group of G complete planner outputs $\{x_i\}_{i=1}^{G}$ from π_θ and decode each output into a trajectory τ_i. An autoregressive planner assigns the resulting reward to the sampled token sequence, whereas a diffusion planner assigns it to the sampled continuous trajectory. In both cases, the total reward is

$$R_i^{(s)} = R_{\text{aux}}(x_i) + \lambda_{\text{plan}} R_{\text{plan}}^{(s)}(\tau_i), \qquad s \in \{\text{run}, \text{walk}\}. \tag{11}$$

Here R_{aux} denotes any native auxiliary objectives of the underlying planner. These objectives and the output representation remain unchanged between the two stages. Rewards are normalized within each sampled group and optimized with the GRPO objective in Section 4. Consequently, the only algorithmic change shared by both planner families is the planning reward and its Run-to-Walk schedule.

A.2 Planning Reward: Run vs. Walk

The planning reward is the only reward component that differs between the Run and Walk stages.

Run Phase Planning Reward. In the Run phase, we directly use the closed-loop PDMS as the planning reward:

$$R_{\text{plan}}^{\text{run}}(\tau_i) = \text{PDMS}(\tau_i). \tag{12}$$

No endpoint/L2 constraint or explicit safety reward is applied. This intentionally encourages the policy to maximize progress under relaxed constraints.

Walk Phase Planning Reward. In the Walk phase, we replace the PDMS-based planning reward with a composite of endpoint and safety rewards. The endpoint reward is defined via the L_1 distance $d_E = \|p_T - g_T\|_1$ between the final predicted point p_T and the expert endpoint g_T:

$$k(d_E) = \min(K,\ \max(0, \lfloor (d_E - 2\Delta)/\Delta \rfloor + 1)), \qquad R_{\text{end}}(d_E) = 1 - \eta\, k(d_E), \tag{13}$$

where $\Delta > 0$ is the step size, $\eta \in (0, 1)$ is the reward decrement, and $K = \lfloor 1/\eta \rfloor$. We use the fixed configuration $\Delta = 10$, $\eta = 0.2$ throughout. The complete Walk planning reward is:

$$R_{\text{plan}}^{\text{walk}}(\tau_i, \tau_i^\star) = \mathbb{I}_{\text{safe}}(\tau_i) \Big(\text{NC}(\tau_i) + \text{DAC}(\tau_i) + R_{\text{end}}(d_E) \Big), \tag{14}$$

where $\mathbb{I}_{\text{safe}}(\tau_i) = 1$ only if $\text{NC}(\tau_i) > 0$, and $\text{DAC}(\tau_i) > 0$; otherwise the entire planning reward is zeroed. Algorithm 1 presents the complete Run-then-Walk scheduling strategy procedure.

B Idealized Theoretical Justification

This appendix provides the full derivations for the idealized theoretical justification summarized in Section 4.3. They are conditional statements about a fixed query distribution, not a claim that every finite-sample GRPO run must improve. Let $q \sim \rho$ and let x be a complete planner sample (an autoregressive response or a diffusion sample) with induced trajectory $\tau(x)$. We write

$$J(\pi) = \mathbb{E}_{\rho,\pi}[SQ], \qquad Q = \alpha P + (1 - \alpha) B, \qquad 0 \le S, P, B \le 1, \tag{15}$$

where P is normalized progress, S is the product of safety/compliance factors, and B contains the remaining quality terms. For PDMS, $S = \text{NC}\,\text{DAC}$ and $\alpha = 5/12$; for EPDMS, $S = \text{NC}\,\text{DAC}\,\text{DDC}\,\text{TLC}$ and $\alpha = 5/16$. Since $0 \le S, Q \le 1$, $J \le \mathbb{E}[S]$ and $J \le \mathbb{E}[Q]$. This is the basic progress–safety bottleneck: high progress cannot compensate for a vanishing safety factor.

Safety direction of the Walk reward. Fix q and let $\pi_R = \pi_{\text{run}}$, $H = \mathbb{I}_{\text{safe}}$, and $W = R_{\text{plan}}^{\text{walk}}$. Let C denote the unchanged architecture-native auxiliary reward and $U = C + \lambda_{\text{plan}} W$. Consider the idealized population problem

$$\pi^\dagger = \arg\max_{\pi} \left\{ \mathbb{E}_\pi[U] - \beta D_{\text{KL}}(\pi \| \pi_R) \right\}, \qquad \pi^\dagger(x \mid q) = \frac{\pi_R(x \mid q) e^{U(q,x)/\beta}}{Z_q}, \tag{16}$$

Algorithm 1 Run-then-Walk training scheduling strategy

Require: SFT policy π_{sft}, queries $\mathcal{Q}$, group size G, auxiliary reward R_{aux}, planning weight λ_{plan}, KL coefficients $\beta_{\text{run}}, \beta_{\text{walk}}$
Ensure: Optimized driving policy π_{walk}

