DO WE REALLY NEED MULTI-STEP REFINEMENT FOR VLA PLANNERS?

Anonymous authors
Paper under double-blind review

ABSTRACT

Pretrained vision-language models bring rich semantic understanding and trajectory reasoning to autonomous driving, spurring a growing family of Vision-Language-Action (VLA) planners. Existing approaches either adopt iterative decoding paradigms inherited from text and image generation, or employ downstream refinement modules widely used in robotic manipulation. However, neither is compatible with the stringent latency of autonomous driving. Since future trajectories are low-dimensional, kinematically constrained, and orders of magnitude simpler than images or text sequences, we challenge the need for such complex pipelines in VLA planning. To this end, we present SnapVLA, a framework that generates the complete trajectory in a single parallel forward pass via masked discrete diffusion. We demonstrate that the demand for multi-step refinement originates from two factors that do not inherently require it: the policy optimization paradigm and the local incoherence among trajectory points. Regarding optimization, single-step diffusion yields an exact sequence likelihood and an unbiased GRPO objective, whereas existing multi-step chains generally requires likelihood approximations. Regarding coherence, we tokenize trajectories into a structured action space and apply a lightweight closed-form smoother, yielding kinematically consistent trajectories without a learned continuous expert. On NAVSIM v1 and v2, SnapVLA achieves 91.0 PDMS and 90.1 EPDMS, on par with state-of-the-art models. Notably, it takes 180 ms per inference on a single A800 GPU, over 2$\times$ faster than the fastest baseline, demonstrating the potential of the single-step paradigm for efficient planning. Code will be released at https://anonymous.4open.science/status/snapvla-D21C.

1 INTRODUCTION

Autonomous driving requires predicting future trajectories that are safe, comfortable, and kinematically feasible under strict real-time constraints. End-to-end approaches learn a direct mapping from sensory observations to future motion, avoiding error accumulation in cascaded perception–prediction–planning systems (Li et al., 2022; Chen et al., 2024; Sun et al., 2025; Zheng et al., 2024; Liao et al., 2025; Wu et al., 2026). Yet driving demonstrations alone provide limited semantic and contextual knowledge for long-tailed and interaction-rich scenarios. Pretrained vision-language models (VLMs) offer a source of such knowledge through large-scale multimodal pretraining, motivating Vision-Language-Action (VLA) planners that extend VLMs to future-trajectory prediction. Consequently, a growing body of VLA planners (Li et al., 2026b; Yuan et al., 2026; Arai et al., 2025; Bai et al., 2023) has explored different ways of generating future trajectories from VLMs.

Existing VLA planners generally follow two paradigms for generating future trajectories, as illustrated in Fig. 1. The first treats trajectories as discrete tokens and generates them autoregressively (Chi et al., 2026; Luo et al., 2025; Chen et al., 2026; Ye et al., 2026). While this design is naturally compatible with the token-prediction interface of VLMs, its sequential decoding makes inference latency grow with trajectory length and can lead to error accumulation along the generated trajectory. The second conditions a separate continuous action predictor, such as a diffusion or flow-matching model, on representations extracted from a VLM (Xiong et al., 2026; Liu et al., 2026; Li et al., 2026b). This design avoids token-level autoregressive decoding and the discrete–continuous gap, but introduces an additional learned module for trajectory prediction. Despite their different implementations, both paradigms introduce iterative computation into trajectory genera-

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.

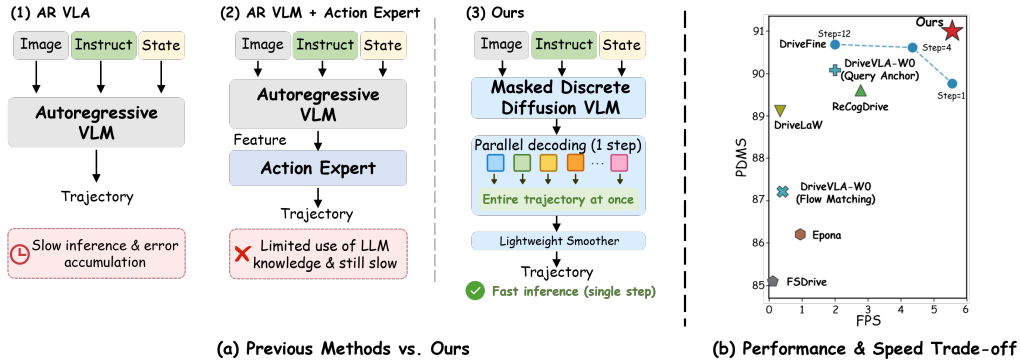


Figure 1: Comparison of trajectory paradigms. (a) Existing VLA planners use multi-step decoding or extra action experts. In our SnapVLA, the full trajectory is predicted in one masked-diffusion pass and refined by a lightweight smoother. (b) Previous methods suffer from high inference latency and experience a large performance drop with an insufficient number of decoding steps on NAVSIM v1 (Dauner et al., 2024). In contrast, SnapVLA achieves state-of-the-art performance with single-step inference. It runs over $2\times$ faster than the fastest baseline, as measured on an A800 GPU.

tion. Although multi-step decoding and refinement are reasonable choices for complex image or language generation, their repeated computation adds to the substantial cost of running a VLM and is ill-suited to latency-sensitive autonomous driving applications. Driving trajectories, in contrast, are low-dimensional, highly structured, and subject to strong kinematic constraints. These properties raise a fundamental question: is multi-step refinement truly necessary for VLA planning, or can a carefully designed single-step policy generate reliable driving trajectories?

A fully parallel single-step design must address two issues that can make iterative generation appear necessary for practical autonomous driving. Group Relative Policy Optimization (GRPO) commonly optimizes VLA with trajectory-level rewards. For a multi-step diffusion policy, however, computing the probability of a final trajectory requires marginalizing over intermediate denoising paths, so existing formulations rely on approximate likelihoods. Without subsequent refinement to compensate for optimization errors, such a policy places greater demands on likelihood accuracy, making an exact sequence likelihood essential. Another challenge is the local kinematic coherence. As all tokens are predicted independently from a fully masked input, neighboring waypoints may be locally inconsistent even when the overall plan is semantically reasonable.

