LLaDA-Nav: Diffusion-Based Action Generation for Visual Navigation

Anonymous authors
Paper under double-blind review

Abstract

Vision-language models (VLMs) provide broad visual-semantic knowledge and promising generalization for embodied AI. To produce navigation actions, autoregressive VLM policies generate action chunks from left to right, conditioning each action on earlier predictions from the same chunk. In our evaluation, however, these policies show limited closed-loop performance in visual navigation. To test whether this feedback pathway contributes to that limitation, we conduct controlled comparisons and prefix interventions that replace model-generated action prefixes with ground-truth ones. Our results suggest that early prediction errors can propagate to later actions, especially under distribution shift. Motivated by this finding, we propose LLaDA-Nav, a visual navigation policy built on a masked diffusion VLM. LLaDA-Nav fully masks the target action block during training and predicts the entire chunk in a single forward pass, eliminating within-chunk target-action reuse. It incorporates previously executed actions as additional context to capture recent robot motion. Across four navigation datasets and unseen HM3D environments, LLaDA-Nav improves open-loop trajectory prediction and achieves 59.03% closed-loop success, exceeding the strongest evaluated baseline by 15.28 percentage points.

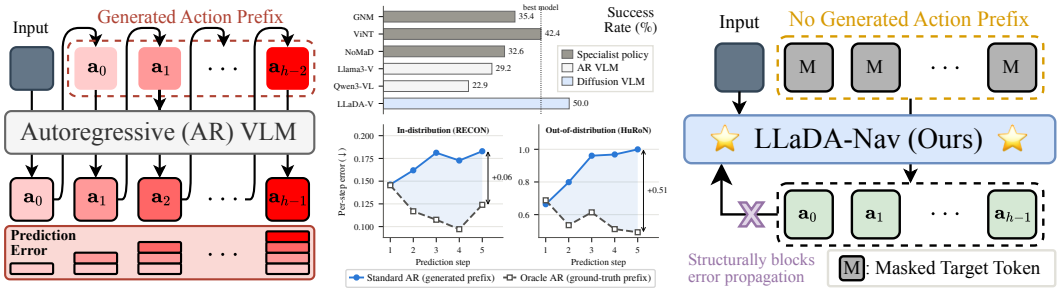
1 Introduction

Visual navigation requires an embodied agent to move through an environment using visual observations while pursuing a spatial objective. Such an objective can be specified in different forms; in the image-goal setting studied here, a target image specifies the destination. Recent approaches range from specialist policies trained on large-scale navigation data (Shah et al., 2023a;b; Sridhar et al., 2024) to world models that predict future observations for planning (Bar et al., 2025; Dong et al., 2025) and systems that incorporate general-purpose visual-language models (VLMs). In particular, VLMs have been used at several levels of the navigation stack, including subgoal imagination (Chen et al., 2026), zero-shot decision guidance (Wang et al., 2026), and action or trajectory prediction (Hirose et al., 2025; Zhang et al., 2026b).

One way to adapt a VLM into a navigation policy is to retain the autoregressive (AR) decoding mechanism used in standard language and VLMs (Bai et al., 2025a; Grattafiori et al., 2024) (Figure 1 (a)). Under this decoding scheme, an action chunk is serialized and decoded from left to right. Each later action is therefore conditioned on earlier predictions from the same chunk, which we term *generated-action-prefix-conditioned generation*. VLM-based control is an active research direction, spanning vision-language-action (VLA) models for robotic manipulation (Zitkovich et al., 2023; O'Neill et al., 2024; Kim et al., 2025; Pertsch et al., 2025) and action or trajectory prediction for navigation (Hirose et al., 2025; Zhang et al., 2026b). Despite this progress, how generated-action-prefix conditioning affects closed-loop navigation remains less clearly characterized.

We therefore begin by evaluating recent AR-based VLMs (Bai et al., 2025a) as action-generating navigation policies. Our analysis yields two empirical observations. First, the evaluated AR policies achieve lower closed-loop success than smaller specialist navigation policies (Figure 1 (b), top). Second, we observe within-chunk error propagation when generated actions are reused as prefixes (Figure 1 (b), bottom). To probe this feedback pathway, we replace model-generated prefixes with ground-truth actions during open-loop evaluation, keeping each trained policy fixed. This intervention reduces later-position errors, with larger reductions under distribution shift (Section 3). These

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.



(a) Autoregressive generation. (b) Generated-prefix analysis. (c) Diffusion-based generation.

Figure 1: Generated action prefixes in VLM navigation. (a) AR VLMs condition later actions on previously generated actions within the same target chunk, causing early errors to propagate. (b) The evaluated AR policies show lower closed-loop success than specialist policies, while replacing generated prefixes with ground-truth prefixes reduces later-position error. (c) LLaDA-Nav avoids generated-prefix conditioning by predicting the full action block from masked target tokens.

results indicate that errors in generated prefixes can affect subsequent action predictions within the same chunk. These findings motivate us to eliminate this within-chunk feedback pathway through *generated-action-prefix-free* generation.