— Stage 1: Run-GRPO (progress discovery) —
Initialize $\pi \leftarrow \pi_{\text{sft}}$, set reference $\pi_{\text{ref}} \leftarrow \pi_{\text{sft}}$
for each query $q \in \mathcal{Q}$ **do**
  Sample outputs $\{x_i\}_{i=1}^{G} \sim \pi(\cdot \mid q)$ and decode trajectories $\{\tau_i\}_{i=1}^{G}$
  for $i = 1$ to G **do**
    $R^i_{\text{plan}} \leftarrow R^{run}_{\text{plan}}(\tau_i)$ ▷ Focus on progress
    $R_i \leftarrow R_{\text{aux}}(x_i) + \lambda_{\text{plan}} R^i_{\text{plan}}$
  end for
  Compute advantages $\hat{A}_i = (R_i - \bar{R})/\sigma_R$ and update π via GRPO with KL penalty β_{run}
end for
Select Run checkpoint π_{run}

— Stage 2: Walk-GRPO (safety repair) —
Initialize $\pi \leftarrow \pi_{\text{run}}$, set reference $\pi_{\text{ref}} \leftarrow \pi_{\text{run}}$
for each query $q \in \mathcal{Q}$ **do**
  Sample outputs $\{x_i\}_{i=1}^{G} \sim \pi(\cdot \mid q)$ and decode trajectories $\{\tau_i\}_{i=1}^{G}$
  for $i = 1$ to G **do**
    $R^i_{\text{plan}} \leftarrow R^{\text{walk}}_{\text{plan}}(\tau_i, \tau_i^\star)$ ▷ Endpoint + safety gate
    $R_i \leftarrow R_{\text{aux}}(x_i) + \lambda_{\text{plan}} R^i_{\text{plan}}$
  end for
  Compute advantages $\hat{A}_i = (R_i - \bar{R})/\sigma_R$ and update π via GRPO with KL penalty β_{walk}
end for
return π_{walk}

where $Z_q = \mathbb{E}_{\pi_R}[e^{U/\beta}]$ and $\beta > 0$. The variational identity

$$\mathbb{E}_\pi[U] - \beta D_{\text{KL}}(\pi \| \pi_R) = \beta \log Z_q - \beta D_{\text{KL}}(\pi \| \pi^\dagger) \tag{17}$$

proves the optimizer. Suppose both values of H have positive reference probability and every passing sample has total reward at least $\gamma > 0$ larger than every failing sample. If $h_R = \Pr_{\pi_R}(H = 1 \mid q)$ and $h^\dagger = \Pr_{\pi^\dagger}(H = 1 \mid q)$, then

$$\frac{h^\dagger}{1 - h^\dagger} = \frac{h_R}{1 - h_R} \frac{\mathbb{E}_{\pi_R}[e^{U/\beta} \mid H = 1]}{\mathbb{E}_{\pi_R}[e^{U/\beta} \mid H = 0]} \geq e^{\gamma/\beta} \frac{h_R}{1 - h_R}, \tag{18}$$

and therefore

$$1 - h^\dagger \leq \frac{1 - h_R}{1 - h_R + h_R e^{\gamma/\beta}}. \tag{19}$$

The Walk construction supplies a concrete sufficient reward gap. Recall that

$$R^{\text{walk}}_{\text{plan}}(\tau_i, \tau_i^\star) = \mathbb{I}_{\text{safe}}(\tau_i) \Big(\text{NC}(\tau_i) + \text{DAC}(\tau_i) + R_{\text{end}}(d_E) \Big),$$

where $\text{NC} \in \{0, 1/2, 1\}$, $\text{DAC} \in \{0, 1\}$, and $R_{\text{end}} \in [0, 1]$. When the safety gate passes, we have $\text{NC} > 0$ and $\text{DAC} > 0$; hence $\text{NC} \geq 1/2$, $\text{DAC} = 1$, and $R_{\text{end}} \geq 0$. Therefore

$$W = R^{\text{walk}}_{\text{plan}} \geq \frac{1}{2} + 1 + 0 = \frac{3}{2}.$$

When the gate fails, $\mathbb{I}_{\text{safe}} = 0$ and thus $W = 0$. Let C denote the unchanged auxiliary reward, whose range at a fixed query is at most ω, and let the total reward be $U = C + \lambda_{\text{plan}} W$. Then every passing sample satisfies

$$U_{\text{pass}} \geq C_{\min} + \frac{3}{2} \lambda_{\text{plan}},$$

while every failing sample satisfies

$$U_{\text{fail}} \le C_{\max}.$$

Hence the reward gap is at least

$$U_{\text{pass}} - U_{\text{fail}} \ge \frac{3}{2}\lambda_{\text{plan}} - (C_{\max} - C_{\min}) \ge \frac{3}{2}\lambda_{\text{plan}} - \omega.$$

Thus $\gamma = (3/2)\lambda_{\text{plan}} - \omega > 0$ is sufficient for all passing samples to dominate all failing samples in total reward. The value of ω must be established from the actual normalization and scale of the auxiliary rewards for the implementation under study; it is not fixed by the gate construction alone.

The idealized result describes the local direction of group-normalized GRPO as well. At the Run checkpoint, clipping is inactive and the KL term has zero first derivative. Along $\pi_\zeta(x \mid q) \propto \pi_R(x \mid q)e^{\zeta H(x)}$, the derivative of the sampled surrogate at $\zeta = 0$ is $G^{-1}\sum_i A_i H_i$, where A_i are fixed Walk advantages and $H_i = H(x_i)$. Since $\sum_i A_i = 0$ and all passing rewards exceed all failing rewards, this derivative is nonnegative and is positive when a group contains both outcomes. Shared neural parameters, finite groups, advantage recomputation, and clipping mean that the actual update need not realize this direction or the Gibbs optimizer. The parameter β in Eq. (16) is therefore not an empirical improvement-rate calibration.