To address these issues, we present SnapVLA, a VLA planner that generates the complete future trajectory in a single parallel forward pass. We represent trajectories in a structured discrete action space and predict all action tokens simultaneously from a fully masked sequence through masked discrete diffusion. This single-step formulation provides a directly computable sequence likelihood, enabling GRPO with an exact policy ratio for the deployed generation process. GRPO improves the planning quality of the single-step policy, while a lightweight closed-form smoother based on third-order difference regularization improves local kinematic consistency. The smoother operates on the predicted trajectory with fixed endpoints, avoiding the need for a learned continuous action expert. On NAVSIM v1, SnapVLA achieves 91.0 PDMS, and on NAVSIM v2 it achieves 90.1 EPDMS, matching current state-of-the-art performance with single-step inference. By comparison, previous methods such as DriveFine suffer a noticeable performance drop when reducing the denoising budget from 12 steps to one, as shown in Fig. 1(b). In contrast, SnapVLA preserves state-of-the-art performance with single-step inference and achieves more than twice the inference throughput of the fastest baseline. It therefore provides a competitive and efficient alternative to existing multi-step VLA planners. Our contributions are summarized as follows:

- We present SnapVLA, a VLA planner combining structured binned action tokens, parallel single-step masked discrete diffusion, and exact-likelihood GRPO. The exact sequence likelihood aligns optimization with the deployed policy, while a lightweight closed-form smoother based on third-order differences improves local consistency without a learned action expert.
- On NAVSIM v1 and v2, SnapVLA achieves 91.0 PDMS and 90.1 EPDMS, respectively, with single-step inference. It also achieves more than twice the inference throughput of the fastest baseline, demonstrating a favorable performance–efficiency trade-off for VLA planning.

2 Related Work

End-to-End Autonomous Driving. End-to-end autonomous driving unifies perception, prediction, and planning, reducing information loss and error accumulation across modular pipelines. UniAD (Hu et al., 2023), VAD (Jiang et al., 2023), and TransFuser (Chitta et al., 2022) represent efforts to learn driving behavior from unified scene representations. Recent generative approaches further model future trajectories directly with diffusion or flow matching (Liao et al., 2025; Xing et al., 2025). Together, these works establish direct trajectory prediction as an alternative to modular planning. However, their ability to generalize to rare and out-of-distribution scenarios remains limited, motivating recent efforts to incorporate the broader semantic and contextual knowledge of vision-language models.

Vision Language Action Models. Pretrained VLMs provide VLA planners with semantic and contextual knowledge that is difficult to acquire from driving demonstrations alone. Building on these capabilities, existing VLA planners extend multimodal understanding to future-trajectory prediction through one of two designs. One line represents trajectories as discrete or textual tokens and generates them autoregressively (Hwang et al., 2024; Shang et al., 2026; Zhou et al., 2026a; Ye et al., 2026). This design is naturally compatible with the token-prediction interface of VLMs, but its sequential decoding incurs latency that grows with the number of generated tokens. Another line uses VLM representations to condition a separate continuous action predictor, such as a diffusion or flow-matching planner (Xiong et al., 2026; Li et al., 2026b). This design avoids token-level decoding and the discrete–continuous gap, but introduces an additional learned module and may require iterative computation to produce the final trajectory. Despite their different implementations, both designs add sequential or iterative computation to VLA trajectory generation, creating a huge deployment cost for latency-sensitive autonomous driving.

Policy Optimization for VLA Planning. Supervised imitation can reproduce demonstrated driving behavior, while reward-based policy optimization provides a direct way to improve trajectory performance. Recent works have applied policy-optimization methods to VLA planners, including AlphaDrive (Jiang et al., 2025), ReCogDrive (Xiong et al., 2026), WAM-Diff (Xu et al., 2025), and DriveFine (Dang et al., 2026). For autoregressive policies, the likelihood of a trajectory factorizes into token-level conditional probabilities and can therefore be evaluated directly. For multi-step diffusion policies, however, a final trajectory may result from multiple intermediate denoising paths, so its exact probability requires marginalizing over these paths. Existing masked-diffusion planners consequently rely on approximate or path-based likelihood estimates for policy optimization, leaving exact trajectory-level optimization for fully parallel masked diffusion largely unexplored.

3 Methodology

This section presents SnapVLA, a VLA planner for efficient trajectory generation. As shown in Fig. 1, SnapVLA utilizes masked discrete diffusion to predict structured action tokens in a single step. We then introduce GRPO with an exact sequence likelihood in Sec. 3.2 and a closed-form smoother in Sec. 3.3 to improve trajectory quality and kinematic consistency, respectively.

3.1 Overview of SnapVLA

Problem Formulation. At each decision step t, VLA models observe a multimodal driving context $c_t = \{I_t, s_t, n_t\}$, where I_t, s_t, and n_t denote the camera observation, the ego-vehicle state, and the navigation instruction, respectively. Given c_t, the planner predicts a future ego trajectory $\tau = (\tau_1, \ldots, \tau_K)$ with K waypoints, where the k-th waypoint is represented as $\tau_k = (x_k, y_k, \psi_k)$ and x_k, y_k, and ψ_k denote its position and heading. The planner samples a token sequence from a policy conditioned on the driving context and obtains the continuous trajectory by de-quantization, which we denote abstractly as $\hat{\tau} \sim \pi_\theta(\cdot \mid c_t)$.

Trajectory Representation. Existing VLA planners generally generate trajectories through one of two interfaces. One type adopts action-expert on top of the VLM features and is capable of predicting trajectories in continuous space (Xiong et al., 2026; Liu et al., 2026; Li et al., 2026b). In contrast, the VLM-direct paradigm instead asks the VLM to generate the trajectory itself with a sequence of discrete tokens (Hwang et al., 2024; Shang et al., 2026; Zhou et al., 2026a; Ye et al., 2026). The

