REASONING-AWARE SPECULATIVE DECODING FOR EFFICIENT VISION-LANGUAGE-ACTION MODELS IN AUTONOMOUS DRIVING

Anonymous authors
Paper under double-blind review

ABSTRACT

Modern Vision-Language-Action (VLA) planners for autonomous driving emit a chain-of-causation (CoC) reasoning step *before* producing a trajectory. The reasoning is autoregressive and dominates inference latency, while the trajectory head is parallel and cheap. Latency is an operational constraint in autonomous driving, so accelerating the reasoning step is the central problem we address. We observe that CoC reasoning has two qualitatively different needs: most tokens continue routine setup that follows naturally from the ego-trajectory history, and a small fraction encode commitments that require fresh visual evidence about an unexpected situation. We split this reasoning into two specialized paths: a *routine reasoner* that handles the predictable continuation by attending to trajectory history, and a *deliberative reasoner* (the unmodified VLA target) that handles novel cases by attending to current visual evidence, using the speculative decoding framework as the architectural template for how the two paths cooperate. Unlike standard speculative decoding, our routine reasoner is not a smaller replica of the target; the two reasoners are deliberately specialized to read different parts of the prompt. We propose two techniques to realize this. First, we introduce **FlatRoPE**, a 1D rotary positional embedding in the draft that breaks the rotational symmetry of the target's 3D M-RoPE, redirecting attention away from visual tokens and onto trajectory-history tokens. Second, we introduce **Action-aware RL (AARL)**, a post-training stage that uses an action-quality reward together with a static-reference KL anchor. Together, our two-reasoner system reduces the reasoning-step running time significantly across four open driving VLA baselines: Alpamayo-R1, Alpamayo-R1.5, AutoVLA and ORION.

1 INTRODUCTION

Modern Vision-Language-Action (VLA) planners for autonomous driving (Wang et al., 2025c; Hwang et al., 2024; Sima et al., 2023) are sequential by construction: the model first emits a chain-of-causation (CoC) reasoning text (15–20 tokens) and *then* produces a trajectory. The reasoning is autoregressive through an 8B+ multimodal backbone; the trajectory is produced in parallel by a flow-matching action head conditioned on the backbone's KV cache. The reasoning step is the bottleneck: it consumes an order of magnitude more wall-clock budget than the trajectory itself, and latency in driving translates directly into how soon a control commitment can be issued. Accelerating the reasoning step is therefore the central problem we address.

We observe that CoC reasoning has two different purposes. Most tokens continue routine setup. The next token follows naturally from the ego trajectory history (recent ego dynamics, partially-emitted CoC). A small fraction encode edge cases that require fresh visual evidence, such as a pedestrian half-stepping off a curb, an oncoming car crossing a center line – where the next token cannot be predicted from history alone. In this work we split the reasoning step into two paths specialized for these two needs: a *routine reasoner* that handles predictable continuation by attending to trajectory history, and a *deliberative reasoner* that handles the novel cases by attending to current visual evidence. The routine reasoner is fast and the deliberative reasoner is heavier; we use the speculative decoding framework (Leviathan et al., 2023; Chen et al., 2023) as the architectural *template* for how

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.

they cooperate – the routine reasoner proposes the next K tokens, the deliberative reasoner verifies them in one pass, and the longest matching prefix is committed.

Critically, the two reasoners are deliberately specialized to read *different* parts of the prompt: the routine reasoner attends primarily to trajectory-history positions where the next routine token can be predicted, while the deliberative reasoner is the unmodified VLA target and keeps its full visual attention to detect novelty. The role of each is set by the kind of token-prediction it is good at, not by parameter count. On the draft side this specialization comes from a position-encoding change we call **FlatRoPE**. Mirroring the target's 3D M-RoPE, the natural choice for a multimodal draft, couples the draft's attention to the visual token cluster and diverts the attention budget away from the trajectory-history positions where routine signal lives (Section 4.2). FlatRoPE replaces 3D M-RoPE with plain 1D rotary, breaking the rotational symmetry of adjacent vision patches and redirecting the attention budget onto trajectory history. We further introduce a reinforcement-learning post-training stage (Section 4.3) with a three-component reward and a static-reference KL anchor that fixes a self-removal pathology in the standard policy anchor. Our contributions:

1. **FlatRoPE**: plain-1D rotary in the draft, motivated by an attention-entropy diagnostic showing that 3D M-RoPE clusters vision tokens and disproportionately reduces attention mass on trajectory-history tokens. FlatRoPE is the mechanism that produces the routine-reasoner-vs-deliberative-reasoner specialization on the draft side. Across two draft architectures and four open driving VLA targets, FlatRoPE consistently outperforms the 3D-mirroring baseline on accepted length.
2. **AARL post-training** with a static-reference KL anchor. The standard policy anchor self-removes as the draft drifts; the static reference does not. The result is the first post-training method we are aware of that improves both validation *and* held-out test accepted length on an open AV VLA.
3. **Empirical validation that this specialization fits autonomous driving.** We measure per-token attention budget at the draft's input fusion (showing FlatRoPE redirects attention to trajectory history as designed), provide a token-level routing diagnostic on real CoC traces (showing the routine reasoner handles setup tokens autonomously while the deliberative reasoner is selectively invoked at visually-grounded decision points), and quantify reasoning-phase and end-to-end latency reduction on held-out validation and test splits.

In summary, these three contributions produce a two-mode decoder whose routine path handles the majority of CoC tokens without invoking the deliberative path. With $L = 4.98$ at block size 16 the routine path handles $\sim 78\%$ of CoC tokens, accelerating the reasoning phase by up to $4.1\times$ ($2.5\times$ end-to-end) at iso-quality on Alpamayo-R1, with consistent gains on Alpamayo-R1.5, AutoVLA and ORION.

2 Related work

Vision-language-action models for driving. End-to-end VLA models for autonomous driving have rapidly converged on a common recipe: a vision-language backbone consumes multi-camera images and ego state, and an action head produces a future trajectory. EMMA (Hwang et al., 2024) demonstrated end-to-end driving from a Gemini-class backbone; DriveLM (Sima et al., 2023) introduced graph-structured visual question answering as a planning interface. DriveGPT4 (Xu et al., 2024) showed that a multimodal LLM can be fine-tuned to emit interpretable driving rationales.