Progress retention under a KL budget. Let $\pi_W = \pi_{\text{walk}}$ and assume the achieved full-sample KL on the same ρ obeys

$$\mathbb{E}_{q\sim\rho} D_{\text{KL}}\big(\pi_W(\cdot \mid q) \| \pi_R(\cdot \mid q)\big) \le \kappa, \qquad \varepsilon_\kappa = \min\{1, \sqrt{\kappa/2}\}. \tag{20}$$

Pinsker's inequality gives conditional total variation at most $\sqrt{D_{\text{KL}}/2}$; Jensen's inequality then bounds its mean by ε_κ. Hence any $[0,1]$-valued statistic changes by at most ε_κ, including P and event indicators. For $a_R = \Pr_R(P \ge p_0)$ and $f_W = \Pr_W(S < s_0)$,

$$\mathbb{E}_W[P] \ge \mathbb{E}_R[P] - \varepsilon_\kappa, \qquad \Pr_W(P \ge p_0, S \ge s_0) \ge [a_R - \varepsilon_\kappa - f_W]_+. \tag{21}$$

On the latter event, $SQ \ge \alpha p_0 s_0$, so

$$J(\pi_W) \ge \alpha p_0 s_0 [a_R - \varepsilon_\kappa - f_W]_+. \tag{22}$$

The bound is nonvacuous only when $a_R > \varepsilon_\kappa + f_W$. A GRPO KL coefficient does not by itself establish Eq. (20): the bound concerns the complete output distribution on ρ, not an unconverted average over tokens or diffusion steps. By data processing, full-output KL also bounds the induced trajectory KL. Conversely, reducing the probability of a fixed failure event by d requires $\kappa \ge 2d^2$ under this bound. Thus zero KL forbids repair, while a very large KL gives little progress retention.

When the final score increases. Safety improvement and progress can be correlated, so marginal submetric improvements are insufficient. Define $s_t = \mathbb{E}_t[S] > 0$ and $c_t = \mathbb{E}_t[SQ]/s_t$ for $t \in \{R, W\}$. If the safety-weighted quality loss is bounded by $c_W \ge c_R - \ell$, then

$$\begin{aligned} J(\pi_W) - J(\pi_R) &= s_W c_W - s_R c_R \\ &= (s_W - s_R)c_R + s_W(c_W - c_R) \\ &\ge (s_W - s_R)c_R - s_W \ell. \end{aligned} \tag{23}$$

Therefore $(s_W - s_R)c_R > s_W \ell$ is sufficient for strict improvement. This condition allows some scenes to worsen and can be estimated from paired per-trajectory scores; separate averages of NC, DAC, and EP do not prove it. If $s_R = 0$, any positive Walk score is an improvement.

Sampling and boundary conditions. For a query with $p_q = \Pr_R(H = 1, P \ge p_0 \mid q)$, an independent group of G samples contains a useful safe-progress candidate with probability $1 - (1 - p_q)^G$. When p_q is small, the Walk signal is rarely observed; if all group members fail the gate, their planning rewards are all zero and there is no relative planning signal from that term. Finite KL cannot create probability on a reference-null event. These facts justify switching after progress discovery but before the Run checkpoint loses useful safe support, without implying a universal epoch or schedule.

Finally, H is a training gate rather than the full evaluation factor S: it permits $\text{NC} = 1/2$, checks TTC, and omits EPDMS's DDC/TLC terms. Endpoint proximity similarly certifies neither path

Table 7: **RL training hyperparameters for different VLM-based planners.**

Hyperparameter	AutoDrive-P^3 (Ye et al., 2026)		ReCogDrive(Li et al., 2025c)	
	Run-GRPO	Walk-GRPO	Run-GRPO	Walk-GRPO
Learning Rate	1×10^{-6}	1×10^{-6}	1×10^{-4}	1×10^{-6}
LR Schedule	constant	constant	cosine decay	cosine decay
Optimizer	AdamW	AdamW	AdamW	AdamW
RL Epochs	3	2	5	1
Global Batch Size	32	32	32	32
GRPO Group Size (G)	8	8	8	8
Clip Parameter (ϵ)	0.2	0.2	0.2	0.2
Precision / Hardware	bf16 / 8×A100	bf16 / 8×A100	bf16 / 8×A100	bf16 / 8×A100
Trajectory Output	8 waypoints (4.0 s)	8 waypoints (4.0 s)	8 poses (4.0 s, 0.5 s)	8 poses (4.0 s, 0.5 s)
Planner Parameterization	Autoregressive tokens	Autoregressive tokens	Diffusion	Diffusion

Table 8: **Three-seed summary on NAVSIMv1 (training seeds).** Values are reported as mean ± standard deviation over three training seeds, with the test seed fixed to 1.