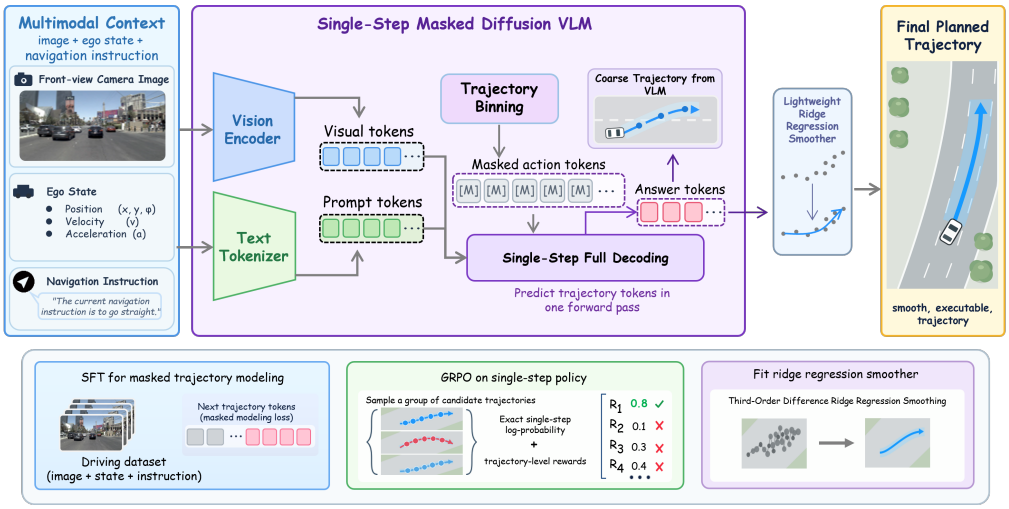


Figure 2: Overview of SnapVLA. Given multimodal driving context, SnapVLA represents trajectories with structured binned tokens and predicts them in parallel from a fully masked sequence. During training, the VLM first learns masked-token reconstruction and is then optimized with GRPO using exact single-step sequence likelihoods and trajectory-level rewards. During inference, the predicted tokens are de-quantized and passed through a non-learned third-order ridge-regression smoother to obtain the final trajectory.

Table 1: Successful trajectory decoding rate (%) under different denoising budgets.

Representation	1 step	48 steps	96 steps	192 steps
Unbinned rate	66.4	98.7	99.6	100.0
Binned rate	100.0	–	–	–

unbinned representation maps each finely discretized coordinate value directly to a numerical action token, preserving a fine-grained correspondence between tokens and coordinates but resulting in a large and weakly structured output space (Xu et al., 2025; 2026). Such an output space makes it difficult for independently predicted tokens to form a valid and kinematically plausible trajectory. This limitation is particularly problematic for single-step generation, where all action tokens are predicted in parallel without an iterative refinement process. As shown in Tab. 1, only 66.4% of unbinned outputs are valid in a single step. Increasing the budget beyond 96 steps recovers nearly all valid decodings, but incurs an approximately linear increase in computation.

Therefore, we resort to a binned representation, which first partitions each coordinate range into fixed quantization intervals and maps each interval index to an action token (Dang et al., 2026). Specifically, we uniformly quantize the longitudinal and lateral coordinates over $[-100\,\mathrm{m}, 100\,\mathrm{m}]$ into 4000 bins, with a resolution of $0.05\,\mathrm{m}$. We quantize the heading over $[-1.7, 1.7]$ rad into 340 bins, with a resolution of 0.01 rad. Each waypoint is represented by three coordinate-specific action tokens in the order of longitudinal position, lateral position, and heading. A trajectory τ with K waypoints is converted into a discrete token sequence through a tokenizer:

$$\mathbf{z}_0 = \mathrm{Tok}(\tau) = \left(z_0^{(1)}, \ldots, z_0^{(L)}\right), \qquad L = 3K, \tag{1}$$

where $z_0^{(i)}$ denotes the i-th trajectory token and L is the total number of tokens. The predicted token indices are de-quantized back to continuous coordinates before trajectory evaluation. In the end, the binned representation achieves 100% decoding validity in a single step.

Masked Discrete Diffusion. During training, the VLM learns to reconstruct trajectory tokens from partially observed sequences. Let M denote the special `[MASK]` token and let $r \in (0, 1]$ be the masking ratio. Given a clean token sequence $\mathbf{z}_0$, we independently replace each token with M with probability r to obtain a corrupted sequence $\mathbf{z}_r$. This masking process is denoted by

$$q_r(\mathbf{z}_r \mid \mathbf{z}_0) = \prod_{\ell=1}^{L} \left[(1-r)\mathbb{I}\left(z_r^{(\ell)} = z_0^{(\ell)}\right) + r\mathbb{I}\left(z_r^{(\ell)} = M\right)\right]. \tag{2}$$

Let $\mathcal{M}_r = \{\ell \mid z_r^{(\ell)} = M\}$ be the set of masked positions. Conditioned on c_t and $\mathbf{z}_r$, the VLM produces a categorical distribution for each position, denoted by $p_\theta^{(\ell)}(v \mid c_t, \mathbf{z}_r)$, where v is a trajectory-token value. We optimize the masked-token reconstruction objective:

$$\mathcal{L}_{\text{diff}} = -\mathbb{E}_{r\sim p_{\text{mask}}(r),\, \mathbf{z}_r\sim q_r(\cdot|\mathbf{z}_0)} \left[\frac{1}{|\mathcal{M}_r|} \sum_{\ell\in\mathcal{M}_r} \log p_\theta^{(\ell)} \left(z_0^{(\ell)} \mid c_t, \mathbf{z}_r \right) \right], \tag{3}$$

where $p_{\text{mask}}(r)$ is the chosen masking-ratio distribution, and samples with $|\mathcal{M}_r| = 0$ are omitted. Thus, the model learns to infer missing trajectory tokens rather than copy visible ones. At inference time, we mask all trajectory positions and predict the complete token sequence in a single parallel forward pass, after which the predicted tokens are de-quantized into a raw trajectory.

3.2 Exact Sequence Likelihood and GRPO

Although single-step decoding substantially reduces inference cost, predicting all trajectory tokens in parallel without iterative refinement can impose an inherent ceiling on trajectory accuracy, since local prediction errors cannot be corrected by subsequent denoising steps. Supervised fine-tuning (SFT) trains the planner to imitate reference trajectories, but does not directly optimize the trajectory-level driving rewards that determine planning quality. We therefore apply the commonly used GRPO after SFT to further improve the quality of single-step plans using driving rewards.