World-model-grounded VLAs such as GAIA-1 (Hu et al., 2023a) and Wayve's LINGO/UniAD-style stacks (Wayve, 2023; Hu et al., 2023b) couple language reasoning with imagined futures. RoboTron-Drive (Huang et al., 2024) and OmniDrive (Wang et al., 2025a) focus on perception-grounded reasoning, while OpenEMMA (Xing et al., 2024) and MiniDrive (Zhang et al., 2024) aim at compact, deployable variants. Alpamayo-R1 (Wang et al., 2025c) and AutoVLA (Zhou et al., 2025) – the two open targets we evaluate against – additionally emit a chain-of-causation (CoC) reasoning step before invoking a continuous flow-matching or diffusion trajectory head, making the autoregressive language portion the dominant inference cost. Our work targets exactly that bottleneck.

Efficiency for autonomous-driving reasoning. Two complementary directions have emerged. The first is *architectural compaction*: distilling the full VLA into a smaller backbone (MiniDrive (Zhang et al., 2024), TinyDrive (Hassani et al., 2025)), pruning attention heads, or quantizing weights

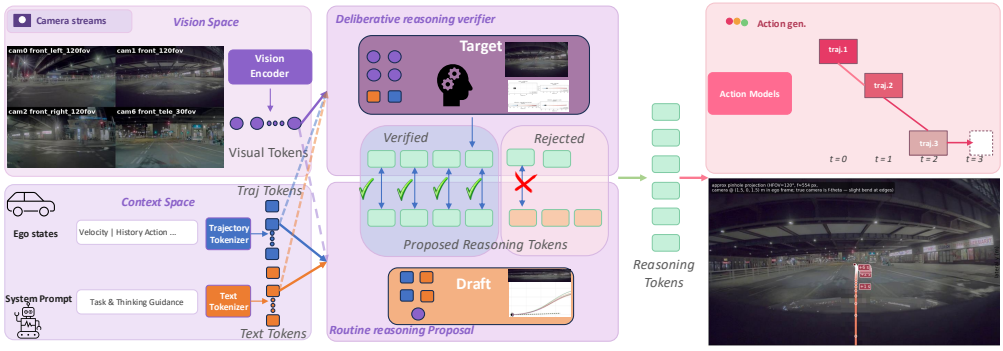

Figure 1: **Reasoning-Aware Speculative Decoding Framework**: The target model encodes visual features and trajectory data into tokens, which a compact draft model then uses to generate reasoning proposals. While the target model maintains a broad visual focus, the draft model prioritizes historical trajectory patterns to perform lightweight, routine-based reasoning. In the event of a novel scenario, any disagreement between the models triggers a rejection by the target, initiating a new round of proposals to ensure accuracy.

and activations for on-vehicle deployment. The second is *reasoning-aware inference*: exiting the reasoning chain early once the decision is settled (Yue et al., 2024), checking at run time that the emitted reasoning still agrees with the action before it is committed (Wu et al., 2025), caching and reusing computation across frames in autoregressive world models (Zeng et al., 2026), or distilling the language step away so that routine clips run vision-only (Gu et al., 2026). Both reduce *when* the deliberative model runs but do not change its per-token cost. Speculative decoding is orthogonal: it preserves the target's exact greedy outputs while spreading the per-step cost across cheap routine-reasoner forwards. To our knowledge, no prior work has applied speculative decoding to AV CoC reasoning.

Speculative decoding and parallel-decoding methods. Leviathan et al. (2023) and Chen et al. (2023) introduced speculative decoding for autoregressive language models. The two main subsequent design axes have been (i) the draft architecture and (ii) the verification topology. On the draft side, Medusa (Cai et al., 2024) adds parallel prediction heads to the target itself; EAGLE/EAGLE-2/EAGLE-3 (Li et al., 2024a;b; 2025) train an autoregressive draft conditioned on multi-layer target hidden states; BiLD (Kim et al., 2023) routes between a small and a large decoder, and KV-reuse drafts (Liu et al., 2026) study how far a draft's KV cache can be carried across long-range speculation; block-diffusion drafts (Arriola et al., 2025; Sahoo et al., 2024) predict groups of tokens in parallel via masked-denoising objectives. On the verification side, tree-decoding variants (Miao et al., 2023; Cai et al., 2024; Li et al., 2024b) verify multiple candidate paths in a single target forward via custom attention masks, trading more verify-time compute for higher accept rate. Lookahead decoding (Fu et al., 2024) eliminates the draft entirely by re-using the target itself in a Jacobi-style parallel decoding loop. Adjacent work lowers the cost of the target's own KV cache by sharing key/value heads (Ainslie et al., 2023). All of these target text-only chat or code completion. None have studied the position-encoding choice for drafts of *multimodal* targets, where the target's native M-RoPE 3D structure couples the draft's attention to the visual cluster: the phenomenon we identify and fix in this paper.

Two-mode reasoning in foundation models. A recurring design pattern in foundation-model inference is to route easy queries past the deliberative path while preserving its outputs: chain-of-thought decomposition with selective rollouts (Wang et al., 2025b) and self-consistency (Wang et al., 2022) both do this at the query level. Standard speculative decoding routes at the token level, with the draft a small imitator of the target trained to match its distribution. We depart from that recipe: our routine reasoner is specialized to attend to trajectory history, while the deliberative reasoner (the unmodified VLA target) keeps its full visual attention, so the two are good at different things by construction and the speculative-decoding loop is the protocol that lets them cooperate. We make this specialization explicit in the AV setting and show where the routine reasoner's attention budget should land: on the small trajectory-history slice where routine signal lives, not on the visual blob the routine tokens never need to inspect.

3 BACKGROUND

We define the two-stage VLA driving target, speculative decoding, and the multimodal rotary position encoding used by the targets we evaluate. Our specific design choices is deferred to Section 4.

3.1 VISION-LANGUAGE-ACTION DRIVING WITH REASONING

Inputs. The target VLA, parameterized by θ_T, consumes a multi-camera observation $\mathbf{I} \in \mathbb{R}^{C\times H\times W}$ (stacked C views), an ego-state / history vector $\mathbf{e} \in \mathbb{R}^{d_e}$, and a textual prompt $\mathbf{P} \in \mathcal{V}^{|\mathbf{P}|}$ over the target tokenizer's vocabulary $\mathcal{V}$. We collect them as $x = (\mathbf{I}, \mathbf{e}, \mathbf{P})$.