Planner	Training recipe	NC ↑	DAC ↑	EP ↑	TTC ↑	Comf ↑	PDMS ↑
AutoDrive-P^3	Original RL	98.87 ± 0.04	97.68 ± 0.02	83.72 ± 0.05	96.58 ± 0.03	99.97 ± 0.01	90.18 ± 0.06
	Run-then-Walk	98.57 ± 0.03	97.32 ± 0.04	88.87 ± 0.02	95.48 ± 0.05	100.00 ± 0.00	91.78 ± 0.03
ReCogDrive	Original RL	98.08 ± 0.11	97.68 ± 0.08	86.47 ± 0.13	94.88 ± 0.09	100.00 ± 0.00	90.58 ± 0.12
	Run-then-Walk	98.61 ± 0.12	97.88 ± 0.05	85.92 ± 0.10	95.78 ± 0.11	100.00 ± 0.00	91.02 ± 0.10

safety nor progress; a tight endpoint reward may increase ℓ, while a loose one may be nondiscriminative. Human-penalty filters, Navhard's multi-stage aggregation, and policy-induced query shifts require protocol-specific analysis. These qualifications apply equally to autoregressive samples and VLM-conditioned diffusion trajectories.

C Training Hyperparameters

Table 7 summarizes the RL training hyperparameters for different VLM-based planners and stages. Note that we stop the first (Run) stage once EP has essentially plateaued, as illustrated in Fig. 3. This is why ReCogDrive requires a longer Run stage than AutoDrive-P^3. AutoDrive-P^3 starts from an SFT checkpoint with EP around 81.7, whereas ReCogDrive starts from an SFT checkpoint with EP around 80. Therefore, ReCogDrive requires a longer progress-discovery phase before EP plateaus, specifically two additional Run epochs compared with AutoDrive-P^3, using five Run epochs in total.

In the Run stage, both AutoDrive-P^3 and ReCogDrive adopt Eq. 12 as the planning reward, i.e., PDMS is directly used as the planning reward to enhance progress exploration. In the Walk stage, both planners adopt Eq. 14 as the planning reward to improve safety. The difference is that AutoDrive-P^3 additionally includes a planning/perception auxiliary reward and combines it with the planning reward through Eq. 11, where $R_{\text{aux}}(x_i)$ denotes the auxiliary reward, to align with its original baseline (Ye et al., 2026). ReCogDrive does not require this additional auxiliary term, as it is consistent with its original baseline (Li et al., 2025c).

D Multi-Seed Evaluation on Training and Test Seeds

We evaluate the stability of each planner with respect to both training seeds and test seeds. For the training-seed study, we train each planner with three independent random seeds while fixing the test seed to 1. For the test-seed study, we train each planner with seed 1 and evaluate it with three different test seeds. All other settings are kept identical. Tables 8 and 9 show that the autoregressive planner (AutoDrive-P^3) is relatively stable across training seeds (around 0.03), whereas the diffusion-based planner (ReCogDrive) is less stable due to stochastic noise (around 0.1). The same trend holds for test seeds.

Table 9: **Three-seed summary on NAVSIMv1 (test seeds).** Values are reported as mean ± standard deviation over three test seeds, with the training seed fixed to 1.

Planner	Training recipe	NC ↑	DAC ↑	EP ↑	TTC ↑	Comf ↑	PDMS ↑
AutoDrive-P^3	Original RL	98.91 ± 0.02	97.71 ± 0.03	83.68 ± 0.04	96.61 ± 0.02	99.96 ± 0.01	90.21 ± 0.04
	Run-then-Walk	98.62 ± 0.03	97.35 ± 0.02	88.92 ± 0.03	95.51 ± 0.04	100.00 ± 0.00	91.82 ± 0.02
ReCogDrive	Original RL	98.12 ± 0.10	97.72 ± 0.07	86.53 ± 0.12	94.92 ± 0.08	100.00 ± 0.00	90.62 ± 0.11
	Run-then-Walk	98.65 ± 0.11	97.92 ± 0.04	85.95 ± 0.10	95.82 ± 0.09	100.00 ± 0.00	91.05 ± 0.09

E Stage-by-Stage Qualitative Walkthrough: SFT → Run → Walk

This section visualizes how a single scene evolves through the three training stages of our pipeline. In Figures 7 and 8, each row contains a wide camera strip (with all four trajectories projected) followed by four bird's-eye-view (BEV) panels that share an identical map/agent layout. From left to right the BEV panels show the *Human* expert (GT, green), the *SFT* policy (blue), the *Run-GRPO* checkpoint (red), and the final *Walk-GRPO* policy (orange). The PDMS and endpoint-progress (EP) scores are annotated inside every panel.

Stage 1: SFT (conservative imitation). The supervised policy faithfully imitates the human demonstrations and is therefore collision-free, but it is systematically *under-progressing*: it brakes early and stops short of where a competent human driver would be, yielding low EP and a capped PDMS (typically 0.83–0.93 in these scenes). The blue BEV panel is labeled *"Safe, but Low Progress."*

Stage 2: Run-GRPO (progress discovery). Optimizing the PDMS-only planning reward without any endpoint/safety constraint pushes the policy to discover much more aggressive, high-progress trajectory modes. As the red BEV panel (*"High Progress, but Collision"*) shows, the Run policy reaches far further down the road, but it frequently leaves the drivable area or collides with an agent, driving its no-collision (NC) score, and hence PDMS, to zero. This is the intentional, temporary safety regression that the Run phase trades for exploration.