GRPO for Diffusion Policies. Applying GRPO to a diffusion policy requires the likelihood of each complete sampled trajectory, since the reward is assigned at the trajectory level. For a conventional multi-step masked diffusion policy, the probability of the final trajectory is consequently obtained by marginalizing over all intermediate S denoising states:

$$\pi_\theta^{\text{multi}}(\mathbf{z}_0 \mid c_t) = \sum_{\mathbf{z}_{1:S-1}} \prod_{s=1}^{S} P_\theta^{(s)}(\mathbf{z}_{s-1} \mid \mathbf{z}_s, c_t), \tag{4}$$

where $P_\theta^{(s)}$ and $\mathbf{z}_s$ denote transition distribution and the predicted token at step s, respectively.

Exact computation requires summing over every possible configuration of the intermediate token sequences. The number of such configurations grows exponentially with both the sequence length L and the number of denoising steps S. Therefore, exact marginalization is computationally prohibitive in practice. Recent masked-diffusion planners (Xu et al., 2025; Dang et al., 2026) use tractable path-based or factorized likelihood estimates for policy optimization. These estimates avoid summing over alternative denoising paths, but can introduce a mismatch between the likelihood used for optimization and the distribution used during deployment. This mismatch is particularly undesirable for a single-step planner, whose predictions cannot be corrected by extra refinements and thus have limited tolerance to likelihood approximation. Whenever the deployed policy admits an exact likelihood, we therefore use it directly rather than introduce an additional surrogate.

Exact Single-Step Likelihood. Unlike a multi-step diffusion policy, SnapVLA samples all L trajectory tokens from the same fully masked input in a single forward pass for reinforcement learning. Let $\mathbf{o} = (o^{(1)}, \ldots, o^{(L)})$ denote the sampled trajectory-token sequence, whose de-quantized trajectory is $Q^{-1}(\mathbf{o})$. Because no intermediate denoising sequences are generated, the deployed policy's sequence log-likelihood is directly computable as

$$\log \pi_\theta^{\text{1step}}(\mathbf{o} \mid c_t) = \sum_{\ell=1}^{L} \log p_\theta^{(\ell)} \left(o^{(\ell)} \mid c_t, \mathbf{M}^L \right). \tag{5}$$

This is the exact log-likelihood of the deployed rollout policy, rather than the log-likelihood of a sampled denoising path or a surrogate policy.

GRPO Optimization. Starting from the SFT-initialized policy, we sample G candidate token sequences $\{\mathbf{o}_i\}_{i=1}^G$ from the old policy for each context c_t. After de-quantization, each candidate is evaluated with a trajectory-level driving reward R_i. We compute a group-relative advantage by subtracting the mean reward within the group as $\hat{A}_i = R_i - \text{mean}\left(\{R_j\}_{j=1}^G\right)$. Using the exact sequence log-likelihood above, the importance ratio for candidate i is computed in log-space as

$$\rho_i = \exp\left(\log \pi_\theta^{\text{1step}}(\mathbf{o}_i \mid c_t) - \log \pi_{\theta_{\text{old}}}^{\text{1step}}(\mathbf{o}_i \mid c_t) \right). \tag{6}$$

The trajectory policy is optimized with the clipped GRPO objective:

$$\mathcal{L}_{\mathrm{GRPO}} = -\frac{1}{G}\sum_{i=1}^{G} \min\left(\rho_i \hat{A}_i, \mathrm{clip}(\rho_i, 1-\epsilon, 1+\epsilon)\hat{A}_i\right) + \beta D_{\mathrm{KL}}\left(\pi_\theta \| \pi_{\mathrm{ref}}\right). \tag{7}$$

3.3 Closed-Form Kinematic Smoothing

Although the preceding components improve general driving performance, such as collision rate and route efficiency, single-step prediction has an inherent limitation: all trajectory tokens are predicted in parallel without conditioning on one another's sampled values. The predictions therefore cannot coordinate during decoding, which can lead to locally inconsistent waypoint transitions and poor comfort. A natural solution is to attach a continuous action expert that models the dependencies among waypoints and refines the VLM output. However, this introduces an additional learned prediction module and extra forward computation, increasing inference cost and weakening the efficiency advantage of single-step planning. We instead apply a lightweight, deterministic post-processing smoother to regularize the de-quantized trajectory. Let $\hat{\tau} \in \mathbb{R}^{K\times 3}$ denote the raw trajectory predicted by the VLM. We obtain the final trajectory $\tilde{\tau}$ by solving

$$\tilde{\tau} = \underset{\tau' \in \mathbb{R}^{K\times 3}}{\mathrm{argmin}} \ \|\tau' - \hat{\tau}\|^2 + \lambda \|D^3 \tau'\|^2 \quad \text{subject to} \quad \tau'_1 = \hat{\tau}_1, \quad \tau'_K = \hat{\tau}_K, \tag{8}$$

where D^3 is the third-order finite-difference operator along the waypoint dimension and λ controls the smoothness–fidelity trade-off. The first term keeps the smoothed trajectory close to the VLM prediction, while the second term penalizes high-order variation between neighboring waypoints. The endpoint constraints preserve the predicted first and final waypoints. Because the objective is quadratic and the constraints are linear, the solution is obtained by solving a fixed linear system at inference time. The smoother is non-learned, uses only the predicted raw trajectory, and does not require ground-truth waypoints or endpoints.

4 Experiments

4.1 Experimental Setup

Datasets and Evaluation Metrics. We evaluate our planner on NAVSIM (Dauner et al., 2024), a standardized benchmark for data-driven motion planning built from real-world nuScenes driving scenes (Caesar et al., 2020). Following the official protocol, we use the navtrain split for training and report results on the held-out navtest split. These splits contain 1,192 and 136 scenes, respectively.

In NAVSIM v1, model performances are evaluated by Predictive Driver Model Score (PDMS) which combines five complementary criteria: no at-fault collision (NC), drivable-area compliance (DAC), ego progress (EP), time-to-collision (TTC), and comfort (C). NAVSIM v2 further evaluates driving-direction compliance (DDC), traffic-light compliance (TLC), lane keeping (LK), and extended comfort (EC), which are incorporated into EPDMS together with the v1 criteria.

