# STAR-DRIVE: SELECTIVE TRAJECTORY ADOPTION AND REPLANNING FOR AUTONOMOUS DRIVING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Vision–language–action (VLA) models bring visual understanding and reasoning to autonomous driving, but their high inference cost makes frequent replanning expensive. In per-step replanning (Full Replan), each valid prediction immediately replaces the current driving plan, coupling the decision to request a new prediction with the decision to use it. However, the controller can continue following the current plan between model calls, and a newly generated plan need not be a better replacement. We introduce STAR-DRIVE, a training-free runtime that decides when to request a new plan and whether to adopt it, while leaving the VLA and feedback controller unchanged. Trajectory-guided scheduling uses the active trajectory's age and upcoming motion to schedule model calls, limiting how long the plan can be reused and querying earlier before planned turns or speed changes. After each query, world-aware trajectory selection compares the candidate with the remaining active trajectory over the same future interval, checking for substantially reduced travel distance and vehicle footprints crossing mapped road boundaries. Rejecting a new plan does not extend the current plan's reuse deadline; recovery rules govern execution when normal reuse is unavailable. Across 913 closed-loop AlpaSim scenes with Alpamayo 1.5, STAR-DRIVE increases the driving score, $\text{Score}_{\text{AF}}$, from 59.63 to 65.70 on the $\times 100$ scale and reduces the off-road rate from 7.89% to 4.16% compared with Full Replan. It achieves these gains with 75.53% fewer VLA calls and a $2.81\times$ speedup in mean end-to-end latency, measured after initial rendering and excluding setup.

## 1 INTRODUCTION

Vision–language–action (VLA) models have shown strong performance in autonomous driving by combining visual understanding, language-based reasoning, and action generation (Renz et al., 2025; Zhou et al., 2025; Wang et al., 2025). However, their high inference cost makes frequent replanning expensive. For example, Alpamayo 1.5 predicts a driving trajectory for the next 6.4 seconds (NVIDIA, 2026). In our Full Replan baseline, however, the model is queried again after just 0.1 seconds of simulated time, and each valid prediction immediately replaces the current plan (NVIDIA et al., 2025). This execution rule couples two decisions: when to request a new prediction and whether to adopt it.

Between model calls, the feedback controller can continue tracking the current trajectory while updating steering and acceleration from the current vehicle state. Keeping this trajectory preserves feedback control, although it cannot account for new scene information until the model is queried again. A new prediction, however, need not be a better replacement. Even with identical model inputs, repeated predictions can differ over the trajectory segment used by the controller (Section 4.2). In one recorded driving scene, a new candidate places the predicted vehicle footprint across a mapped road edge, while the predicted vehicle footprints along the current trajectory remain within the mapped lane regions (Figure 3). These observations motivate evaluating a candidate before replacing the plan already being followed.

Prior work offers several ways to reduce the cost of repeated planning. Driving systems such as DriveVLM-Dual and AsyncDriver combine large-model reasoning with a separate real-time planner (Tian et al., 2025; Chen et al., 2024). The large model can run less frequently than the planner, and adaptive driving systems also adjust when to invoke it (Zhang et al., 2025b). For action-chunking



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

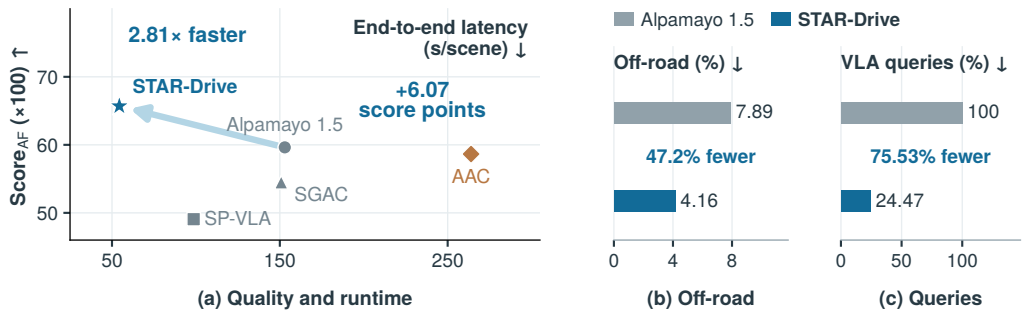



Figure 1: **Maintaining closed-loop driving quality with fewer VLA calls.** Results on 913 closed-loop AlpaSim scenes compared with Alpamayo 1.5 (Full Replan). (a) STAR-DRIVE improves $\text{Score}_{\text{AF}}$ by 6.07 points ($\times 100$ scale) while achieving a $2.81\times$ speedup in mean end-to-end latency. (b) Off-road scene rate decreases by 47.2%. (c) VLA calls decrease by 75.53%.

policies, executing several predicted actions before querying again reduces model calls (Chi et al., 2023). Adaptive action-chunking methods determine how many predicted actions to execute before replanning, using signals such as action entropy or attention (Liang et al., 2026; Wang et al., 2026). SGAC selectively updates action chunks by comparing newly generated actions with those remaining in the action queue (So et al., 2025). This decision is made after generating the candidate and therefore does not avoid that inference call.

For a frozen driving VLA, scheduling the next query depends on which trajectory is actually being followed. Rejecting a candidate does not reset the active trajectory's age, so the next query must account for its age and remaining turns or speed changes. We ask: **When should a driving VLA generate a new plan, and when should that plan replace the one already being followed?**

We introduce STAR-DRIVE, a training-free runtime that coordinates these decisions through the current plan, leaving the VLA and feedback controller unchanged (Figure 2). The active trajectory has two roles: its age and planned motion guide when to query, and its remaining trajectory provides the baseline for evaluating the next prediction. Each adopted normal trajectory is assigned a maximum reuse duration; rejecting a candidate does not restart this duration. Repeated queries therefore cannot indefinitely extend normal reuse of the same trajectory. Recovery rules govern execution when normal reuse is unavailable (Section 3.3).

*Trajectory-guided scheduling* limits the age of the active trajectory and requests an earlier prediction before planned turns or speed changes. These signals come from the stored trajectory and can be evaluated without another model call. After a query, *world-aware trajectory selection* compares the candidate with the remaining active trajectory over the same future interval. It compares their path lengths and checks the predicted vehicle footprints along both trajectories against mapped lane regions and road boundaries. A candidate can be rejected for a substantially shorter path length over the same future interval. It can also be rejected for a road-edge violation when the predicted vehicle footprints along the remaining active trajectory stay within the mapped lane regions. Selection, retention, and recovery determine the reference trajectory that the existing controller tracks from the current vehicle state.

We evaluate STAR-DRIVE on 913 closed-loop AlpaSim scenes with Alpamayo 1.5. Compared with Full Replan, it raises the driving score, $\text{Score}_{\text{AF}}$, from 59.63 to 65.70 ($\times 100$ scale) and reduces the off-road rate from 7.89% to 4.16% (Figure 1). It achieves these gains with 75.53% fewer VLA calls and a $2.81\times$ speedup in mean end-to-end latency, measured after initial rendering and excluding setup (Section 4.1). On the same timing metric, STAR-DRIVE is $1.8$–$4.9\times$ faster than the evaluated driving adaptations of SP-VLA, SGAC, and AAC.