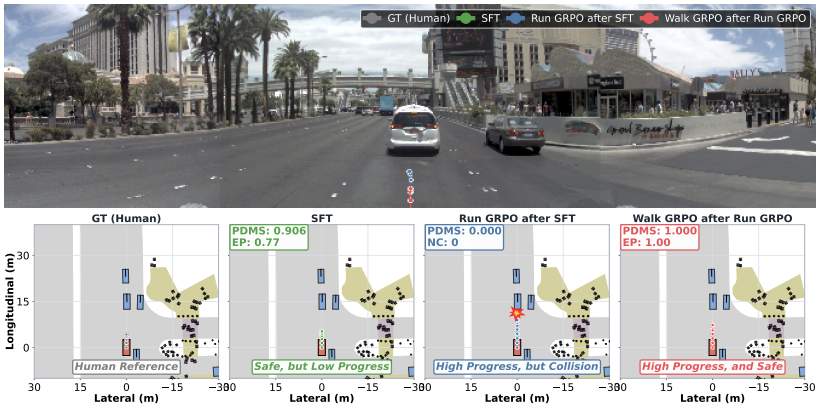
Stage 3: Walk-GRPO (safety repair). Re-introducing the endpoint and safety rewards *after* the Run phase lets the policy keep the newly discovered progress while repairing the unsafe behavior. The orange BEV panel (*"High Progress, and Safe"*) recovers a collision-free trajectory that nonetheless travels almost as far as the Run policy, lifting both EP and PDMS well above the SFT baseline (often to 0.95–1.00 PDMS). Across the six scenes below, spanning straight cruising, signalized intersections, crosswalk approaches, and following large vehicles (buses/trucks), the Walk policy consistently dominates SFT on progress while matching its safety, which is precisely the behavior our Run-then-Walk ordering is designed to produce.

F Qualitative Comparison

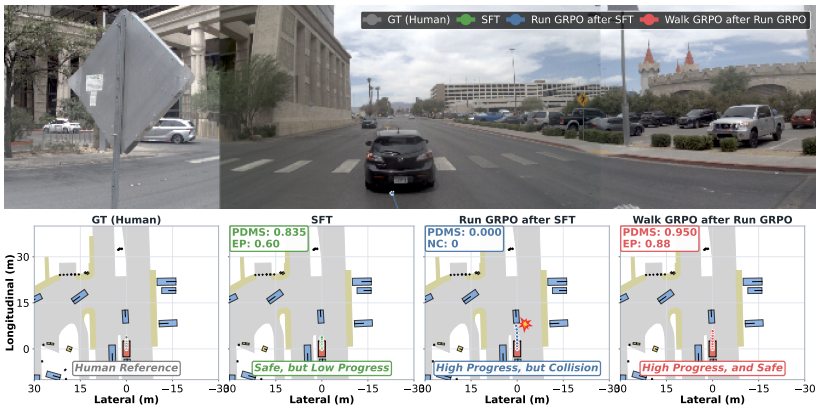
The following four pages present NAVSIM cases for a qualitative comparison. Each triplet shows the human reference (green), the corresponding baseline (orange), and Run-then-Walk (blue) in the same scene. Figures 9 and 10 isolate the loss of driving efficiency from conservative Walk-then-Run training in AutoDrive-P^3 (Ye et al., 2026); Figures 11 and 12 isolate the collision risk of progress-dominant Walk-and-Run training in ReCogDrive (Li et al., 2025c).

Walk-then-Run (AutoDrive-P^3): safe but low progress. In Figures 9 and 10, the baseline trajectories remain safe but end earlier than ours. This conservative behavior reduces driving efficiency despite avoiding unsafe maneuvers, whereas discovering progress before safety refinement yields longer safe trajectories.

Walk-and-Run (ReCogDrive): excessive progress seeking compromises safety. In all six examples in Figures 11 and 12, the baseline pushes forward but collides: NC is 0 and the collision-gated PDMS is 0. Run-then-Walk retains useful forward progress while avoiding the collision (NC 1 in each case).



(a) Straight urban cruising: SFT stops short (EP 0.77); Walk reaches full progress (EP 1.00, PDMS 1.00) while staying centered in lane.

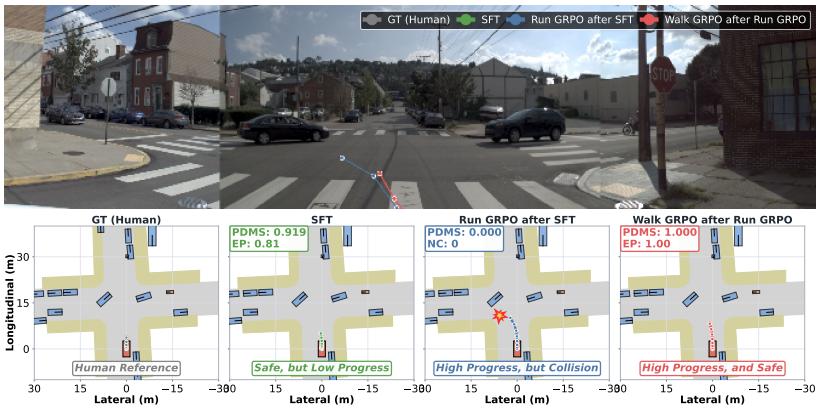


(b) Open road: the Run policy overshoots off-road (PDMS 0); Walk restores a safe path that still advances far (EP $0.60 \rightarrow 0.88$, PDMS $0.835 \rightarrow 0.950$).