Implementation Details. Our training and inference pipeline consists of four stages. In *Stage 1*, we adapt the LaViDa-8B VLM (Li et al., 2026a), equipped with a SigLIP-so400M visual encoder, to driving data using approximately 800k VQA samples and 400k trajectory samples for 10 epochs. In *Stage 2*, we fine-tune the VLM on NAVSIM (Dauner et al., 2024; Caesar et al., 2020) for 5 epochs using the binned trajectory-token representation described in Sec. 3.1. In *Stage 3*, we optimize the single-step policy with GRPO for 2 epochs on a filtered NAVSIM subset using groups of $G = 16$ sampled trajectories. The GRPO reward combines the PDMS components as $\mathrm{Reward} = \mathrm{Format} \times \mathrm{NC} \times \mathrm{DAC} \times (5\,\mathrm{EP} + 5\,\mathrm{TTC} + 2\,\mathrm{C})/12$, where $\mathrm{Format} \in [0, 1]$ indicates whether the trajectory is successfully decoded. In *Stage 4*, we apply the non-learned closed-form kinematic smoother to the de-quantized trajectory, with the smoothness coefficient set to $\lambda = 0.05$. The initial VLM adaptation uses 32 NVIDIA A800 GPUs, while the subsequent stages use 8 NVIDIA A800 GPUs.

4.2 Main Results

NAVSIM v1 Results. The results on NAVSIM v1 are summarized in Tab. 2. Our method, SnapVLA, achieves a PDMS of 91.0 using only single-step masked discrete trajectory prediction. This result is

Table 2: Quantitative comparison with existing methods on the NAVSIM v1 benchmark. C: Camera, L: LiDAR. Best results are in **bold**, second best are underlined.

Method	Venue	Sensors	NC↑	DAC↑	TTC↑	C.↑	EP↑	PDMS↑
Human	–	–	100	100	100	99.9	87.5	94.8
End-to-End Methods								
DiffusionDrive (Liao et al., 2025)	CVPR'25	C&L	98.2	96.2	94.7	**100**	82.2	88.1
VADv2 (Jiang et al., 2026)	ICLR'26	C	98.3	97.4	95.7	**100**	82.3	89.3
ResAD (Zheng et al., 2026)	CVPR'26	C&L	98.9	97.8	94.9	**100**	87.0	90.6
World Model Methods								
Epona (Zhang et al., 2025)	ICCV'25	C	97.9	95.1	93.8	99.9	80.4	86.2
DriveLaW (Xia et al., 2026)	CVPR'26	C	99.0	97.1	96.7	**100**	81.3	89.1
FSDrive (Zeng et al., 2026)	NeurIPS'25	C	98.2	93.8	93.3	99.9	80.1	85.1
VLM-based Methods								
AutoVLA (Zhou et al., 2026b)	NeurIPS'25	C	98.4	95.6	**98.0**	99.9	81.9	89.1
WAM-Flow (Xu et al., 2026)	CVPR'26	C	**99.2**	98.3	97.0	99.7	82.3	90.3
DriveFine (Dang et al., 2026)	arXiv'26	C	98.6	97.9	95.2	99.9	85.5	90.7
ExploreVLA (Sheng et al., 2026)	ECCV'26	C	98.8	98.4	96.5	99.9	83.5	90.4
AutoDrive-P^3 (Ye et al., 2026)	ICLR'26	C	99.1	97.4	96.5	**100**	84.8	90.6
VLM + Action Expert Methods								
DriveVLA-W0 (Li et al., 2026b)	ICLR'26	C	98.7	**99.1**	95.3	99.3	83.3	90.2
ReCogDrive (Xiong et al., 2026)	ICLR'26	C	97.9	97.3	94.9	**100**	**87.3**	90.8
SpanVLA (Zhou et al., 2026c)	arXiv'26	C	99.1	97.1	95.2	**100**	86.3	90.3
Vega (Zuo et al., 2026)	arXiv'26	C	98.9	95.3	96.1	**100**	81.6	87.9
Ours								
SnapVLA	–	C	98.8	98.3	96.3	**100**	84.8	**91.0**

Table 3: Comparison with existing methods on the NAVSIM v2 benchmark.

Method	NC↑	DAC↑	DDC↑	TLC↑	EP↑	TTC↑	LK↑	C.↑	EC↑	EPDMS↑
WAM-Flow	98.5	94.5	99.5	99.8	86.9	96.8	97.4	97.6	73.9	84.7
DriveFine	98.7	97.3	99.5	99.8	**88.7**	97.8	97.7	**98.4**	83.8	89.7
ExploreVLA	98.8	96.2	**99.6**	99.8	87.1	98.2	**97.8**	98.3	**86.8**	88.8
ReCogDrive	98.3	95.2	99.5	99.8	87.1	97.5	96.6	98.3	86.5	83.6
DriveVLA-W0	98.5	**99.1**	98.0	99.7	86.4	98.1	93.2	97.9	58.9	86.1
SpanVLA	98.8	96.7	99.2	99.8	86.3	97.7	95.0	98.3	85.1	86.4
Vega	**98.9**	95.3	99.4	**99.9**	87.0	**98.4**	96.5	98.3	76.3	86.9
SnapVLA	98.8	98.4	99.3	99.8	87.8	98.3	97.1	98.2	82.7	**90.1**

competitive with substantially more elaborate VLA and action-expert pipelines, while retaining the low-latency inference structure of a single-step planner. The strong aggregate score is supported by high safety and compliance scores, including 98.8 NC, 98.3 DAC, and 100 C.

NAVSIM v2 Results. The results on NAVSIM v2 are reported in Tab. 3. SnapVLA achieves an EPDMS of 90.1, the highest aggregate score among the methods listed in the table. The model obtains particularly strong scores on NC, DAC, TLC, and TTC, indicating that the structured token representation, exact-likelihood GRPO, and closed-form smoothing remain effective under reactive traffic evaluation. The component metrics also show that the aggregate improvement does not come at the cost of basic safety or rule compliance.

Inference Efficiency. We benchmark as many publicly available or reproducible VLA planners as possible under a unified evaluation protocol, as summarized in Tab. 4. Despite using single-step inference, SnapVLA attains a strong PDMS of 91.0. More importantly, it requires only 0.18 s per inference on a single NVIDIA A800 GPU, about half the time of DriveFine and corresponding to roughly twice the inference throughput of the fastest baseline. These results highlight the obvious efficiency advantage of single-step trajectory generation.

4.3 Ablation Study

This section details our ablation study and analyzes the effects of the individual components.

Effect of Binned Trajectory Representation. The unbinned representation has limited reliability for single-step decoding. After Stage 2 fine-tuning, its one-step configuration obtains only 57.5 PDMS and 57.2 EPDMS, whereas increasing the denoising budget to 192 steps recovers these met-

Table 4: Comparison of planning performance and inference efficiency on NAVSIM v1. All inference times in the table are measured on a single NVIDIA A800 GPU. AE denotes action expert.