Component ablations on 100 development scenes evaluate both runtime decisions. With candidate checks enabled, sparse querying raises $\text{Score}_{\text{AF}}$ from 63.31 to 72.06. Under the same query rule, candidate checks and their associated recovery triggers raise the score from 67.58 to 72.06 and reduce off-road episodes from five to two (Section 4.3).

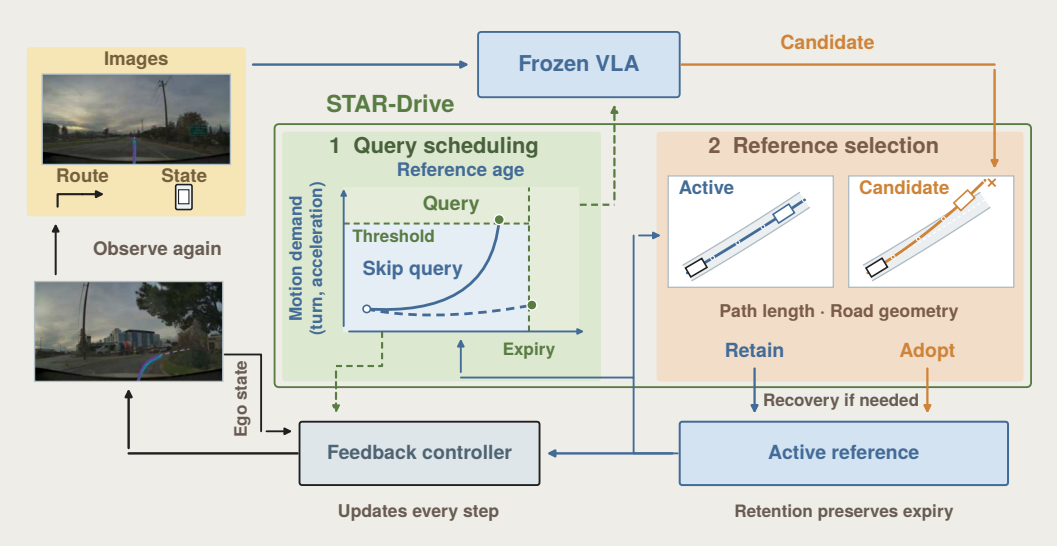



Figure 2: **STAR-Drive separates VLA querying from trajectory adoption.** (1) Query scheduling uses the active trajectory's age and upcoming motion to decide when to call the VLA; blue denotes skipped queries. (2) Reference selection compares the candidate with the remaining active trajectory using path length and road geometry. Retaining the active reference leaves its reuse deadline unchanged. The feedback controller updates commands from the current ego state at every control step.

Our contributions are:

- **Coordinated querying and plan adoption.** We introduce a training-free runtime in which the current trajectory guides both query timing and candidate selection, preserving the adopted trajectory's original reuse deadline whenever it is retained.
- **Trajectory-guided scheduling and world-aware trajectory selection.** We schedule queries from trajectory age and planned motion, compare candidates with the remaining active trajectory using path length and map-based geometric checks, and define recovery when normal reuse is unavailable.
- **Closed-loop quality and efficiency gains.** On 913 AlpaSim scenes, we improve driving score and off-road rate while reducing VLA calls and measured end-to-end latency. Component ablations quantify the score gains from sparse querying and candidate checks with their associated recovery rules.

## 2 Related Work

**VLA Models for Autonomous Driving.** Language-based driving systems address scene understanding, instruction following, and planning (Sima et al., 2024; Shao et al., 2024; Ma et al., 2024; Mao et al., 2024). Driving VLAs connect visual and language understanding with action prediction (Renz et al., 2025; Xu et al., 2025; Hwang et al., 2025; Zhou et al., 2025). Inference cost motivates dual-system planning (Tian et al., 2025; Chen et al., 2024), adaptive invocation (Zhang et al., 2025b; Tang et al., 2025), and predictive feature reuse (Hamdan et al., 2025). Related hybrid planners allocate computation between rule-based and learned planning (Fan et al., 2025). Alpamayo-R1 (Wang et al., 2025) combines chain-of-causation reasoning and flow-matching trajectory prediction in one model. Because each output spans multiple control steps, its inference cost can also be amortized by continuing to track the predicted trajectory. This shifts attention from generating trajectories to managing their execution: STAR-Drive decides when to query again and whether to replace the active trajectory, with the model and controller unchanged.

**Efficient VLA Inference.** VLA efficiency depends on both prediction cost and how predictions are used. Distillation and parallel decoding reduce prediction cost (Prasad et al., 2024; Kim et al., 2025), while FAST compresses action sequences into fewer tokens (Pertsch et al., 2025); model scheduling and token pruning further reduce inference workload (Li et al., 2026). Across control steps, action chunks support temporal ensembling and receding-horizon execution (Zhao et al., 2023; Chi et al., 2023), while asynchronous generation overlaps inference with execution (Black et al., 2025). The resulting responsiveness–consistency trade-off motivates adaptive query timing based on entropy, attention, plan likelihood, or learned policies (Liang et al., 2026; Wang et al., 2026; Zhou et al., 2023; Honda et al., 2024). This timing problem also connects to event- and self-triggered control (Heemels et al., 2012; Li & Shi, 2014) and resource-aware MPC (Gommans & Heemels, 2015; Brunner et al., 2016). Timing alone, however, does not determine whether a new prediction should replace the current plan. Selective action-chunk updates, candidate selection, and methods that incorporate planning history address continuity across successive predictions (So et al., 2025; Liu et al., 2025; Song et al., 2025; Zhang et al., 2025a), while predictive safety filters assess proposed controls against model-based constraints (Wabersich & Zeilinger, 2021). STAR-DRIVE brings these two runtime decisions together: sparse queries produce candidates, which are compared with the active trajectory from the current vehicle state using time-aligned path-length comparisons and road-geometry checks. The active trajectory thus links inference scheduling to candidate evaluation around a frozen VLA.

# 3 METHOD

STAR-DRIVE coordinates querying and adoption through the active trajectory: its age and planned motion guide the next query, and its remaining path provides the baseline for candidate selection. The feedback controller updates at every control step.

## 3.1 PROBLEM FORMULATION

We consider a frozen trajectory-generating VLA $\pi_\theta$ and a fixed feedback controller $\mathcal{C}$. At control step $t$, corresponding to a driver callback in AlpaSim, the simulation time is $T_t$. When the query indicator $q_t = 1$, the model maps observation and navigation inputs $X_t$ to candidate $\widehat{P}_t$. At every control step, the controller maps ego state $s_t$ and reference $P_t^{\mathrm{ref}}$ to command $u_t$:

$$\widehat{P}_t = \pi_\theta(X_t; \xi_t) \quad \text{when } q_t = 1, \qquad u_t = \mathcal{C}(s_t, P_t^{\mathrm{ref}}), \tag{1}$$

where $\xi_t$ denotes generation randomness. Reference reuse preserves current-state feedback: steering and acceleration continue to update.

Full Replan adopts each valid prediction immediately. STAR-DRIVE instead selects among retaining the active reference, adopting the candidate, and entering recovery.