Two-stage output. The target produces a rationale $\mathbf{r} \in \mathcal{V}^n$ – a chain-of-causation (CoC) for Alpamayo-R1, a chain-of-thought (CoT) for AutoVLA – followed by an action trajectory $\boldsymbol{\tau} \in \mathbb{R}^{T_f\times d_a}$ over a horizon of T_f control steps in d_a-dimensional action space (typically $d_a = 2$ for longitudinal acceleration and lateral curvature).

Stage 1 (autoregressive reasoning). The rationale is generated token by token by an autoregressive language model:

$$\mathbf{r}_t \sim \pi_T(\cdot \mid x, \mathbf{r}_{1:t-1}; \theta_T), \qquad t = 1, \dots, n, \tag{1}$$

terminating at a model-specific stop token.

Stage 2 (action generation). Conditioned on x and the committed rationale $\mathbf{r}$, the target emits the trajectory through a model-specific function g_T:

$$\boldsymbol{\tau} = g_T(x, \mathbf{r}; \theta_T)\,. \tag{2}$$

The two targets instantiate g_T differently. **Alpamayo-R1** produces $\boldsymbol{\tau}$ via a flow-matching diffusion head reading the backbone's KV cache after position $|\mathbf{P}| + n$:

$$g_T^{\text{Alpamayo}}(x, \mathbf{r}) = \mathrm{ODE}\big[v_{\theta_T}\big(\boldsymbol{\tau}_s, s;\, \mathrm{KV}(x, \mathbf{r})\big)\big]_{s=0\to 1}\,, \tag{3}$$

where v_{θ_T} is a learned vector field, s is the diffusion time, and $\boldsymbol{\tau}_0 \sim \mathcal{N}(0, I)$. AutoVLA emits the trajectory as a discrete token sequence decoded autoregressively from the same backbone:

$$g_T^{\text{AutoVLA}}(x, \mathbf{r}) = \mathrm{detokenize}\big(\mathbf{a}_{1:T_f}\big)\,, \quad \mathbf{a}_t \sim \pi_T(\cdot \mid x, \mathbf{r}, \mathbf{a}_{1:t-1}; \theta_T)\,. \tag{4}$$

Latency budget. Let C_T be the per-token cost of one target forward pass and C_g the cost of g_T. The total inference cost is

$$\underbrace{n \cdot C_T}_{\text{Stage 1}} + \underbrace{C_g}_{\text{Stage 2}}\,. \tag{5}$$

3.2 SPECULATIVE DECODING

A draft D with parameters θ_D defines a surrogate token distribution $\pi_D(\cdot \mid x, \mathbf{r}_{1:t-1}; \theta_D)$ over $\mathcal{V}$. Speculative decoding replaces Equation 1 with an iterative *propose / verify / commit* loop. Let i index the loop iteration, $\mathbf{r}_{1:t_i}$ denote the prefix already committed, and K a block size.

Propose. The draft emits K candidate tokens in a single forward:

$$\hat{\mathbf{r}}_{t_i+1:t_i+K} = \mathrm{Propose}_K(\pi_D, x, \mathbf{r}_{1:t_i})\,. \tag{6}$$

Verify. The target evaluates them in one batched forward pass:

$$\mathbf{r}^\star_{t_i+j} = \arg\max_{v\in\mathcal{V}} \pi_T\big(v \mid x,\, \mathbf{r}_{1:t_i},\, \hat{\mathbf{r}}_{t_i+1:t_i+j-1};\, \theta_T\big), \quad j = 1, \dots, K. \tag{7}$$

Commit. The longest matching prefix is the accepted length:

$$\ell_i \;=\; \max\Big\{l \in \{0, \dots, K\} : \hat{\mathbf{r}}_{t_i+j} = \mathbf{r}^\star_{t_i+j}\ \forall j \le l\Big\}, \tag{8}$$

and the iteration commits $\ell_i + 1$ tokens (the matched prefix plus a "bonus" from the target at position $\ell_i + 1$):

$$\mathbf{r}_{t_i+1:t_i+\ell_i+1} \;=\; (\hat{\mathbf{r}}_{t_i+1}, \dots, \hat{\mathbf{r}}_{t_i+\ell_i}, \mathbf{r}^\star_{t_i+\ell_i+1}), \qquad t_{i+1} \;=\; t_i + \ell_i + 1. \tag{9}$$

The loop terminates when a stop token is committed.

Mean accepted length.

$$L \;=\; \mathbb{E}_i[\ell_i + 1] \;\in\; [1, K+1], \tag{10}$$

where $C_T^{(K)}$ and $C_D^{(K)}$ are the per-iteration target-verify and draft-propose costs. $\mathcal{S}$ models the *reasoning phase* (the CoC decode loop) alone; end-to-end wall-clock additionally includes the fixed vision-prompt prefill, which speculative decoding does not accelerate.

3.3 Multimodal rotary position encoding (M-RoPE)

The targets we evaluate use a multimodal extension of rotary position encoding (M-RoPE), introduced with Qwen2-VL (Wang et al., 2024), on every attention layer.

1D rotary. Let $\mathbf{q} \in \mathbb{R}^h$ be a per-head query at sequence position $m \in \mathbb{N}$ with h even, and let $\boldsymbol{\omega} \in \mathbb{R}^{h/2}$ be a fixed frequency vector (typically $\omega_d = b^{-2d/h}$ for a base b, e.g. $b = 10{,}000$). Standard 1D rotary embedding pairs adjacent dimensions and rotates each pair by an angle proportional to position:

$$\mathrm{RoPE}(\mathbf{q}, m) \;=\; \mathbf{R}(m, \boldsymbol{\omega})\,\mathbf{q}, \qquad \mathbf{R}(m, \boldsymbol{\omega}) = \bigoplus_{d=0}^{h/2-1} \begin{pmatrix} \cos(m\omega_d) & -\sin(m\omega_d) \\ \sin(m\omega_d) & \cos(m\omega_d) \end{pmatrix}, \tag{11}$$

where $\bigoplus$ denotes block-diagonal concatenation.