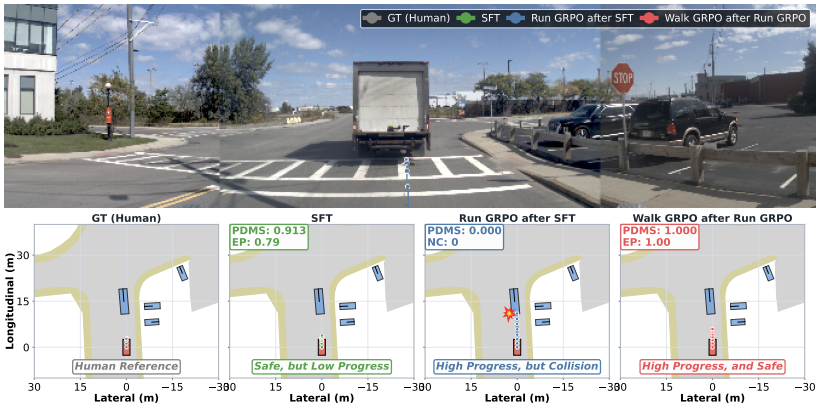


(c) Intersection turn: Walk tracks the human arc more tightly than SFT while extending progress (EP $0.84 \rightarrow 0.97$, PDMS $0.934 \rightarrow 0.989$).

Figure 7: **Stage-by-stage qualitative results (part 1/2).** Human / SFT / Run / Walk trajectories on three NAVSIM scenes. SFT is safe but short, Run is far but unsafe (NC $= 0$), and Walk is both far and safe.



(a) Crosswalk approach: Walk yields safely yet still clears the junction (EP $0.81 \rightarrow 1.00$, PDMS $0.919 \rightarrow 1.00$).



(b) Following a truck: SFT over-brakes behind the lead vehicle; Walk keeps a safe gap while progressing fully (EP $0.79 \rightarrow 1.00$, PDMS $0.913 \rightarrow 1.00$).



(c) Bus interaction: Walk advances past the SFT stopping point without violating safety (EP $0.84 \rightarrow 0.97$, PDMS $0.933 \rightarrow 0.985$).

Figure 8: **Stage-by-stage qualitative results (part 2/2).** Three additional scenes. In every case the Walk-GRPO policy (orange) inherits the long-range progress discovered by Run-GRPO (red) but removes its collisions/off-road events, ending close to or above the human progress level while remaining collision-free.