Method	Backbone	Paradigm	AE	PDMS↑	Infer Time↓
FSDrive	Qwen2-VL-2B	Autoregressive		85.1	8.94s
Epona	DiT-2.5B	Autoregressive	✓	86.2	1.05s
DriveLAW	–	Diffusion		89.1	3.05s
ReCogDrive	InternVL3-8B	Autoregressive	✓	89.6	0.36s
DriveVLA-W0	Emu3-8B	Autoregressive	✓	90.2	0.50s
DriveFine	LaViDa-8B	Masked diffusion		90.7	0.39s
Ours	LaViDa-8B	Masked diffusion		91.0	0.18s

Table 5: Ablation study of the main components of SnapVLA on NAVSIM v1 and v2. Here, UB and B denote unbinned and binned trajectory representations, respectively; Traj. Rep. and Denois. denote trajectory representation and denoising, respectively.

ID	Stage 1 SFT	Stage 2 SFT	Stage 3 GRPO	Stage 4 Smoother	Traj. Rep.	Denois. Steps	V1 PDMS↑	V2 EPDMS↑	V2 EC↑
0	✓				UB	1	11.5	10.9	3.1
1	✓					192	84.9	85.4	86.4
2	✓	✓			UB	1	57.5	57.2	50.1
3	✓	✓				192	87.3	87.9	87.1
4	✓	✓			B	1	88.9	85.5	49.8
5	✓	✓	✓			1	91.0	86.0	42.8
6	✓	✓	✓	✓		1	91.0	90.1	82.7

rics to 87.3 and 87.9, respectively, as reported in Tab. 5. In contrast, binned tokens achieve 88.9 PDMS and 85.5 EPDMS in one step, confirming their suitability for single-step inference.

Effect of GRPO. Starting from the binned single-step model, GRPO improves the task-level planning metrics, increasing PDMS from 88.9 to 91.0 and EPDMS from 85.5 to 86.0, as shown in Tab. 5. However, the EC score decreases from 49.8 to 42.8, indicating that trajectory-level reward optimization does not explicitly enforce local waypoint continuity or comfort.

Effect of Smoother. Our smoother is designed to address this discontinuity. As shown in Tab. 5, smoothing preserves the PDMS at 91.0 while improving EPDMS from 86.0 to 90.1 and EC from 42.8 to 82.7. This improvement stems from enhanced trajectory continuity: as shown in Tab. 6, the smoother reduces Jerk_x from 3.632 to 0.254 and Jerk_{xy} from 4.332 to 0.324. The reduction in jerk indicates smoother local motion and explains the substantial gains in EC and EPDMS, while leaving the task-level planning score unchanged. Overall, these results demonstrate that the lightweight smoother effectively alleviates local discontinuities introduced by single-step prediction.

Qualitative Analysis. Fig. 3 provides qualitative evidence for the effects of the proposed components. At Stage 2, the unbinned representation often fails to produce a valid trajectory under one-step decoding, as shown by the failed cases in examples (a) and (b). Even when a trajectory can be decoded, its quality can be worse than that of the 192-step counterpart, with visible oscillations in cases such as (d). In contrast, the binned representation consistently produces valid trajectories in a single step across the examples. Adding GRPO further improves the task-level behavior: it resolves safety-related failures in (a), improves insufficient progress in (d), and corrects time-to-collision issues in (b) and (c). These examples occasionally reveal the trajectory discontinuity problem. The final Stage 4 smoother regularizes local variations and produces smoother, more executable trajectories, completing the progression from reliable decoding to high-quality single-step planning.

4.4 Discussion

Potential of single step VLM Planning. SnapVLA performs trajectory planning with a single parallel VLM forward pass, avoiding iterative decoding. Its structured discrete action space allows the VLM to use its native token-prediction interface without an additional learned action expert. The resulting trajectory is then regularized by a deterministic smoother, separating semantic planning

Table 6: Comparison of trajectory smoothness before and after closed-form smoothing. Jerk_x, Jerk_y, and Jerk_{xy} denote longitudinal, lateral, and combined mean absolute jerk, respectively.

Method	$\text{Jerk}_x \downarrow$	$\text{Jerk}_y \downarrow$	$\text{Jerk}_{xy} \downarrow$
SnapVLA w/o smoother	3.632	1.420	4.332
SnapVLA w/ smoother	0.254	0.133	0.324