M-RoPE 3D. For multimodal sequences mixing text and image patches, the target assigns each token a three-axis position $\mathbf{m} = (m^T, m^H, m^W) \in \mathbb{N}^3$ – temporal, height, and width coordinates within the camera-frame grid. The head dimensions are partitioned by an `mrope_section` vector (s^T, s^H, s^W) with $s^T + s^H + s^W = h/2$. M-RoPE applies a separate rotation on each section driven by the corresponding axis position:

$$\mathrm{MRoPE}(\mathbf{q}, \mathbf{m}) \;=\; \begin{bmatrix} \mathbf{R}_{s^T}(m^T,\, \boldsymbol{\omega}^T)\,\mathbf{q}^{(T)} \\ \mathbf{R}_{s^H}(m^H,\, \boldsymbol{\omega}^H)\,\mathbf{q}^{(H)} \\ \mathbf{R}_{s^W}(m^W,\, \boldsymbol{\omega}^W)\,\mathbf{q}^{(W)} \end{bmatrix}, \tag{12}$$

where $\mathbf{q} = [\mathbf{q}^{(T)};\, \mathbf{q}^{(H)};\, \mathbf{q}^{(W)}]$ is the head split along the feature dim, $\mathbf{R}_s$ is the s-pair block-rotation of Equation 11 restricted to $2s$ dimensions, and $(\boldsymbol{\omega}^T, \boldsymbol{\omega}^H, \boldsymbol{\omega}^W)$ are the frequency vectors of the three axes; in practice each axis is allocated only a fraction of the head dimension (concrete values for the backbone we use are given in Section 5). For text tokens the three coordinates collapse to the sequence index, $m^T = m^H = m^W = m$, and Equation 12 reduces to Equation 11. For image patches the three coordinates encode the frame and grid coordinates of the patch.

4 Method

4.1 The two-mode decoder: specialized draft and visual target

Building on Section 3.2, we instantiate the routine reasoner D as a small transformer that ingests the target's hidden states at one or more layers as conditioning and produces $\pi_D(\cdot \mid x, \mathbf{r}_{1:t})$. The hidden-state conditioning is deliberate. D has a small parameter budget, well under 1B, far too small to reproduce the target's visual encoder from scratch: re-running visual reasoning inside D would require scaling it until the speedup vanishes, or accepting a degraded visual representation. We instead let D *reuse* the visual understanding the target has already computed, feeding target hidden states, sliced from one or more layers, into D as cross-attention K/V context. The target does the heavy visual work once, and D's parameters are spent on combining that context with the trajectory-history positions to predict the next routine token. D is therefore a specialist rather than a general mimic of the target: its attention is steered toward trajectory-history positions (recent ego dynamics plus the partially-emitted CoC), and the position-encoding choice in Section 4.2 is what produces that specialization. The consequence is that D is fast at routine tokens, where trajectory history is sufficient, and intentionally weak on novel-situation tokens, where fresh visual evidence is required. The deliberative reasoner is the unmodified VLA target with its native 3D M-RoPE intact, keeping full visual attention for verification and for emitting the bonus token when the routine reasoner is rejected. We evaluate the two draft families established in the spec-decoding literature: a *block diffusion* draft that computes $\hat{\mathbf{r}}_{t+1:t+K}$ via a single masked-denoising step, and an *autoregressive* draft with multi-layer feature fusion that emits K tokens via a depth-bounded chain. Their sizes, layer counts, fusion schemes and references are deferred to Section 5. The trajectory head g_T is unchanged from Equations 3-4; only Stage 1 of Equation 5 is replaced by the two-mode loop.

4.2 FLATROPE: FROM 3D M-ROPE TO 1D IN THE DRAFT

Background: target's 3D M-RoPE. The multimodal language-model backbones used by current VLA targets apply 3D M-RoPE with hyperparameters `mrope_section` $= (s_T, s_H, s_W)$ summing to the rotary half-dim $D_r = D_{\text{head}}/2$, and a boolean `interleaved`. Concretely, for each rotary-pair index $d \in \{0, \dots, D_r - 1\}$ we define an axis assignment $a(d) \in \{T, H, W\}$:

$$a(d) = \begin{cases} T, & 0 \le d < s_T, \\ H, & s_T \le d < s_T + s_H, \\ W, & s_T + s_H \le d < D_r, \end{cases} \quad \text{(blocked)}, \qquad a(d) = \{T, H, W\}_{(d \bmod 3)} \quad \text{(interleaved)}. \tag{13}$$

The specific MLLM family we evaluate against (along with concrete values of `mrope_section` and the `interleaved` flag) is deferred to Section 5; the analysis below holds for any 3D-M-RoPE backbone with s_T, s_H, s_W each substantially smaller than D_r. Each token i carries a 3D coordinate $\big(p_T(i), p_H(i), p_W(i)\big)$, and the rotation angle at rotary-pair d is

$$\phi_d(i) = \theta_d \, p_{a(d)}(i), \qquad \theta_d = 10000^{-2d/D_{\text{head}}}. \tag{14}$$

For text and ego-history tokens, all three coordinates collapse to the sequence index $p_T = p_H = p_W = m$, so $\phi_d(i) = \theta_d \, m(i)$ at every d and 3D M-RoPE reduces to 1D RoPE. For vision patches the three coordinates are independent grid offsets, and the rotation is genuinely three-dimensional.

Why mirroring is harmful. A natural draft design ingests target hidden states (already rotated by target M-RoPE) and applies the same 3D M-RoPE on its own queries – mirroring (s_T, s_H, s_W) and `interleaved`. Two adjacent vision patches at (p_T, p_H, p_W) and $(p_T, p_H, p_W{+}1)$ differ only in the s_W-dim subspace of ϕ (similarly for height-adjacent patches in the s_H subspace). Because each spatial axis is allocated only a fraction s_W/D_r (or s_H/D_r) of the rotary dimensions – typically less than one third in current MLLM backbones – the post-rotation Q/K differ over only that small subspace and remain near-aligned in the rest. The dot-product after rotation collapses onto the cluster, the softmax cannot resolve individual patches, and the draft's attention budget pools on the visual cluster as a whole.