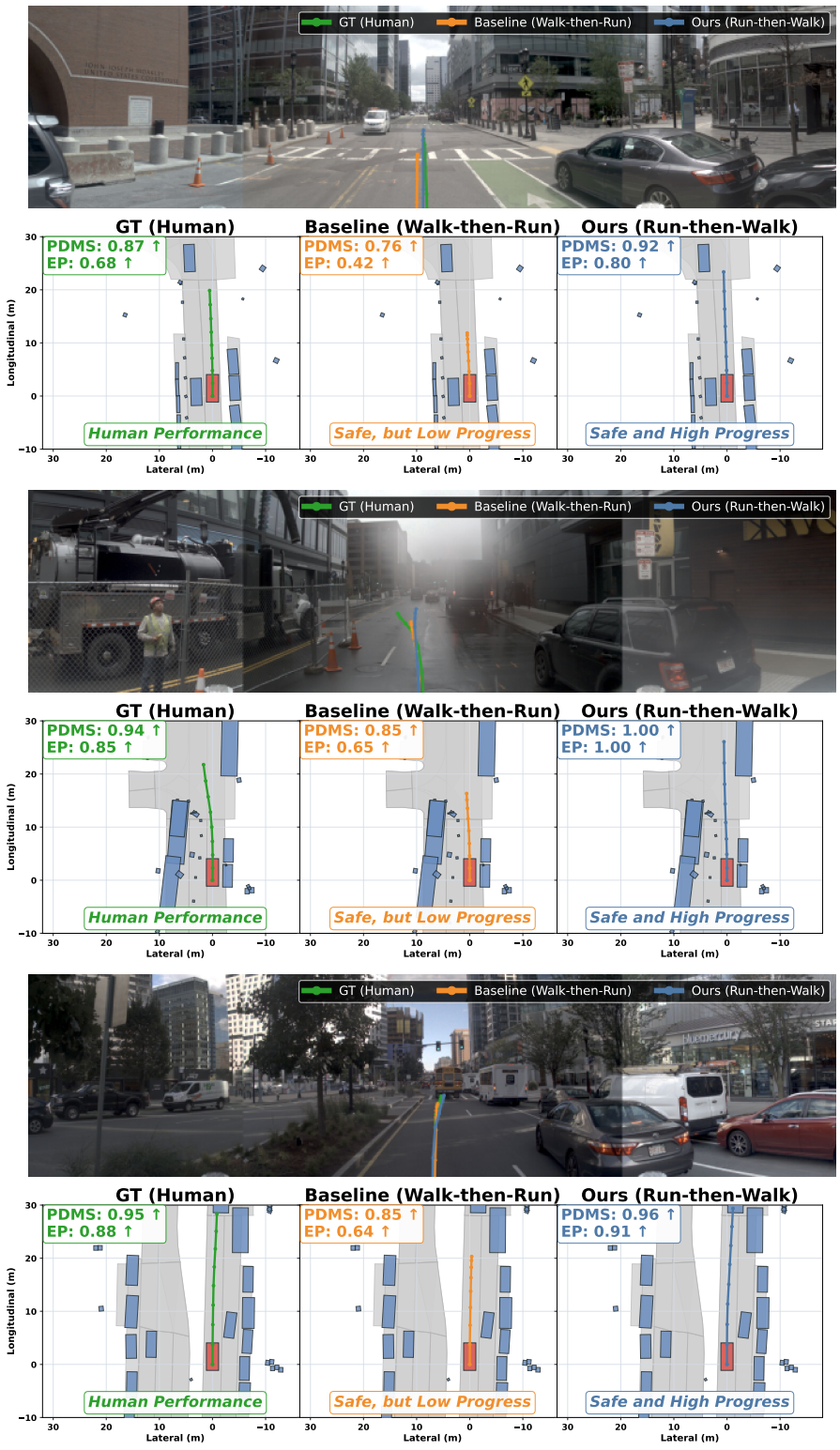


Figure 9: **Qualitative comparison with Walk-then-Run (AutoDrive-P^3) part 1/2.** Across three scenes, the orange safety-first baseline stops short despite remaining safe. Run-then-Walk (blue) travels farther safely and raises both ego progress and PDMS, exposing the driving-efficiency cost of excessive conservatism.

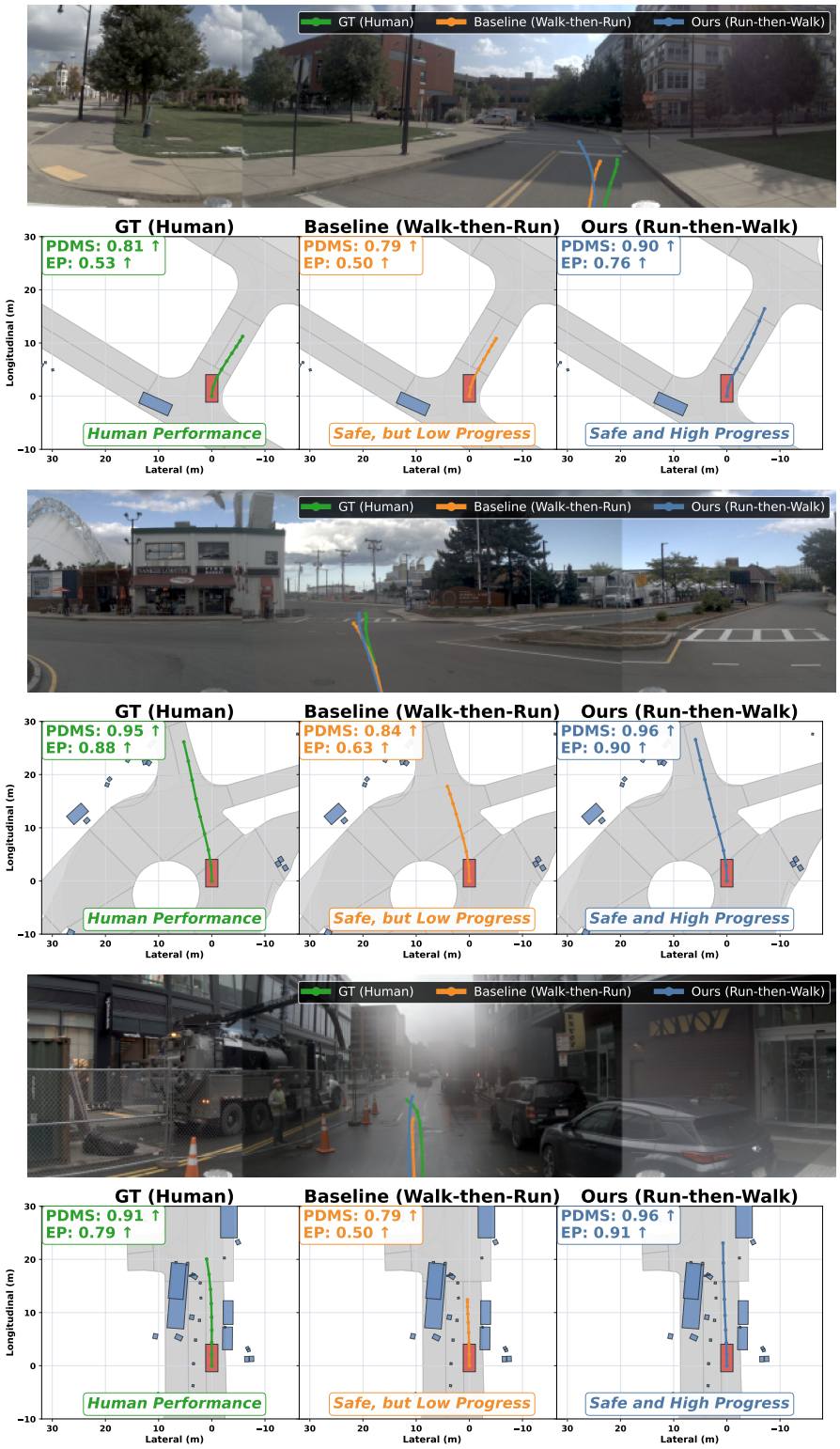


Figure 10: **Qualitative comparison with Walk-then-Run (AutoDrive-P^3) part 2/2.** Three additional scenes confirm that the safety-first baseline remains safe but sacrifices endpoint progress. Run-then-Walk reaches farther safe endpoints and improves driving efficiency.