Let $A_t^-$ store the active trajectory and its metadata immediately before control step $t$. The metadata include the original prediction pose and timestamps, the adoption step $\tau$, and the reuse deadline $e$. The *query rule* schedules inference from this record; the *selection rule* updates it after candidate evaluation. Queries complete synchronously before reference selection and control.

## 3.2 TRAJECTORY-GUIDED SCHEDULING

The active reference provides two scheduling signals before inference: its age measures how long the current plan has been used, and its upcoming motion identifies planned changes in heading or speed. Reading these stored quantities avoids a model call to decide whether to make one. The age limit bounds reuse even along a straight, constant-speed path; motion triggers allow an earlier query near a planned maneuver.

A normal (non-braking) reference adopted at control step $\tau$ has age $a_t = t - \tau$ and expiry $e = \tau + H$, where $H$ bounds ordinary retention in control steps ($t < e$). Let $m_t = 1$ when either the maximum absolute heading change or the maximum absolute acceleration within the lookahead exceeds its threshold. For an active normal reference,

$$q_t = \mathbb{I}[a_t \geq H \ \vee \ m_t = 1]. \tag{2}$$

The lookahead advances with reference age to examine the upcoming portion of the stored trajectory. Appendix A.1 defines the heading and acceleration statistics. The runtime queries whenever the

active record is empty and at every control step during braking recovery. Otherwise, if neither trigger fires, the runtime skips the VLA call and the controller tracks the time-aligned active reference.

Only adoption of a new normal trajectory starts a new reuse period; querying or retaining a reference does not reset its expiry. This prevents repeated candidate rejections from renewing the same normal reference. If trajectory comparison is unavailable at expiry, the recovery rules allow retention for that control step only. The evaluated reuse duration and thresholds are specified in Section 4.1.

### 3.3 World-Aware Trajectory Selection

Selection compares the candidate with the remaining active trajectory using path length and road geometry, subject to reference expiry and recovery.

**Time alignment.** A trajectory is usable for comparison when its shape, finite values, timestamps, and future coverage pass validation and it can be expressed at the comparison times in the current vehicle frame. We interpolate the stored trajectory on its original time axis and transform it from its prediction pose to the current frame:

$$P_t^- = \mathrm{Align}(A_t^-, s_t, \{T_t + \delta_k\}_k), \tag{3}$$

Here $\delta_k$ are the candidate's future time offsets. Both trajectories are evaluated at the same future timestamps in the current ego-vehicle coordinate frame. Retaining the original timestamps prevents replaying elapsed motion (Appendix A.2).

**Relative path length.** The time-aligned active path represents the motion the controller would continue tracking if the candidate were rejected. It therefore provides a local baseline for detecting a large reduction in planned travel. Normalizing by its length scales the threshold to the current plan's motion. With $r_0(P)$ the current vehicle position and $r_j(P)$ aligned future positions, we compute over $N_\rho$ segments

$$L(P) = \sum_{j=1}^{N_\rho} \|r_j(P) - r_{j-1}(P)\|_2, \qquad \rho_t = \frac{L(\widehat{P}_t)}{L(P_t^-)}. \tag{4}$$

A candidate triggers path-length rejection when its path length is less than $\gamma_\rho$ times that of the remaining active trajectory over the same future interval. A usable, unexpired active trajectory is then retained, subject to braking priority. For the ratio, distances below $\epsilon_s = 10^{-3}$ m are treated as stationary: two stationary paths give $\rho_t = 1$, while a stationary active path paired with a moving candidate proceeds directly to the map check (Appendix A.3). Component and threshold controls appear in Section 4.3.

**Road-geometry checks.** We check the predicted rectangular vehicle footprints along each trajectory against mapped lane regions and road edges. The assessment $e_{\mathrm{lane}}(P)$ is OnRoad when all sampled footprints are covered by lane regions, and OffRoad when a sampled footprint extends beyond lane coverage and satisfies the road-edge violation test in Appendix A.3. Map-based retention requires the candidate to satisfy the road-edge violation test and all sampled footprints of the remaining active trajectory to lie within the mapped lane regions. Subject to braking priority, a usable normal reference is retained when:

$$\left.\begin{aligned} e_{\mathrm{lane}}(\widehat{P}_t) &= \mathrm{OFFROAD}, \\ e_{\mathrm{lane}}(P_t^-) &= \mathrm{ONROAD}, \qquad t < e \end{aligned}\right\} \implies \mathrm{RETAIN}. \tag{5}$$

The checking horizon spans the longer of the normal reuse duration and estimated stopping time:

$$N_t^{\mathrm{lane}} = \max\left(H, \left\lceil \frac{\hat{v}_t}{b\Delta_c} \right\rceil\right), \tag{6}$$

where $\hat{v}_t$ is estimated ego speed, $b$ is nominal braking deceleration, and $\Delta_c$ is the control-step interval. An uncovered footprint without a detected road-edge violation is PossibleExit; missing or unusable geometry is Unavailable. These two states leave the map rejection condition inactive. Path-length-based retention does not assign OnRoad status. Appendix A.3 specifies sampling and geometric tolerances.

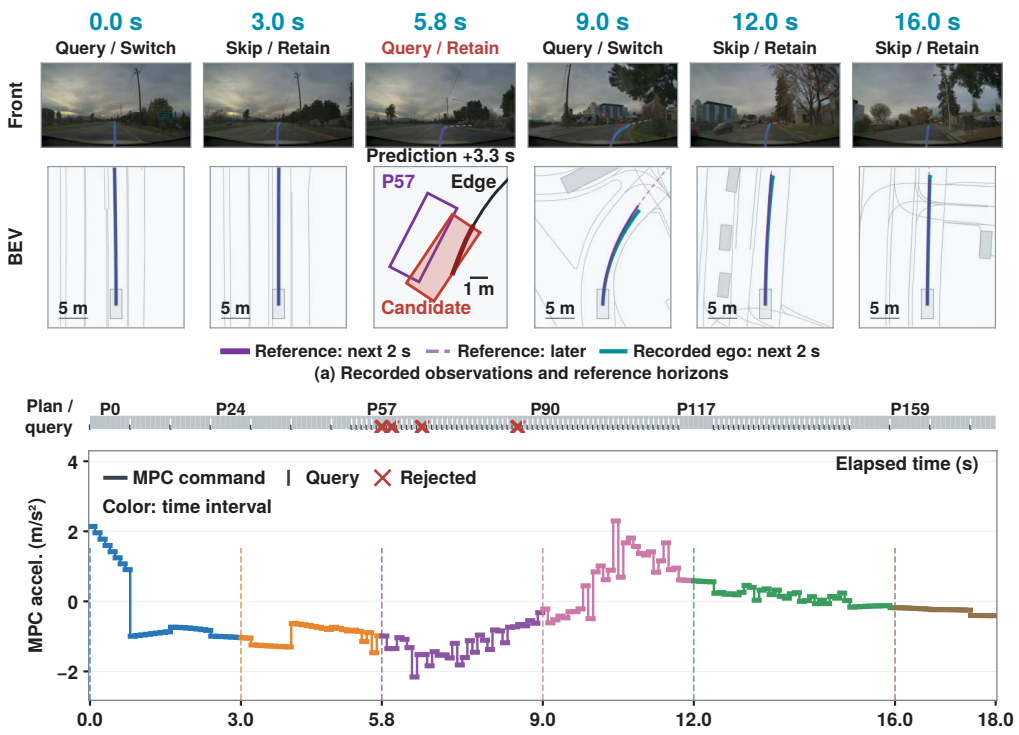



Figure 3: **Rejecting an off-road candidate while continuing closed-loop control.** (a) Front-camera and bird's-eye views. At 5.8 s, the inset compares vehicle footprints predicted 3.3 s ahead: the candidate crosses the road edge, whereas reference P57 stays inside. STAR-DRIVE therefore retains P57. Purple denotes the reference (solid: next 2 s; dashed: later), and cyan denotes recorded ego motion (next 2 s). Below: MPC acceleration during this 18-s illustrative rollout shows control updates between VLA calls. Ticks mark queries, red crosses mark rejections, and colors indicate time intervals.