FlatRoPE projection. We replace the draft's 3D rotation with a 1D rotation indexed by the sequence position only. For every token i in the draft input – vision, text, ego-history alike – we set

$$p_T^{\text{flat}}(i) = p_H^{\text{flat}}(i) = p_W^{\text{flat}}(i) \;=\; m(i) \in \{0, 1, \dots, T_{\text{seq}} - 1\}. \tag{15}$$

Substituted into Equation 14, the per-dim phase becomes $\phi_d^{\text{flat}}(i) = \theta_d \, m(i)$, identical for every d *regardless of the axis assignment* $a(d)$. Two important consequences follow.

(i) **`mrope_section` and `interleaved` are structurally irrelevant under FlatRoPE.** Both *blocked* and *interleaved* axis assignments in Equation 13 collapse to the same scalar phase $\theta_d m(i)$, since the axis-selected coordinate is identical across T, H, W. Mismatch between the draft's declared `mrope_section` and the target's no longer affects the draft's queries or keys. We retain only the target's θ_d schedule (same head dim, same base 10000) so target-derived hidden states and draft-side queries share frequency tables; the position projection itself is $\mathbb{R}^3 \to \mathbb{R}$ via $(p_T, p_H, p_W) \mapsto m$.

(ii) **The asymmetry between target and draft is intentional.** The target's hidden states are still produced under the spatially-aware 3D rotation (the target is unmodified); but the draft attends to them with queries rotated by 1D phases. After FlatRoPE projection, two vision patches with sequence positions $m_i \neq m_j$ differ in rotation across *every* rotary-pair d by $\theta_d(m_j - m_i)$, not just the s_W or s_H subspace. Q/K post-rotation are well separated, the softmax discriminates between patches, and the freed budget redistributes onto trajectory-history positions where the next routine token lives.

Empirical confirmation. We measure the consequence on the layer-0 cross-attention pattern of a converged warm-SFT block-diffusion draft (Table 1). Visual tokens occupy ∼96% of the prompt sequence; trajectory and committed-CoC tokens are the remaining 4%. Under 3D M-RoPE the softmax pools mass on the visual cluster (84.5% of total mass) and starves trajectory tokens (7.1%) and the draft's own generated tokens (8.4%). FlatRoPE pulls the visual mass down to 70.8%, restoring the freed budget to trajectory tokens (8.4%) and generated tokens (20.8%) – exactly the positions where the next routine-token signal lives. Within-trajectory-and-text attention entropy correspondingly rises from 0.404 to 0.540, indicating the softmax is now discriminating among trajectory and text positions rather than concentrating on a single sink token.

Table 1: Attention budget at layer-0 cross-attention of a converged block-diffusion draft, averaged over 32 heads and one test clip. 3D M-RoPE drains attention from the action-relevant trajectory and generated tokens onto the visual cluster; 1D restores it.

Token type	1D rotary (ours)	3D M-RoPE (baseline)
visual tokens	0.708	**0.845**
trajectory tokens	**0.084**	0.071
generated tokens	**0.208**	0.084
entropy within trajectory and text (norm.)	**0.540**	0.404

4.3 ACTION-AWARE RL POST-TRAINING

Problem formulation. After supervised fine-tuning, we want to further train the draft to maximize the realized speedup. From Equation 10, this is bounded by the mean accepted length. Define a per-prompt proposal $\hat{\mathbf{r}}_{1:K} \sim \pi_D^\theta(\cdot \mid x)$ and its acceptance length against the target's greedy continuation $\mathbf{r}^\star$:

$$\ell\big(\hat{\mathbf{r}}_{1:K};\, x\big) = \max\big\{\, k \in \{0, \dots, K\} \;:\; \hat{r}_j = r_j^\star \;\forall j \le k \,\big\}. \tag{16}$$

The post-training objective is to find draft parameters θ that maximize the expected acceptance length:

$$\theta^\star \;=\; \arg\max_\theta \; \mathbb{E}_{x\sim\mathcal{D}}\, \mathbb{E}_{\hat{\mathbf{r}}\sim\pi_D^\theta(\cdot|x)} \big[\, \ell(\hat{\mathbf{r}};\, x) \,\big]. \tag{17}$$

Why RL. Equation 17 is non-differentiable in θ: the acceptance length is a $\max$ over hard token-level argmax matches, with no smooth surrogate. The standard supervised proxy – per-position cross-entropy of π_D^θ against $\mathbf{r}^\star$ – treats every position uniformly and ignores the structural fact that an error at position k erases all gains at positions $> k$. Two well-known consequences are: (i) two policies with the same average per-position CE can have quite different L, depending on how their errors distribute across positions; and (ii) once SFT has converged, lowering CE further yields almost no L improvement (the residual CE comes mostly from genuinely uncertain positions). We therefore treat Equation 17 directly as a single-step contextual bandit, sample K rollouts per prompt, and apply group-relative policy optimization (GRPO) to estimate the gradient of $\mathbb{E}[\ell]$.

Action-aware reward. The acceptance length ℓ itself is a sparse 0/1-style signal – it provides no gradient until at least one position matches and it does not distinguish among rollouts that fail at the same position. We therefore replace ℓ with a denser proxy reward grounded in the trajectory the target ultimately produces. For each prompt x we sample $K = 5$ rollouts (stochastic only at greedy-rejected positions; greedy-matched positions are frozen at the deterministic argmax to reduce variance). For the k-th rollout we form a contaminated rationale $\tilde{\mathbf{r}}_k$ by replacing the first N rejected positions of $\mathbf{r}^\star$ with the k-th sample, run the target's flow-matching head g_T to obtain $\boldsymbol{\tau}_k^{\text{mix}} = g_T(x, \tilde{\mathbf{r}}_k)$, and compute

$$R_k \;=\; w_{\text{traj}}\, R_k^{\text{traj}} \;+\; w_{\text{text}}\, R_k^{\text{text}}, \qquad R_k^{\text{traj}} = -\big\|\boldsymbol{\tau}_k^{\text{mix}} - \boldsymbol{\tau}^\star\big\|_2^2, \quad R_k^{\text{text}} = \frac{1}{N} \sum_{j \in \mathcal{R}_k^{[:N]}} \mathbb{1}\big[\hat{r}_{k,j} = r_j^\star\big], \tag{18}$$

where $\boldsymbol{\tau}^\star = g_T(x, \mathbf{r}^\star)$ is the target's greedy trajectory and $\mathcal{R}_k^{[:N]}$ are the substituted positions. The reward is 'action-aware' in the sense that token-level disagreements with the target are penalized in proportion to how much they perturb the target's eventual trajectory, giving the draft direct gradient signal toward the positions whose token choice actually matters for downstream control.