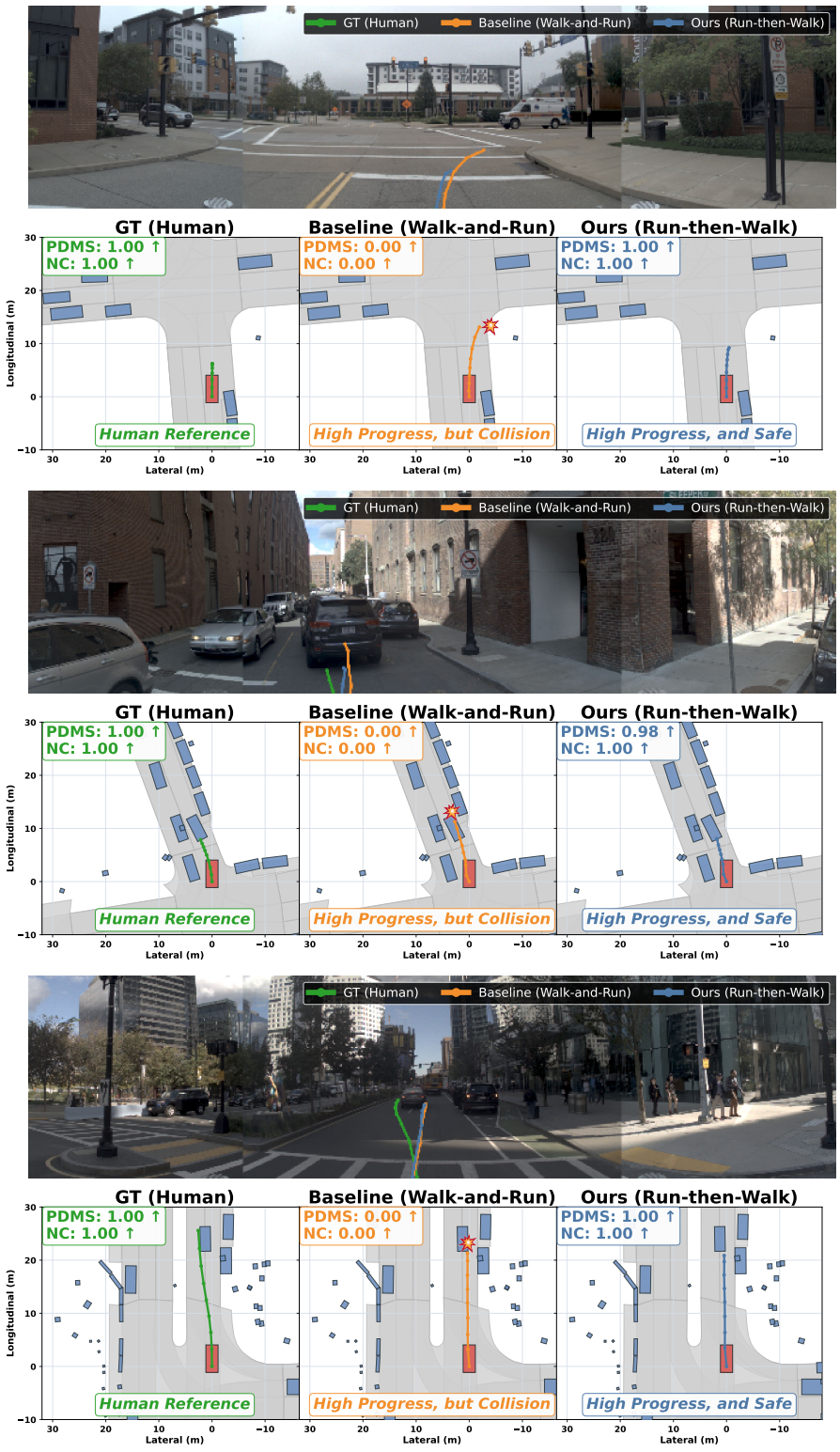


Figure 11: **Qualitative comparison with Walk-and-Run (ReCogDrive), part 1/2.** Three scenes show the cost of overemphasizing progress: the orange baseline advances aggressively but collides in each case. The blue Run-then-Walk trajectories continue forward without collision, avoiding the baseline's safety failure.



Figure 12: **Qualitative comparison with Walk-and-Run (ReCogDrive), part 2/2.** Three additional scenes show the same failure mode: progress-dominant training produces unsafe trajectories, while Run-then-Walk preserves forward motion with NC 1.