**Braking recovery and continuation.** If no braking reference is eligible for retention, braking is triggered by either rejection condition at the reuse deadline of the normal trajectory, or by an off-road candidate without an active trajectory whose sampled footprints are fully covered by mapped lane regions. Recovery reuses the active path's geometry and retimes its arc length to bring the vehicle to a stop. Adopting the braking trajectory requires a full-horizon road assessment under the trigger-specific criteria in Appendix A.4. A braking reference has reuse duration $H_R$ and expires at $e = t + H_R$; the VLA is queried at every control step during braking.

Four construction failures permit one-step execution of a format-valid candidate: an unavailable checking horizon or one exceeding the braking trajectory's duration or reuse limit; missing lane-coverage information required by the triggering condition; a path shorter than the stopping distance; or missing active-trajectory geometry or lane information. An off-road candidate without an active trajectory from which to construct a braking reference follows the same rule. This branch applies no reuse checks, clears the active record, and forces a query at the next control step. Other unresolved recovery cases return FALLBACKREQUIRED with no executable reference. Appendix A.4 details braking construction and one-step candidate execution.

## 4 EXPERIMENTS

We evaluate driving quality and computation under different query schedules and trajectory-adoption rules. Adapted inference-time baselines distinguish query frequency from work per query; component ablations and matched-state diagnostics examine selection and feedback during reuse.

Table 1: **Driving quality with a frozen VLA on 913 scenes.** STAR-DRIVE attains the highest mean $\text{Score}_{\text{AF}}$ and lowest off-road rate among the evaluated configurations. SP-VLA, SGAC, and AAC denote the adaptations specified in the setup. Best values are bold.

| Method | Scene Score$_{\text{AF}}$ ↑ ($\times 100$) | Collision$_{\text{AF}}$ ↓ (%) | OffRoad↓ (%) | $P_{\text{rel}}$ ↑ | $D_{\text{traj}}$ ↓ (m) | Plan Dev.↓ (m) |
|---|---|---|---|---|---|---|
| Alpamayo 1.5 | 59.63 | 1.75 | 7.89 | 0.842 | 1.53 | 0.348 |
| SP-VLA | 49.07 | 1.97 | 6.57 | 0.757 | 1.09 | 0.811 |
| SGAC | 54.49 | 2.85 | 6.13 | 0.827 | **1.05** | **0.005** |
| AAC | 58.65 | 1.42 | 6.57 | 0.851 | 1.57 | 0.100 |
| **STAR-DRIVE** | **65.70** | **1.10** | **4.16** | **0.882** | 1.30 | 0.105 |

### 4.1 EXPERIMENTAL SETUP

**Benchmark.** We evaluate closed-loop driving in AlpaSim on the `public_2601` scene suite (NVIDIA et al., 2025). The evaluation includes 913 scenes, each with 84 driver callbacks and 101 simulator steps, yielding 76,692 callbacks per method. The approximately 10-s simulation includes 1.7 s of ground-truth initialization. Control updates occur every 0.1 s in simulation time; inference wall-clock time is measured separately and is not imposed as an execution delay in the simulator.

**Base model.** We use Alpamayo 1.5 10B (NVIDIA, 2026) as the trajectory generator, with the unchanged feedback controller updating commands between queries.

**Baselines.** We compare with per-callback replanning and three inference-time mechanisms adapted to the Alpamayo–AlpaSim interface. **Alpamayo 1.5** is the FULL REPLAN baseline: it invokes the policy and installs a new reference at every driver callback. **SP-VLA**: our adaptation transfers its scheduling and ridge-generation mechanism (Li et al., 2026) to supply trajectories when Alpamayo is skipped. We evaluate **SGAC** (So et al., 2025) and **AAC** (Liang et al., 2026) using their official implementations. All methods use the same 913 scenes.

**Implementation details.** The reported evaluation uses the trajectory-guided schedule. Native predictions contain $K = 64$ waypoints with $\Delta_w = \Delta_c = 0.1$ s. We set the age limit for ordinary retention to $H = 8$ callbacks (0.8 s). We set the motion-trigger lookahead to $N = 20$ and compare $N_\rho = 20$ future segments in the path-length check; early queries use heading and acceleration thresholds of $5^\circ$ and $2.5\,\text{m/s}^2$, and the path-length ratio threshold is $\gamma_\rho = 0.5$. Recovery uses nominal deceleration $b = 4\ \text{m/s}^2$ and per-reference lifetime $H_R = 40$ callbacks. Road geometry comes from the AlpaSim map interface; numerical geometry tolerances are in Appendix A.3. The evaluated recovery configuration is specified in Appendix A.1.

**Metrics.** Our primary driving metrics are $\text{Score}_{\text{AF}}$, at-fault collision, and off-road rate. We also report relative progress, distance to the ground-truth trajectory, and deviation between consecutive plans. Efficiency is measured by full VLA invocation rate, Driver latency, and end-to-end latency. Invocation rate uses driver callbacks as the denominator. Driver latency averages wall-clock policy-driver execution time over all callbacks, including VLA queries and reference reuse. End-to-end latency measures the wall-clock interval from the first policy event after initial rendering to simulation-loop timer stop, in seconds per scene. It includes subsequent rendering, simulation, and first-call model warmup, and excludes setup and the initial render. Speedup divides the FULL REPLAN mean end-to-end latency by each method's mean. Driving metrics are aggregated over 913 scenes, except plan deviation, which has 903 valid records per method without imputation. Appendix Table 4 gives the metric definitions and aggregation rules.

### 4.2 DRIVING QUALITY AND MEASURED RUNTIME

**Improved driving with fewer queries.** With the same frozen model and controller, STAR-DRIVE improves $\text{Score}_{\text{AF}}$ from 59.63 to 65.70 and reduces off-road rate from 7.89% to 4.16% while using

Table 2: **STAR-DRIVE reduces both VLA calls and measured runtime.** Across 913 scenes and 76,692 callbacks per method, STAR-DRIVE uses 24.47% of Full Replan's VLA calls and achieves a 2.81× end-to-end speedup. Best values are bold.

| Method | VLA calls↓ (%) | Driver↓ (ms/callback) | End-to-end latency↓ (s/scene) | Speedup↑ |
|---|---|---|---|---|
| Alpamayo 1.5 | 100.00 | 1591.33 | 152.95 | 1.00× |
| SP-VLA | 57.14 | 929.35 | 98.67 | 1.55× |
| SGAC | 100.00 | 1553.83 | 150.82 | 1.01× |
| AAC | 25.18 | 2902.34 | 263.94 | 0.58× |
| **STAR-DRIVE** | **24.47** | **422.71** | **54.37** | **2.81×** |

24.47% of Full Replan's VLA calls (Tables 1 and 2). The off-road reduction corresponds to 72 versus 38 affected scenes, a 47.2% relative decrease, and accompanies increased relative progress from 0.842 to 0.882. At-fault collisions fall from 16 to ten scenes (1.75% to 1.10%); Wrong Lane incidence increases from 16.54% to 19.82%.