To this end, masked diffusion VLMs (DLMs) offer a suitable starting point because their masked-token interface supports parallel prediction without requiring a generated left-to-right prefix (Yang et al., 2025). However, standard iterative decoding of DLMs still reveals target tokens and feeds recovered tokens back to predict the remaining positions. We therefore propose LLaDA-Nav, which adapts LLaDA-V (You et al., 2026a) to visual navigation without within-chunk target-action reuse.

During both training and inference, LLaDA-Nav appends a fully masked action block to the input, using one mask token for each tokenized component of the target action chunk. From this fully masked input, it predicts all action tokens simultaneously in a single forward pass, yielding generated-action-prefix-free action-chunk generation. To preserve short-term motion context without relying on target predictions, LLaDA-Nav also conditions on recently executed actions available at deployment. Extensive experiments across four navigation datasets and unseen HM3D environments demonstrate that LLaDA-Nav outperforms state-of-the-art baselines.

- We formulate and test a focused failure hypothesis for autoregressive VLM navigation: generated-action-prefix-conditioned action chunks are vulnerable to within-chunk error propagation, which may contribute to closed-loop degradation after execution.
- We propose LLaDA-Nav, which adapts a pretrained diffusion VLM to visual navigation through Prefix-Free Action Block Modeling (PAM) and Action Context Conditioning (ACC). PAM removes within-chunk action reuse via fully masked, single-pass action prediction, while ACC preserves action context using previously executed actions.
- We conduct extensive open-loop and closed-loop evaluations, demonstrating substantial performance improvements over the evaluated autoregressive VLM baselines and quantifying the contribution of deployable action context through ablations.

2 Related Work

Visual navigation. Goal-conditioned visual navigation has progressed from navigation-specific policies toward world models and general-purpose multimodal models. Early approaches learn goal-conditioned control from diverse real-world navigation experience (Shah et al., 2022), while subsequent methods improve cross-robot scalability (Shah et al., 2023a), temporal context modeling (Shah et al., 2023b), and diffusion-based action generation (Sridhar et al., 2024). Recent VLM-based approaches incorporate pretrained semantic knowledge at different levels of the navigation stack. GeniNav (Chen et al., 2026) and SCOPE (Wang et al., 2026) use VLMs for high-level subgoal imagination and exploration guidance, whereas OmniVLA (Hirose et al., 2025) and NavFoM (Zhang et al., 2026b) move toward direct action or trajectory prediction. In parallel, world-model approaches exploit imagined future observations (Bar et al., 2025), memory-augmented planning (Dong et al.,

2025), and diverse generative models as world models (Zhang et al., 2026a). Our work instead isolates the action-decoding interface and examines how generated-action-prefix conditioning affects the stability of VLM-generated action chunks.

Autoregressive action tokens and exposure bias. RT-2 (Zitkovich et al., 2023) and OpenVLA (Kim et al., 2025) show that robot actions can be represented within a VLM's token-generation interface. AR factorization gives each later token access to earlier actions, which can capture dependencies within a chunk. During standard AR action-token training, each prediction is conditioned on ground-truth preceding tokens, whereas inference uses model-generated tokens. This discrepancy is a form of *exposure bias* (Ranzato et al., 2015). In this setting, an erroneous generated action can influence later predictions through the AR feedback path.

However, prior language-generation studies show that such error propagation need not produce monotonic positional degradation (Wu et al., 2018; He et al., 2021). We therefore use within-chunk error growth only for the trend measured in our navigation experiments. Our structural claim is deliberately narrower than a general criticism of AR: we examine generated-action-prefix-conditioned action chunks in the tested navigation settings.

Large diffusion language models. Large diffusion language models (DLMs) replace left-to-right factorization with iterative masked-token prediction. LLaDA (Nie et al., 2025) establishes this formulation for language modeling, and LLaDA-V (You et al., 2026a) and LaViDa (Li et al., 2025) extend this formulation to visual-language understanding. MMaDA (Yang et al., 2025) and LLaDA-o (You et al., 2026b) further explore image generation. Recent works have also adapted diffusion-based language or multimodal models to robot control. LLaDA-VLA (Wen et al., 2025) predicts tokenized robot actions using a pretrained DLM, while Discrete Diffusion VLA (Liang et al., 2026) generates discretized action chunks through iterative discrete denoising. While DLM has shown promise for action generation in manipulation, its potential for visual navigation remains underexplored. Our method addresses this gap through fully masked action-block training and single-pass decoding without within-chunk target-action reuse.

3 Preliminaries and Analysis

Image-goal navigation. An image-goal navigation policy predicts an action chunk for the next h steps ($\boldsymbol{a}_{0:h-1}$) from the current observation ($\boldsymbol{I}_o$) and a goal image ($\boldsymbol{I}_g$). Each $\boldsymbol{a}_i$ is a per-step body-frame pose increment defined as $\boldsymbol{a}_i = (a_i^x, a_i^y, a_i^\phi)$. The policy is evaluated under two settings: open-loop evaluation assesses such one-shot action generation, while closed-loop evaluation assesses whether the agent completes the navigation episode in a simulator.