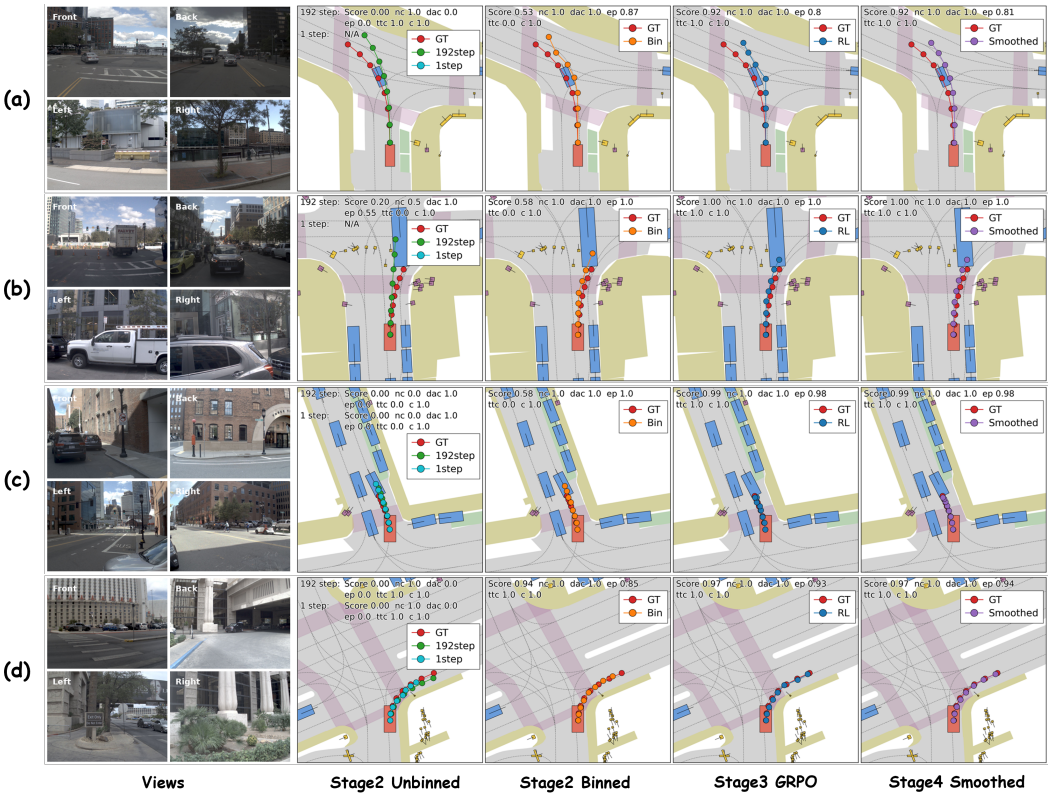


Figure 3: Qualitative comparison of trajectory predictions across different stages.

from local geometric refinement. The same paradigm may be extended to richer latent representations (Mei et al., 2026), such as future scene, world representation (Zhang et al., 2026; 2025; Wang et al., 2024; Zhao et al., 2026), or reasoning features (Zeng et al., 2026).

Limitations and Future Work. SnapVLA is currently evaluated with monocular front view images, while multi camera and richer temporal inputs remain unexplored. The discrete trajectory representation uses fixed quantization intervals, which limit the balance between vocabulary size and trajectory precision. Future work may explore richer visual inputs, more flexible trajectory representations, and tighter joint optimization between the two modules.

5 Conclusion

We revisit the assumption that VLA planners require multi-step refinement. We show that this requirement stems not from the planning task itself, but from two sources: the policy-optimization scheme, and the local incoherence that fully factorized decoding introduces. We present SnapVLA, which generates the complete trajectory in a single parallel forward pass via masked discrete diffusion over a structured action space. Single-step decoding admits an exact sequence likelihood, allowing GRPO to be applied without the approximations that multi-step chains require, and improving base planning performance. A lightweight smoother restores kinematic consistency, removing the need for a learned continuous expert. On NAVSIM v1 and v2, SnapVLA matches state-of-the-art accuracy while running over $2\times$ faster than existing VLA planners. These results suggest that single-step discrete diffusion is a sufficient and efficient paradigm for VLA planning.

AI USE STATEMENT

In this work, generative AI tools were used only for providing suggestions on color choices for scientific figures and for editing portions of the manuscript to improve readability. We did not use generative AI tools for any of the tasks requiring mandatory disclosure under the ICLR AI policy. All AI-assisted suggestions and edits were reviewed and approved by the authors. The authors take full responsibility for the final content of this work.

REFERENCES

Hidehisa Arai, Keita Miwa, Kento Sasaki, Kohei Watanabe, Yu Yamaguchi, Shunsuke Aoki, and Issei Yamamoto. Covla: Comprehensive vision-language-action dataset for autonomous driving. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 1933–1943. IEEE, 2025.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *2020 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 11618–11628. IEEE, 2020.

Canyu Chen, Yuguang Yang, Zhewen Tan, Yizhi Wang, Ruiyi Zhan, Haiyan Liu, Xuanyao Mao, Jason Bao, Xinyue Tang, Linlin Yang, et al. Devil is in narrow policy: Unleashing exploration in driving vla models. *arXiv preprint arXiv:2603.06049*, 2026.

Li Chen, Penghao Wu, Kashyap Chitta, Bernhard Jaeger, Andreas Geiger, and Hongyang Li. End-to-end autonomous driving: Challenges and frontiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):10164–10183, 2024.

Haohan Chi, Huan-ang Gao, Ziming Liu, Jianing Liu, Chenyu Liu, Jinwei Li, Kaisen Yang, Yangcheng Yu, Zeda Wang, Wenyi Li, et al. Impromptu vla: Open weights and open data for driving vision-language-action models. *Advances in Neural Information Processing Systems*, 38, 2026.

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

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

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

Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, et al. Planning-oriented autonomous driving. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17853–17862. IEEE, 2023.

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

Bo Jiang, Shaoyu Chen, Qing Xu, Bencheng Liao, Jiajie Chen, Helong Zhou, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. Vad: Vectorized scene representation for efficient autonomous driving. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 8306–8316. IEEE, 2023.

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

Bo Jiang, Shaoyu Chen, Hao Gao, Bencheng Liao, Qian Zhang, Wenyu Liu, and Xinggang Wang. Vadv2: End-to-end vectorized autonomous driving via probabilistic planning. In *International Conference on Learning Representations*, volume 2026, pp. 68886–68900, 2026.

Guofa Li, Yifan Yang, Shen Li, Xingda Qu, Nengchao Lyu, and Shengbo Eben Li. Decision making of autonomous vehicles in lane change scenarios: Deep reinforcement learning approaches with risk awareness. *Transportation research part C: emerging technologies*, 134:103452, 2022.

Shufan Li, Konstantinos Kallidromitis, Hritik Bansal, Akash Gokul, Yusuke Kato, Kazuki Kozuka, Jason Kuen, Zhe Lin, Kai-Wei Chang, and Aditya Grover. Lavida: A large diffusion language model for multimodal understanding. *Advances in Neural Information Processing Systems*, 38: 105101–105134, 2026a.

Yingyan Li, Shuyao Shang, Weisong Liu, Bing Zhan, Haochen Wang, Yuqi Wang, Yuntao Chen, Xiaoman Wang, Yasong An, Chufeng Tang, et al. Drivevla-w0: World models amplify data scaling law in autonomous driving. In *International Conference on Learning Representations*, volume 2026, pp. 7890–7911, 2026b.

Bencheng Liao, Shaoyu Chen, Haoran Yin, Bo Jiang, Cheng Wang, Sixu Yan, Xinbang Zhang, Xiangyu Li, Ying Zhang, Qian Zhang, et al. Diffusiondrive: Truncated diffusion model for end-to-end autonomous driving. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12037–12047. IEEE, 2025.

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

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

Xiaodong Mei, Diankun Zhang, Hongwei Xie, Guang Chen, Hangjun Ye, and Dan Xu. Lvdrive: Latent visual representation enhanced vision-language-action autonomous driving model. *arXiv preprint arXiv:2605.22089*, 2026.