**Comparison with inference-time baselines.** SGAC achieves the lowest trajectory distance and plan deviation, whereas STAR-DRIVE achieves the highest $\text{Score}_{\text{AF}}$ and lowest off-road rate (Table 1). SP-VLA reduces calls through scheduling and substitute predictions but scores 49.07 versus 65.70 for STAR-DRIVE. AAC has a similar invocation rate (25.18% versus 24.47%) but a lower score (58.65 versus 65.70).

**Call reduction and realized runtime savings.** STAR-DRIVE executes 18,768 VLA queries over 76,692 callbacks, removing 75.53% of calls while retaining feedback updates at every callback. Mean Driver latency falls from 1591.33 to 422.71 ms/callback, a 3.76× reduction. Mean end-to-end latency decreases from 152.95 to 54.37 s per scene, a 2.81× speedup over the common interval that also includes subsequent rendering and simulation (Table 2).

Runtime depends on the cost of obtaining the scheduling signal as well as on query frequency. Similar call rates conceal different inference workloads: STAR-DRIVE decides from the existing reference, whereas the evaluated AAC implementation generates 20 sequential expert samples to choose an execution length. This adaptation has a mean end-to-end latency of 263.94 s per scene, compared with 54.37 s for STAR-DRIVE. SGAC queries at every callback and has end-to-end latency close to per-callback replanning; SP-VLA reduces end-to-end latency to 98.67 s.

**Trajectory prediction variability.** At 249 fixed-conditioning contexts from ten auxiliary scenes, we sample 20 trajectories per context. Pairwise XY average displacement error over the controller's 1–2 s tracking window has pooled median 0.300 m and 90th percentile 0.939 m.

**Local command continuity during reuse.** We compare STAR-DRIVE and Full Replan at the first eligible control step in each scene where their vehicle states and previous commands match. One run retains its trajectory and the other queries again, yielding 860 diagnostic pairs. STAR-DRIVE produces a smaller longitudinal command change in 851 pairs (99.0%) and a smaller steering change in 780 pairs (90.7%). These smaller immediate command changes support local command continuity during reuse (Appendix C.2).

### 4.3 ABLATION STUDIES

**Component ablations.** Table 3 separates sparse querying from candidate checks on 100 development scenes, alongside the native Alpamayo 1.5 baseline on the same scenes. With checks enabled, sparse querying raises score from 63.31 to 72.06 while using 23.69% of the calls. With sparse querying enabled, candidate checks and their associated recovery triggers raise score from 67.58 to 72.06 and reduce off-road episodes from five to two. Enabling early queries with the other runtime rules unchanged raises $\text{Score}_{\text{AF}}$ from 69.17 to 72.06, while increasing the query rate from 13.61% to 23.69%. The no-early-query variant has zero at-fault collisions and off-road episodes, versus two of each for the complete runtime (Table 5).

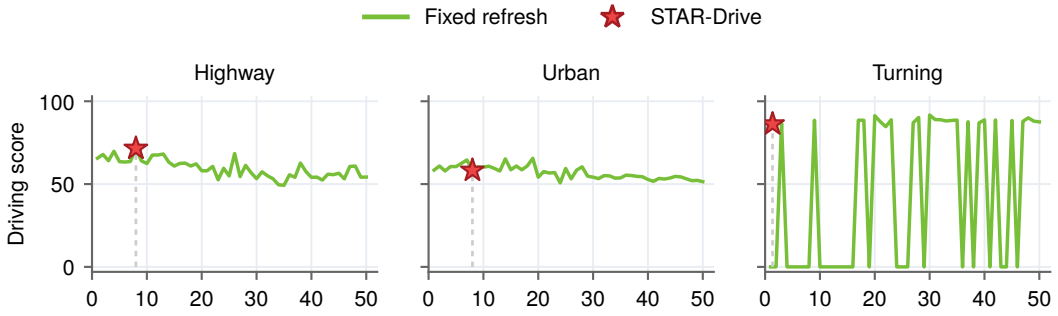



Figure 4: **Fixed-interval replanning is especially sensitive to query interval in the turning scene.** Green curves: driving score ($\times 100$) versus fixed replanning interval $H = 1, \ldots, 50$ callbacks. Red stars: STAR-DRIVE at its mean realized query interval.

Table 3: **Matched baseline and ablations on 100 scenes.** Native Alpamayo 1.5 queries every callback without STAR-DRIVE selection or recovery. For STAR-DRIVE, disabling sparse queries means every-callback inference; disabling candidate checks removes path-length and road-geometry checks plus their recovery triggers. Structural validation and remaining recovery rules stay active.

| Sparse queries | Candidate checks | $\text{Score}_{\text{AF}} \uparrow$ ($\times 100$) | $\text{Collision}_{\text{AF}} \downarrow$ (%) | OffRoad $\downarrow$ (%) | Queries $\downarrow$ (%) |
|---|---|---|---|---|---|
| Alpamayo 1.5 (native) | | 61.33 | 3 | 6 | 100.00 |
| × | ✓ | 63.31 | 3 | 2 | 100.00 |
| ✓ | × | 67.58 | 2 | 5 | 22.46 |
| **STAR-Drive** | | 72.06 | 2 | 2 | 23.69 |

**Individual components and parameters.** On the same 100-scene panel, disabling path-length or road-geometry checks and their associated recovery triggers gives scores of 68.48 and 67.97; replacing braking with one-step candidate execution gives 67.25 (Table 5). Setting $H = 3$ gives $\text{Score}_{\text{AF}}$ 64.33 at 42.38% queries versus 72.06 at 23.69% for $H = 8$; $\gamma_\rho = 0.25$ and 0.75 give scores 68.87 and 69.17, respectively.

### 4.4 EXECUTION BEHAVIOR ACROSS QUERY INTERVALS

On the 100-scene development panel, the complete runtime scores 72.06 with 23.69% queries, compared with 66.15 / 25.00% and 65.23 / 20.24% for nearby periodic execution configurations. These controls retain candidate checks and use local braking between scheduled queries.

**Beyond query frequency: controlling plan execution.** Figure 4 compares STAR-DRIVE with fixed-interval replanning without selection or recovery. STAR-DRIVE matches the highway sweep's peak score of 71.54 at a mean query gap of eight callbacks. On the turning scene, fixed replanning scores zero at both $H = 1$ and $H = 8$, versus 86.24 for STAR-DRIVE, showing that frequent queries alone are insufficient. Urban and turning sweep maxima are 65.55 and 91.70, versus 58.26 and 86.24 for STAR-DRIVE; Section 4.3 isolates component contributions.