GRPO update. The advantage is the group-relative baseline $\hat{A}_k = R_k - \bar{R}$ with $\bar{R} = \frac{1}{K}\sum_k R_k$. The policy gradient accumulates only over the sampled (rejected) positions: greedy positions have zero log-prob gradient by construction. Combined with the static-reference KL anchor (below), the per-prompt loss is

$$\mathcal{L}(\theta) \;=\; -\frac{1}{K}\sum_{k=1}^{K} \hat{A}_k \sum_{j \in \mathcal{R}_k} \log \pi_\theta\big(\hat{r}_{k,j} \,\big|\, x, \tilde{\mathbf{r}}_{k,<j}\big) \;+\; \beta \cdot \frac{1}{|\mathcal{A}_{\text{ref}}|} \sum_{j \in \mathcal{A}_{\text{ref}}} \mathrm{KL}\big(\pi_\theta(\cdot \mid x, \cdot) \,\|\, \pi_{\text{ref}}(\cdot \mid x, \cdot)\big), \tag{19}$$

with $\beta = 0.02$ in our runs.

Static-reference anchor. The KL term in Equation 19 is gated by an anchor set $\mathcal{A} \subseteq \{1, \dots, K\}$. The standard recipe defines the anchor against the *live* policy: $\mathcal{A}_{\text{policy}}(\theta) = \{j : \arg\max \pi_\theta(\cdot|...) = r_j^\star\}$.

This gating self-removes during drift: as θ updates, a position whose argmax flips from correct to wrong is reclassified out of the anchor set, exactly when the regularizer would have corrected the flip. Empirically this manifests as `draft_final` regressing relative to mid-training checkpoints. We instead use a *static* reference set $\mathcal{A}_{\text{ref}} = \{j : \arg\max \pi_{\text{ref}}(\cdot|...) = r_j^\star\}$ fixed at the start of post-training – positions where the frozen SFT-init was already correct receive permanent anchoring, positions where it was wrong remain free to move under the policy gradient. The single-line code-level diff can be found in appendix.

5 EXPERIMENTS

Targets and drafts. We evaluate four open AV VLA targets. **Alpamayo-R1** (Wang et al., 2025c): 10B parameters, an 8B Qwen3-VL backbone and a flow-matching trajectory head, whose autoregressive component emits CoC reasoning. **Alpamayo-R1.5**: its successor on the same backbone. **AutoVLA** (Zhou et al., 2025): a Qwen2.5-VL-3B backbone with discrete trajectory tokens, evaluated on our reproduction. **ORION** (Fu et al., 2025): a LLaVA-LLaMA-7B driving VLA whose backbone is natively 1D-RoPE. We pair each target with two drafts: **DFlash** (Arriola et al., 2025), a block-diffusion network over multi-layer target hidden states with block size $K{=}16$, and **EAGLE-3** (Li et al., 2025), an autoregressive 1-layer draft with feature fusion (hiddens at layers $\{1, 17, 32\}$ of the 36-layer Qwen3-VL backbone) and a 7-step training rollout. For each pair we train two variants, 3D M-RoPE rotary in the draft (the baseline) and 1D arange rotary (ours), under the same loss and hyperparameters.

Data & Metrics. Alpamayo-R1: 18000 training clips with cached target greedy CoC outputs (mean output length 17.4 tokens). Held-out splits: validation (300 scenes) and test (200 scenes). Mean accepted length L at $K{=}16$ verification, evaluated greedily at temperature 0. Higher is better.

5.1 MAIN RESULT: FLATROPE ACROSS ARCHITECTURES AND TARGETS

Table 2 reports mean accepted length L and two measured speedups for each (target, draft, rotary) cell: the *reasoning speedup* (wall-clock of the CoC decode phase alone, prefill excluded) and the *end-to-end speedup* (whole pipeline including the fixed vision-prompt prefill). The two differ by the prefill time of target model: on Alpamayo-class targets the $\sim$0.4s prefill bounds e2e to $\approx 2.3\times$ regardless of draft quality, while the reasoning phase itself accelerates by up to $6.9\times$; on long-output targets (AutoVLA) the two converge.

(i) FlatRoPE consistently improves over 3D-mirroring across draft families and targets. On Alpamayo-R1 + block-diffusion draft, FlatRoPE lifts test L by $+0.20$ ($4.75 \rightarrow 4.95$), raising the reasoning speedup from $3.32\times$ to $3.89\times$ and end-to-end from $2.32\times$ to $2.48\times$; the same ordering holds on the validation split. On Alpamayo-R1 + autoregressive draft, FlatRoPE reaches $L = 7.41$ (vs. 7.33). The gains repeat on Alpamayo-R1.5, where the block-diffusion draft adds $+0.45$ ($5.32\times \rightarrow 6.46\times$ reasoning) and the autoregressive draft adds $+0.65$ ($4.12\times \rightarrow 4.81\times$ reasoning, e2e $2.43\times \rightarrow 2.45\times$), and on AutoVLA ($+0.08$, e2e $2.30\times \rightarrow 2.38\times$). ORION admits no 3D mirror because its backbone is natively 1D, and the same FlatRoPE drafts transfer unchanged: the block-diffusion draft reaches $L = 5.32$ with a $3.21\times$ reasoning speedup ($1.49\times$ e2e), and the autoregressive draft reaches $L = 6.20$ with $2.22\times$ ($1.48\times$ e2e).

(ii) AARL adds a further small but consistent gain on top of FlatRoPE. With AARL post-training ($K{=}5$ rollouts, static-reference anchor), Alpamayo-R1 + block-diffusion reaches $L = 4.98$ and lifts the reasoning speedup from $3.89\times$ to $4.12\times$ ($2.49\times$ e2e); the autoregressive draft ticks up to $L = 7.43$. On Alpamayo-R1.5 the block-diffusion draft gains $+0.05$ ($11.13 \rightarrow 11.18$, reasoning $6.46\times \rightarrow 6.89\times$) and the autoregressive draft gains $+0.17$ ($10.75 \rightarrow 10.92$, reasoning $4.81\times \rightarrow 4.96\times$), while AutoVLA improves end-to-end to $2.39\times$. The same gain carries to ORION's native-1D draft ($5.32 \rightarrow 5.36$, reasoning $3.21\times \rightarrow 3.22\times$), whose end-to-end speedup is already prefill-bound near $1.5\times$. Gains hold on both validation and test splits.