Shuyao Shang, Yuntao Chen, Yuqi Wang, Yingyan Li, and ZHAO-XIANG ZHANG. Drivedpo: Policy learning via safety dpo for end-to-end autonomous driving. *Advances in Neural Information Processing Systems*, 38:81565–81585, 2026.

Zihao Sheng, Xin Ye, Jingru Luo, Sikai Chen, and Liu Ren. Explorevla: Dense world modeling and exploration for end-to-end autonomous driving. *arXiv preprint arXiv:2604.02714*, 2026.

Wenchao Sun, Xuewu Lin, Yining Shi, Chuang Zhang, Haoran Wu, and Sifa Zheng. Sparsedrive: End-to-end autonomous driving via sparse scene representation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8795–8801. IEEE, 2025.

Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, Jiagang Zhu, and Jiwen Lu. Drivedreamer: Towards real-world-drive world models for autonomous driving. In *European conference on computer vision*, pp. 55–72. Springer, 2024.

Yang Wu, Qiang Meng, Zhaojiang Liu, Youquan Liu, Jian Yang, and Jin Xie. Distill to think, foresee to act: Cognitive-physical reinforcement learning for autonomous driving. *arXiv preprint arXiv:2605.21139*, 2026.

Tianze Xia, Yongkang Li, Lijun Zhou, Jingfeng Yao, Kaixin Xiong, Haiyang Sun, Bing Wang, Kun Ma, Guang Chen, Hangjun Ye, et al. Drivelaw: Unifying planning and video generation in a latent driving world. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 39701–39712, 2026.

Zebin Xing, Xingyu Zhang, Yang Hu, Bo Jiang, Tong He, Qian Zhang, Xiaoxiao Long, and Wei Yin. Goalflow: Goal-driven flow matching for multimodal trajectories generation in end-to-end autonomous driving. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1602–1611. IEEE, 2025.

Kaixin Xiong, Xiangyu Guo, Fang Li, Sixu Yan, Gangwei Xu, Lijun Zhou, Long Chen, Haiyang Sun, Bing Wang, Kun Ma, et al. Recogdrive: A reinforced cognitive framework for end-to-end autonomous driving. In *International Conference on Learning Representations*, volume 2026, pp. 157518–157556, 2026.

Mingwang Xu, Jiahao Cui, Feipeng Cai, Hanlin Shang, Zhihao Zhu, Shan Luan, Yifang Xu, Neng Zhang, Yaoyi Li, Jia Cai, et al. Wam-diff: A masked diffusion vla framework with moe and online reinforcement learning for autonomous driving. *arXiv preprint arXiv:2512.11872*, 2025.

Yifang Xu, Jiahao Cui, Zhihao Zhu, Hanlin Shang, Shan Luan, Mingwang Xu, Feipeng Cai, Neng Zhang, Yaoyi Li, Jia Cai, et al. Wam-flow: Parallel coarse-to-fine motion planning via discrete flow matching for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24918–24928, 2026.

Yuqi Ye, Zijian Zhang, Junhong Lin, Shangkun Sun, Changhao Peng, and Wei Gao. Autodrive-p3: Unified chain of perception-prediction-planning thought via reinforcement fine-tuning. *arXiv preprint arXiv:2603.28116*, 2026.

Zhenlong Yuan, Chengxuan Qian, Jing Tang, Rui Chen, Zijian Song, Lei Sun, Xiangxiang Chu, Yujun Cai, Dapeng Zhang, and Shuo Li. Autodrive-r^2: Incentivizing reasoning and self-reflection capacity for vla model in autonomous driving. In *International Conference on Learning Representations*, volume 2026, pp. 97934–97955, 2026.

Shuang Zeng, Xinyuan Chang, Mengwei Xie, Xinran Liu, Yifan Bai, Zheng Pan, Mu Xu, and Xing Wei. Futuresightdrive: Thinking visually with spatio-temporal cot for autonomous driving. *Advances in Neural Information Processing Systems*, 38:67299–67318, 2026.

Jinqing Zhang, Zehua Fu, Zelin Xu, Wenying Dai, Qingjie Liu, and Yunhong Wang. Resworld: Temporal residual world model for end-to-end autonomous driving. *arXiv preprint arXiv:2602.10884*, 2026.

Kaiwen Zhang, Zhenyu Tang, Xiaotao Hu, Xingang Pan, Xiaoyang Guo, Yuan Liu, Jingwei Huang, Li Yuan, Qian Zhang, Xiao-Xiao Long, et al. Epona: Autoregressive diffusion world model for autonomous driving. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 27220–27230. IEEE, 2025.

Zongchuang Zhao, Xin Zhou, Tianyang Xu, Zhengyang Sun, Kaixuan Zhou, Honglin Li, Dingkang Liang, and Xiang Bai. Simwam: A simple world action model for end-to-end autonomous driving. *arXiv preprint arXiv:2608.07468*, 2026.

Wenzhao Zheng, Ruiqi Song, Xianda Guo, Chenming Zhang, and Long Chen. Genad: Generative end-to-end autonomous driving. In *European Conference on Computer Vision*, pp. 87–104. Springer, 2024.

Zhiyu Zheng, Shaoyu Chen, Haoran Yin, Xinbang Zhang, Jialv Zou, Xinggang Wang, Qian Zhang, and Lefei Zhang. Resad: Normalized residual trajectory modeling for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3729–3739, 2026.

Xingcheng Zhou, Xuyuan Han, Feng Yang, Yunpu Ma, Volker Tresp, and Alois Knoll. Opendrivevla: Towards end-to-end autonomous driving with large vision language action model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 13782–13790, 2026a.

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. *Advances in Neural Information Processing Systems*, 38: 27920–27956, 2026b.

Zewei Zhou, Ruining Yang, Yiluan Guo, Sherry X Chen, Tao Feng, Kateryna Pistunova, Yishan Shen, Lili Su, Jiaqi Ma, et al. Spanvla: Efficient action bridging and learning from negative-recovery samples for vision-language-action model. *arXiv preprint arXiv:2604.19710*, 2026c.

Sicheng Zuo, Yuxuan Li, Wenzhao Zheng, Zheng Zhu, Jie Zhou, and Jiwen Lu. Vega: Learning to drive with natural language instructions. *arXiv preprint arXiv:2603.25741*, 2026.