## 5 CONCLUSION

STAR-DRIVE uses a persistent trajectory reference to connect sparse VLA querying, candidate comparison, and feedback tracking. With the model and controller unchanged, it improves driving score and off-road rate across 913 AlpaSim scenes, removes 75.53% of VLA calls, and achieves a $2.81\times$ measured end-to-end speedup. These results demonstrate that the execution of existing predictions is a practical source of quality and efficiency gains for driving VLAs.

### AI USE STATEMENT

We used generative AI tools to assist with manuscript preparation, including drafting, language polishing, and figure preparation. The authors take responsibility for the final content of this paper.

## REFERENCES


Kevin Black, Manuel Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. In *Advances in Neural Information Processing Systems*, volume 38, pp. 33383–33407, 2025. doi: 10.52202/085713-1122. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/300ccb2187dedd4edcc07f7e76d8e553-Abstract-Conference.html.

Florian David Brunner, Maurice Heemels, and Frank Allgöwer. Robust self-triggered MPC for constrained linear systems: A tube-based approach. *Automatica*, 72:73–83, 2016. doi: 10.1016/j.automatica.2016.05.004.

Yuan Chen, Zi-han Ding, Ziqin Wang, Yan Wang, Lijun Zhang, and Si Liu. Asynchronous large language model enhanced planner for autonomous driving. In *Computer Vision – ECCV 2024*, volume 15094 of *Lecture Notes in Computer Science*, pp. 22–38, 2024. doi: 10.1007/978-3-031-72764-1_2. URL https://www.ecva.net/papers/eccv_2024/papers_ECCV/html/5239_ECCV_2024_paper.php.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin C. M. Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems XIX*, 2023. doi: 10.15607/RSS.2023.XIX.026. URL https://roboticsproceedings.org/rss19/p026.html.

Yuqi Fan, Zhiyong Cui, Zhenning Li, Yilong Ren, and Haiyang Yu. SAH-drive: A scenario-aware hybrid planner for closed-loop vehicle trajectory generation. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 15719–15735, 2025. URL https://proceedings.mlr.press/v267/fan25c.html.

T. M. P. Gommans and W. P. M. H. Heemels. Resource-aware MPC for constrained nonlinear systems: A self-triggered control approach. *Systems & Control Letters*, 79:59–67, 2015. doi: 10.1016/j.sysconle.2015.03.003.

Shadi Hamdan, Chonghao Sima, Zetong Yang, Hongyang Li, and Fatma Guney. ETA: Efficiency through thinking ahead, a dual approach to self-driving with large models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 26529–26538, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Hamdan_ETA_Efficiency_through_Thinking_Ahead_A_Dual_Approach_to_Self-Driving_ICCV_2025_paper.html.

W. P. M. H. Heemels, K. H. Johansson, and P. Tabuada. An introduction to event-triggered and self-triggered control. In *51st IEEE Conference on Decision and Control*, pp. 3270–3285, 2012. doi: 10.1109/CDC.2012.6425820. URL https://doi.org/10.1109/CDC.2012.6425820.

Kohei Honda, Ryo Yonetani, Mai Nishimura, and Tadashi Kozuno. When to replan? an adaptive replanning strategy for autonomous navigation using deep reinforcement learning. In *2024 IEEE International Conference on Robotics and Automation*, pp. 6650–6656, 2024. doi: 10.1109/ICRA57147.2024.10611474. URL https://arxiv.org/abs/2304.12046.

Jyh-Jing Hwang, Runsheng Xu, Hubert Lin, Wei-Chih Hung, Jingwei Ji, Kristy Choi, Di Huang, Tong He, Paul Covington, Benjamin Sapp, Yin Zhou, James Guo, Dragomir Anguelov, and Mingxing Tan. EMMA: End-to-end multimodal model for autonomous driving. *Transactions on Machine Learning Research*, 2025. URL https://openreview.net/forum?id=kH3t5lmOU8.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. In *Robotics: Science and Systems XXI*, 2025. doi: 10.15607/RSS.2025.XXI.017. URL https://roboticsproceedings.org/rss21/p017.html.

Huiping Li and Yang Shi. Event-triggered robust model predictive control of continuous-time nonlinear systems. *Automatica*, 50(5):1507–1513, 2014. doi: 10.1016/j.automatica.2014.03.015.

Ye Li, Yuan Meng, Zewen Sun, Kangye Ji, Chen Tang, Jiajun Fan, Xinzhu Ma, Shu-Tao Xia, Zhi Wang, and Wenwu Zhu. SP-VLA: A joint model scheduling and token pruning approach for VLA model acceleration. In *International Conference on Learning Representations*, pp. 38155–38184, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/4072543747a14bbed76284cf2c04b9e9-Abstract-Conference.html.

Yuanchang Liang, Xiaobo Wang, Kai Wang, Shuo Wang, Xiaojiang Peng, Haoyu Chen, David Kim Huat Chua, and Prahlad Vadakkepat. Adaptive action chunking at inference-time for vision-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20802–20811, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Liang_Adaptive_Action_Chunking_at_Inference-time_for_Vision-Language-Action_Models_CVPR_2026_paper.html.

Yuejiang Liu, Jubayer Hamid, Annie Xie, Yoonho Lee, Max Du, and Chelsea Finn. Bidirectional decoding: Improving action chunking via guided test-time sampling. In *International Conference on Learning Representations*, pp. 4594–4627, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/0d78dd998f7b9ac79604d47a2d79bb0d-Abstract-Conference.html.

Yingzi Ma, Yulong Cao, Jiachen Sun, Marco Pavone, and Chaowei Xiao. Dolphins: Multimodal language model for driving. In *European Conference on Computer Vision (ECCV)*, pp. 403–420, 2024. doi: 10.1007/978-3-031-72995-9_23. URL https://www.ecva.net/papers/eccv_2024/papers_ECCV/papers/06203.pdf.

Jiageng Mao, Junjie Ye, Yuxi Qian, Marco Pavone, and Yue Wang. A language agent for autonomous driving. In *Conference on Language Modeling (COLM)*, 2024. URL https://openreview.net/forum?id=UPE6WYE8vg.

NVIDIA. Alpamayo 1.5 10B model card. Hugging Face model release, 2026. URL https://huggingface.co/nvidia/Alpamayo-1.5-10B.

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

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. FAST: Efficient action tokenization for vision-language-action models. In *Robotics: Science and Systems XXI*, 2025. doi: 10.15607/RSS.2025.XXI.012. URL https://www.roboticsproceedings.org/rss21/p012.html.

Aaditya Prasad, Kevin Lin, Jimmy Wu, Linqi Zhou, and Jeannette Bohg. Consistency policy: Accelerated visuomotor policies via consistency distillation. In *Robotics: Science and Systems XX*, 2024. doi: 10.15607/RSS.2024.XX.071. URL https://roboticsproceedings.org/rss20/p071.html.

Katrin Renz, Long Chen, Elahe Arani, and Oleg Sinavski. SimLingo: Vision-only closed-loop autonomous driving with language-action alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11993–12003, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Renz_SimLingo_Vision-Only_Closed-Loop_Autonomous_Driving_with_Language-Action_Alignment_CVPR_2025_paper.html.

Hao Shao, Yuxuan Hu, Letian Wang, Guanglu Song, Steven L. Waslander, Yu Liu, and Hongsheng Li. LMDrive: Closed-loop end-to-end driving with large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15120–15130, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Shao_LMDrive_Closed-Loop_End-to-End_Driving_with_Large_Language_Models_CVPR_2024_paper.html.