5.2 AARL ABLATIONS

Table 3 shows the contribution of the static-reference anchor and the choice of K within AARL on Alpamayo + DFlash 1D. The static-reference anchor adds $+0.010$ val and $+0.005$ test over the policy-anchor variant under the same hyperparameters. Increasing K above 5 yields marginal val

Table 2: Unified cross-target results on the held-out test split. We report accepted length L, the reasoning speedup (the CoC decode phase alone, prefill excluded) and the end-to-end speedup, with K the speculation depth per verification. E2E includes the fixed vision-prompt prefill, which bounds it to $\approx 2.3\times$ on Alpamayo-class targets whatever the draft quality (Amdahl's law), and is lower on ORION, whose CoC decode is a smaller share of total latency. Alpamayo-R1 and AutoVLA e2e are isolated-H100, and Alpamayo-R1.5 e2e is measured on L40S.

Draft	Rotary	K	Accepted length L	Reasoning speedup	E2E
Alpamayo-R1 (AR baseline)					
EAGLE-3	3D M-RoPE	16	7.33	3.31×	2.24 ×
EAGLE-3	FlatRoPE (ours)	16	7.41	3.33×	2.28 ×
EAGLE-3	FlatRoPE + AARL (ours)	16	7.43	3.36×	**2.28** ×
DFlash	3D M-RoPE	16	4.75	3.32×	2.32×
DFlash	FlatRoPE (ours)	16	*4.95*	3.89×	2.48×
DFlash	FlatRoPE + AARL (ours)	16	**4.98**	**4.12**×	**2.49**×
Alpamayo-R1.5 (AR baseline, L40S)					
DFlash	3D M-RoPE	16	10.68	5.32×	2.70×
DFlash	FlatRoPE (ours)	16	11.13	6.46 ×	2.71×
DFlash	FlatRoPE + AARL (ours)	16	11.18	6.89 ×	**2.72**×
EAGLE-3	3D M-RoPE	16	10.10	4.12×	2.43×
EAGLE-3	FlatRoPE (ours)	16	10.75	4.81×	2.45×
EAGLE-3	FlatRoPE + AARL (ours)	16	10.92	4.96×	**2.47**×
AutoVLA (AR baseline)					
DFlash	3D M-RoPE	8	2.99	2.42×	2.30×
DFlash	FlatRoPE (ours)	8	**3.07**	2.43×	2.38×
DFlash	FlatRoPE + AARL (ours)	8	**3.07**	**2.45**×	**2.39**×
ORION (natively 1D backbone; no 3D mirror exists)					
DFlash	FlatRoPE (native 1D)	8	5.32	3.21×	**1.49**×
DFlash	FlatRoPE + AARL (ours)	8	5.36	3.22×	1.48×
EAGLE-3	FlatRoPE (native 1D)	8	**6.20**	2.22×	1.48×

Table 3: AARL ablations on Alpamayo + DFlash 1D. lr $= 10^{-6}$, N=3 contamination width, $w_{\text{traj}} = 1.0$, $w_{\text{text}} = 0.5$, 2 epochs. (We use lower N and epoch for quick results)

Configuration	best val L	val Δ	best test L	test Δ
no RL	4.124	0	4.208	0
RL, policy anchor	4.534	+0.010	4.220	+0.012
RL, static-ref anchor (ours)	**4.544**	+0.020	**4.225**	+0.017
K=5 (default)	**4.544**	+0.020	**4.225**	+0.017
K=10	4.548	+0.024	4.214	+0.006
K=15	4.543	+0.019	4.220	+0.012

gains but loses test, the same val/test divergence we observed when raising the learning rate – both indicate over-fitting val sample-specific quirks. We adopt K=5 as the default.

6 Conclusion

We addressed the autoregressive-reasoning bottleneck in VLA driving planners by splitting the reasoning step into two specialized paths coordinated through speculative decoding: a routine reasoner that attends to trajectory history, and a deliberative reasoner (the unmodified VLA target) that keeps its full visual attention. **FlatRoPE** is the mechanism that produces this specialization, and it consistently raises acceptance length across two draft architectures and four open driving VLA targets; AARL post-training with a static-reference KL anchor adds further improvement. The combined system is, to our knowledge, the first VLA decoder that accelerates reasoning by up to $4.1\times$ on a 10B open AV target and $6.9\times$ on its successor ($2.5\times$ and $2.7\times$ end-to-end).

REFERENCES

Joshua Ainslie et al. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In *EMNLP*, 2023.

Marianne Arriola, Aaron Gokaslan, Justin T. Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models, 2025. ICLR.

Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D. Lee, Deming Chen, and Tri Dao. Medusa: Simple LLM inference acceleration framework with multiple decoding heads, 2024. ICML.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling, 2023.

Haoyu Fu, Diankun Zhang, Zongchuang Zhao, Jianfeng Cui, Dingkang Liang, Chong Zhang, Dingyuan Zhang, Hongwei Xie, Bing Wang, and Xiang Bai. Orion: A holistic end-to-end autonomous driving framework by vision-language instructed action generation, 2025.

Yichao Fu et al. Break the sequential dependency of LLM inference using lookahead decoding. In *ICML*, 2024.

Jing Gu, Niccolò Cavagnero, and Gijs Dubbelman. Orion-Lite: Distilling LLM reasoning into efficient vision-only driving models, 2026. CVPR Workshops.

Hossein Hassani, Soodeh Nikan, and Abdallah Shami. TinyDrive: Multiscale visual question answering with selective token routing for autonomous driving, 2025.

Anthony Hu et al. GAIA-1: A generative world model for autonomous driving, 2023a.

Yihan Hu et al. Planning-oriented autonomous driving. In *CVPR*, 2023b.

Zhijian Huang, Chengjian Feng, Feng Yan, Baihui Xiao, Zequn Jie, Yujie Zhong, Xiaodan Liang, and Lin Ma. RoboTron-Drive: All-in-one large multimodal model for autonomous driving, 2024.