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 (ECCV)*, pp. 256–274, 2024. doi: 10.1007/978-3-031-72943-0_15. URL https://www.ecva.net/papers/eccv_2024/papers_ECCV/html/6870_ECCV_2024_paper.php.

Junhyuk So, Chiwoong Lee, Shinyoung Lee, Jungseul Ok, and Eunhyeok Park. Improving generative behavior cloning via self-guidance and adaptive chunking. In *Advances in Neural Information Processing Systems*, volume 38, pp. 84432–84456, 2025. doi: 10.52202/085713-2830. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/79ce24f9e8d3c4ff5919240eac78a782-Abstract-Conference.html.

Ziying Song, Caiyan Jia, Lin Liu, Hongyu Pan, Yongchang Zhang, Junming Wang, Xingyu Zhang, Shaoqing Xu, Lei Yang, and Yadan Luo. Don't shake the wheel: Momentum-aware planning in end-to-end autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22432–22441, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Song_Dont_Shake_the_Wheel_Momentum-Aware_Planning_in_End-to-End_Autonomous_Driving_CVPR_2025_paper.html.

Zhipeng Tang, Sha Zhang, Jiajun Deng, Chenjie Wang, Guoliang You, Yuting Huang, Xinrui Lin, and Yanyong Zhang. VLMPlanner: Integrating visual language models with motion planning. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 5040–5049, 2025. doi: 10.1145/3746027.3755730. URL https://doi.org/10.1145/3746027.3755730.

Xiaoyu Tian, Junru Gu, Bailin Li, Yicheng Liu, Yang Wang, Zhiyong Zhao, Kun Zhan, Peng Jia, XianPeng Lang, and Hang Zhao. DriveVLM: The convergence of autonomous driving and large vision-language models. In *Proceedings of the 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 4698–4726, 2025. URL https://proceedings.mlr.press/v270/tian25c.html.

Kim Peter Wabersich and Melanie N. Zeilinger. A predictive safety filter for learning-based control of constrained nonlinear dynamical systems. *Automatica*, 129:109597, 2021. doi: 10.1016/j.automatica.2021.109597.

Haoxuan Wang, Gengyu Zhang, Yan Yan, Ramana Rao Kompella, and Gaowen Liu. VLA knows its limits: Adaptive execution horizons for robot policies. In *European Conference on Computer Vision (ECCV)*, pp. 607–623, 2026. doi: 10.1007/978-3-032-37196-6_33. URL https://link.springer.com/chapter/10.1007/978-3-032-37196-6_33.

Yan Wang, Wenjie Luo, Junjie Bai, et al. Alpamayo-R1: Bridging reasoning and action prediction for generalizable autonomous driving in the long tail, 2025. URL https://arxiv.org/abs/2511.00088.

Zhenhua Xu, Yan Bai, Yujia Zhang, Zhuoling Li, Fei Xia, Kwan-Yee K. Wong, Jianqiang Wang, and Hengshuang Zhao. DriveGPT4-V2: Harnessing large language model capabilities for enhanced closed-loop autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17261–17270, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Xu_DriveGPT4-V2_Harnessing_Large_Language_Model_Capabilities_for_Enhanced_Closed-Loop_Autonomous_CVPR_2025_paper.html.

Bozhou Zhang, Nan Song, Xin Jin, and Li Zhang. Bridging past and future: End-to-end autonomous driving with historical prediction and planning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6854–6863, 2025a. URL https://openaccess.thecvf.com/content/CVPR2025/html/Zhang_Bridging_Past_and_Future_End-to-End_Autonomous_Driving_with_Historical_Prediction_CVPR_2025_paper.html.

Ruifei Zhang, Junlin Xie, Wei Zhang, Weikai Chen, Xiao Tan, Xiang Wan, and Guanbin Li. AdaDrive: Self-adaptive slow-fast system for language-grounded autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5112–5121, 2025b. URL https://openaccess.thecvf.com/content/ICCV2025/html/Zhang_AdaDrive_Self-Adaptive_Slow-Fast_System_for_Language-Grounded_Autonomous_Driving_ICCV_2025_paper.html.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Robotics: Science and Systems XIX*, 2023. doi: 10.15607/RSS.2023.XIX.016. URL https://roboticsproceedings.org/rss19/p016.html.

Siyuan Zhou, Yilun Du, Shun Zhang, Mengdi Xu, Yikang Shen, Wei Xiao, Dit-Yan Yeung, and Chuang Gan. Adaptive online replanning with diffusion models. In *Advances in Neural Information Processing Systems*, volume 36, pp. 44000–44016, 2023. doi: 10.52202/075280-1909. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/893a5db6100028ec814cfd99fe92c31b-Abstract-Conference.html.

Zewei Zhou, Tianhui Cai, Seth 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*, volume 38, pp. 27920–27956, 2025. doi: 10.52202/085713-0942. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/2843fccca5bedd369a4764848b9bd546-Abstract-Conference.html.

# A Implementation Details

## A.1 Evaluated configuration

The evaluated runtime includes the one-step candidate execution described in Appendix A.4. The candidate is not stored for reuse, and these control steps are included in all reported driving and runtime metrics.

**Motion-trigger implementation.** The waypoint and callback intervals coincide, $\Delta_w = \Delta_c$. At age $a_t$, stored positions $p_k$ and headings $\psi_k$ exclude the prepended ego pose. For $0 \le j < N$, $v_j = \|p_{a_t+j+1} - p_{a_t+j}\|_2/\Delta_w$, and

$$\begin{aligned} G_\psi(t) &= \max_{1\le j<N} |\operatorname{wrap}(\psi_{a_t+j} - \psi_{a_t})|, \\ G_a(t) &= \max_{0\le j<N-1} |v_{j+1} - v_j|/\Delta_w, \\ m_t &= \mathbb{I}[G_\psi(t) > \gamma_\psi \ \vee \ G_a(t) > \gamma_a]. \end{aligned} \tag{7}$$

Parameter values are listed in Section 4.1.

## A.2 Trajectory alignment

Candidates require valid shape, finite values, timestamps, and sufficient future time coverage. At simulator time $T_t = T_0 + t\Delta_c$ ($T_0$: rollout start), interpolate $A_t^-$ at candidate times $T_t + \delta_k$. For interpolated samples $(\widetilde{\mathbf{p}}_{\tau,k}, \widetilde{\psi}_{\tau,k})$ and prediction/current rig poses $(\mathbf{c}_\tau, \varphi_\tau)$, $(\mathbf{c}_t, \varphi_t)$ in a common local frame,

$$\mathbf{p}^-_{t,k} = R(\varphi_t)^\top \left(R(\varphi_\tau)\widetilde{\mathbf{p}}_{\tau,k} + \mathbf{c}_\tau - \mathbf{c}_t\right), \tag{8}$$

$$\psi^-_{t,k} = \operatorname{wrap}(\varphi_\tau + \widetilde{\psi}_{\tau,k} - \varphi_t). \tag{9}$$

Here $R$ is a planar rotation; the current ego pose is prepended separately.

## A.3 Geometry and comparison details

With stationary tolerance $\epsilon_s = 10^{-3}$ m, two stationary paths give $\rho_t = 1$; an active stationary path and moving candidate proceed to the map check. Vehicle rectangles $B_j(P)$ use scene-specific dimensions and rig-to-footprint offset $\mathbf{d}_{\text{box}}$, with center $\mathbf{c}_j + R(\varphi_j)\mathbf{d}_{\text{box}}$. For nearby lane-polygon union $\mathcal{L}_j$ and closest road edge $\mathcal{E}_j$, the road-edge violation test is

$$\begin{aligned} e_{\text{lane}}(P) = \textsc{OffRoad} \quad \Longleftrightarrow \quad & \exists j : \operatorname{area}(B_j(P) \setminus \mathcal{L}_j) > \epsilon_A \\ & \wedge \ \operatorname{dist}(B_j(P), \mathcal{E}_j) < \epsilon_d, \end{aligned} \tag{10}$$

with $\epsilon_A = 10^{-6}$ m$^2$ and $\epsilon_d = 10^{-3}$ m. OnRoad: complete sampled lane coverage; PossibleExit: uncovered sampled area without a detected road-edge violation; Unavailable: missing, malformed, or insufficient-horizon geometry. PossibleExit and Unavailable alone do not trigger map-based rejection. When trajectory comparison is unavailable, unless braking has priority, retain an alignable, temporally valid reference or return FallbackRequired. The sampled checking horizon is given by Eq. (6).

## A.4 Recovery and continuation details

When trajectory comparison is unavailable, the active trajectory may be retained at $t = e$ for that control step only; ordinary retention requires $t < e$. When no braking reference is eligible for retention, braking is triggered by a path-length or off-road rejection at the normal trajectory's reuse deadline, or an off-road candidate without an active trajectory whose sampled footprints are fully covered by mapped lane regions. Estimate $\hat{v}_t$ from the latest two ego poses and retime the remaining active polyline by arc length:

$$s(\delta) = \begin{cases} \hat{v}_t\delta - \frac{1}{2}b\delta^2, & 0 \le \delta \le \hat{v}_t/b, \\ \hat{v}_t^2/(2b), & \delta > \hat{v}_t/b. \end{cases} \tag{11}$$

Interpolate at $s(k\Delta_w)$ and repeat the stop pose to fill $K$ points; source geometry must cover $d_{\text{stop}} = \hat{v}_t^2/(2b)$. Assess the full generated horizon: the off-road trigger without complete lane coverage of the sampled footprints along the active trajectory requires a complete, error-free assessment with no detected road-edge violation; other triggers require full sampled lane coverage. Braking references have $H_R = 40$ callbacks. A braking reference may be retained after candidate rejection or when trajectory comparison is unavailable, provided continuation checks and temporal validity hold; otherwise, selection may attempt another construction.

One-step candidate execution applies to four construction failures: an unavailable checking horizon or one exceeding the braking trajectory's duration or reuse limit; missing lane-coverage information required by the triggering condition; active geometry shorter than the stopping distance; or missing active-trajectory geometry or lane information. It requires only candidate format validity, executes the candidate for one control step without storing it for reuse, clears active state, and queries again next callback. An off-road candidate without an active trajectory from which to construct a braking reference follows the same rule. Other unresolved recovery failures return FALLBACKREQUIRED with no executable reference.

## B EVALUATION DEFINITIONS AND RUNTIME ACCOUNTING

**Evaluation accounting.** Driving metrics are averaged over the same set of 913 scenes; event rates report the percentage of affected scenes. Within each scene, driving metrics use evaluator-valid timesteps up to and including the first collision (of any type) or off-road event. Plan deviation uses the same 903 valid scenes for every method without imputation. Runtime is measured over matched post-render simulation intervals with common start and end simulation timestamps.

Table 4: Definitions and aggregation of reported metrics.

| Metric | Definition | Aggregation |
|---|---|---|
| $\text{Score}_{\text{AF}}$ | Progress-based driving score, set to zero by an at-fault collision or off-road event. | Mean over scenes ($\times 100$) |
| $\text{Collision}_{\text{AF}}$ | Any evaluator-attributed front or lateral ego collision. | Any per scene; % of scenes |
| OffRoad | Ego footprint is not contained in the union of nearby lane polygons and lies within 1 mm of the nearest road edge. | Any per scene; % of scenes |
| Wrong Lane | Ego heading differs by more than $2\pi/3$ from a lane-centerline heading for a lane containing the ego centroid. | Any per scene; % of scenes |
| Progress ($P_{\text{rel}}$) | Normalized ego-centroid projection onto the ground-truth trajectory truncated at the current time. | Min per scene; mean over scenes |
| $D_{\text{traj}}$ | Distance from ego centroid to the full ground-truth trajectory. | Max per scene; mean over scenes |
| Plan Dev. | Exponentially weighted mean $\ell_2$ distance between consecutive plans at overlapping future timestamps. | Mean per scene; mean over 903 valid scenes |
| Driver latency | Wall-clock policy-driver execution time, including both VLA queries and reference reuse. | Mean per callback |
| End-to-end latency | Wall-clock interval from the first policy event after initial rendering to simulation-loop timer stop; includes subsequent rendering, simulation, and first-call model warmup; excludes setup and first render. | Mean seconds per scene |
| VLA invocations | Driver callbacks that query the VLA, irrespective of internal sampling count. | Count and rate per 100 callbacks |
| Speedup | FULL REPLAN mean end-to-end latency divided by method mean. | Ratio of aggregate means |

For $\text{Score}_{\text{AF}}$, progress is the normalized ego-centroid projection onto the full ground-truth trajectory at the final retained evaluation timestep, clipped to $[0, 1]$, and divided by $0.8$, with the result capped at

one. Ground-truth paths shorter than 5 m receive full progress credit. An at-fault collision or off-road event sets the score to zero. This progress term differs from the time-truncated $P_{\mathrm{rel}}$ above.

# C Supplementary Controls and Diagnostics

All component comparisons use 84 callbacks per scene; scores are scaled by 100 and rates are percentages.

## C.1 Component ablations

Table 5: **Detailed component ablations on 100 scenes.** These $H = 8$ variants extend Table 3. Check ablations also disable associated recovery triggers; structural validation and remaining recovery rules are retained.

| Configuration | $\mathbf{Score_{AF}} \uparrow$ ($\times 100$) | $\mathbf{Collision_{AF}} \downarrow$ (%) | OffRoad $\downarrow$ (%) | Queries $\downarrow$ (%) |
|---|---|---|---|---|
| **STAR-Drive** | 72.06 | 2 | 2 | 23.69 |
| Without early queries | 69.17 | 0 | 0 | 13.61 |
| Without path-length check | 68.48 | 2 | 3 | 23.69 |
| Without road-geometry check | 67.97 | 2 | 5 | 22.48 |
| One-step candidate execution instead of braking | 67.25 | 2 | 6 | 23.93 |

## C.2 Matched-state command changes

We select each scene's first boundary where STAR-Drive retains its trajectory and Full Replan queries, requiring logged vehicle state and previous commands to match within absolute tolerance $10^{-9}$. Both rollouts must pass the controller-domain screen and provide the next 0.1 s controller output, yielding 860 eligible pairs. We compare absolute changes from the previous to next commands; Section 4.2 reports the results.