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, 2024.

Sehoon Kim, Karttikeya Mangalam, Suhong Moon, Jitendra Malik, Michael W. Mahoney, Amir Gholami, and Kurt Keutzer. Speculative decoding with big little decoder. In *NeurIPS*, 2023.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *ICML*, 2023.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE: Speculative sampling requires rethinking feature uncertainty, 2024a.

Yuhui Li et al. EAGLE-2: Faster inference of language models with dynamic draft trees, 2024b. EMNLP.

Yuhui Li et al. EAGLE-3: Scaling up inference acceleration of large language models via training-time test, 2025.

Tianyu Liu, Yuhao Shen, Xinyi Hu, Baolin Zhang, Hengxin Zhang, Jun Dai, Jun Zhang, Shuang Ge, Lei Chen, Yue Li, and MingCheng Wan. When hidden states drift: Can KV caches rescue long-range speculative decoding?, 2026.

Xupeng Miao et al. SpecInfer: Accelerating generative large language model serving with tree-based speculative inference and verification, 2023. ASPLOS.

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T. Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models, 2024. NeurIPS.

Chonghao Sima et al. DriveLM: Driving with graph visual question answering, 2023.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-VL: Enhancing vision-language model's perception of the world at any resolution, 2024.

Shihao Wang, Zhiding Yu, Xiaohui Jiang, Shiyi Lan, Min Shi, Nadine Chang, Jan Kautz, Ying Li, and Jose M. Alvarez. OmniDrive: A holistic vision-language dataset for autonomous driving with counterfactual reasoning, 2025a. CVPR.

Xiaoqiang Wang, Suyuchen Wang, Yun Zhu, and Bang Liu. System-1.5 reasoning: Traversal in language and latent spaces with dynamic shortcuts, 2025b. NeurIPS.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models, 2022.

Yan Wang, Wenjie Luo, Junjie Bai, Yulong Cao, Tong Che, Ke Chen, Yuxiao Chen, Jenna Diamond, Yifan Ding, Wenhao Ding, Liang Feng, Greg Heinrich, Jack Huang, Peter Karkus, Boyi Li, Pinyi Li, Tsung-Yi Lin, Dongran Liu, Ming-Yu Liu, Langechuan Liu, Zhijian Liu, Jason Lu, Yunxiang Mao, Pavlo Molchanov, Lindsey Pavao, Zhenghao Peng, Mike Ranzinger, Ed Schmerling, Shida Shen, Yunfei Shi, Sarah Tariq, Ran Tian, Tilman Wekel, Xinshuo Weng, Tianjun Xiao, Eric Yang, Xiaodong Yang, Yurong You, Xiaohui Zeng, Wenyuan Zhang, Boris Ivanovic, and Marco Pavone. Alpamayo-r1: Bridging reasoning and action prediction for generalizable autonomous driving in the long tail, 2025c.

Wayve. LINGO-1: Exploring natural language for autonomous driving. Wayve blog post, 2023. https://wayve.ai/thinking/lingo-natural-language-autonomous-driving/.

Yilin Wu, Anqi Li, Tucker Hermans, Fabio Ramos, Andrea Bajcsy, and Claudia Pérez-D'Arpino. Do what you say: Steering vision-language-action models via runtime reasoning-action alignment verification, 2025.

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, 2024.

Zhenhua Xu et al. DriveGPT4: Interpretable end-to-end autonomous driving via large language model. *IEEE Robotics and Automation Letters*, 2024.

Yang Yue, Yulin Wang, Bingyi Kang, Yizeng Han, Shenzhi Wang, Shiji Song, Jiashi Feng, and Gao Huang. DeeR-VLA: Dynamic inference of multimodal large language models for efficient robot execution, 2024. NeurIPS.

Yixiao Zeng, Jianlei Zheng, Chaoda Zheng, Shijia Chen, Mingdian Liu, Tongping Liu, Tengwei Luo, Yu Zhang, Boyang Wang, Linkun Xu, Siyuan Lu, Bo Tian, and Xianming Liu. X-Cache: Cross-chunk block caching for few-step autoregressive world models inference, 2026.

Enming Zhang, Xingyuan Dai, Min Huang, Yisheng Lv, and Qinghai Miao. MiniDrive: More efficient vision-language models with multi-level 2d features as text tokens for autonomous driving, 2024.

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, 2025.

A AI USE STATEMENT

We use AI to help with implementation and polish the paper draft. All the methodologies are proposed by authors.

B REPRODUCIBILITY STATEMENT

The source code is available at: `https://anonymous.4open.science/r/spec_auto-E45B`. We are committed to release all the checkpoints and code beyond the acceptance.

C IMPLEMENTATION DETAILS

The source code is available at: `https://anonymous.4open.science/r/spec_auto-E45B` **1D rotary in the draft.** For each draft architecture we replace the rotary embedding module with a vanilla 1D rotary keyed on the absolute sequence position only. The rest of the architecture is unchanged. Position ids passed to the draft's attention are simple `torch.arange` shapes; the target's M-RoPE 3D positions are not touched.

Static-reference anchor: code-level diff. See Section 4.3.

The single-line code-level diff is:

```
# policy anchor (prior art):
anchor_mask = (pi_logits.argmax(-1) == r_star)        # depends on live pi
# static-reference anchor (ours):
anchor_mask = (pi_ref_logits.argmax(-1) == r_star)    # frozen ref only
```

Reward chunking for memory. For $K \geq 8$, the per-step rollout batch through the target's flow-matching head exceeds GPU memory. We chunk along the K dimension at `k_chunk_size`, accumulate per-chunk action predictions, and concatenate before computing the reward. Each chunk uses an independent random seed for the diffusion sampler so within-group rollouts remain independent.

D ROUTING DIAGNOSTIC: WHEN IS THE DELIBERATIVE REASONER INVOKED?

For a typical 17-token CoC, the routine reasoner handles the first 4–5 tokens at $\sim 80\%$ acceptance (setup tokens like "the car is"), drops to $\sim 50\%$ around the verb / action tokens ("slowing", "yielding"), and recovers near the trailing punctuation. Invocation of the deliberative reasoner is therefore concentrated in the middle of the chain where the action commitment is encoded, which matches the two-mode intuition: deliberation is needed exactly where the decision crystallizes.