Autoregressive VLMs. Given a multimodal condition $\boldsymbol{c}$ composed of image-text inputs and an output sequence $\boldsymbol{x}_0 = (x_0^1, \ldots, x_0^N)$, an AR VLM factorizes the conditional sequence distribution from left to right (Figure 1(a)):

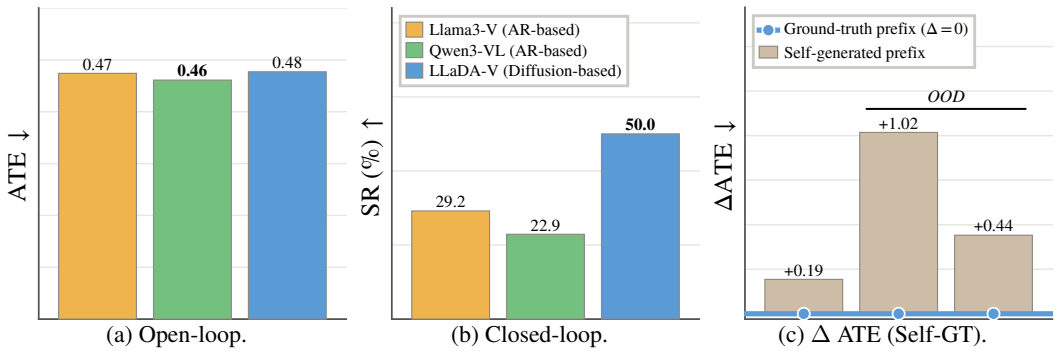
$$p_\theta^{\mathrm{AR}}(\boldsymbol{x}_0 \mid \boldsymbol{c}) = \prod_{i=1}^{N} p_\theta\left(x_0^i \mid \boldsymbol{x}_0^{<i}, \boldsymbol{c}\right), \tag{1}$$

where $\boldsymbol{x}_0^{<i} = (x_0^1, \ldots, x_0^{i-1})$ denotes the preceding token prefix. Accordingly, each token is generated sequentially conditioned on $\boldsymbol{c}$ and previously generated tokens.

Masked diffusion language models. Under the same multimodal condition $\boldsymbol{c}$ and target sequence $\boldsymbol{x}_0$, a masked DLM replaces left-to-right factorization with masked-token prediction. A timestep $t \in (0, 1]$ is sampled, and each target token is independently replaced with a special mask token $[\mathrm{M}]$ with probability t, yielding a masked sequence $\boldsymbol{x}_t$. The model is trained to recover the original tokens at the masked positions. The training objective ($\mathcal{L}_{\mathrm{DLM}}$) is defined as:

$$\mathcal{L}_{\mathrm{DLM}} = -\mathbb{E}_{t, \boldsymbol{x}_0, \boldsymbol{x}_t, \boldsymbol{c}}\left[\frac{1}{t}\sum_{i=1}^{N} \mathbf{1}[x_t^i = [\mathrm{M}]] \log p_\theta\left(x_0^i \mid \boldsymbol{x}_t, \boldsymbol{c}\right)\right]. \tag{2}$$

At inference, generation begins from a fully masked sequence. At each iteration, the model predicts multiple masked positions in parallel, retains a subset of the recovered tokens, and feeds the



(a) Open-loop. (b) Closed-loop. (c) Δ ATE (Self-GT).

Figure 2: VLM policy comparison and fixed-backbone prefix intervention. (a) Llama3-V, Qwen3-VL, and LLaDA-V achieve comparable in-distribution ATE on RECON. (b) Their closed-loop success rates differ substantially, with masked-diffusion LLaDA-V outperforming the evaluated AR policies. (c) In an AR-style LLaDA-V variant, self-generated prefixes yield higher ATE than ground-truth prefixes on RECON, SCAND, and HuRoN (left to right), with larger gaps on the OOD datasets.

partially recovered sequence back to predict the remaining positions. Although DLM avoids a fixed left-to-right order, its standard iterative decoding still reuses recovered target tokens as context for predicting the remaining masked positions.

3.1 Diagnosing Generated-Action-Prefix Conditioning

As discussed in Section 1, we investigate whether general-purpose VLMs can serve as navigation policies. We adapt Llama3-V and Qwen3-VL-8B-Instruct (Bai et al., 2025a) using standard AR action generation. For LLaDA-V (You et al., 2026a), we use its masked-token interface for prefix-free action generation: the target action block is fully masked during training, and all action tokens are predicted simultaneously in a single forward pass at inference. All policies are trained on the same RECON data using the same action representation and prediction horizon.

As shown in Figure 2 (a), the three policies achieve comparable absolute trajectory error (ATE) on in-distribution RECON. However, their closed-loop performance differs substantially (Figure 2 (b)). The prefix-free LLaDA-V policy achieves 50.0% success, compared with 29.2% for Llama3-V and 22.9% for Qwen3-VL. The two AR policies also underperform smaller specialist navigation policies (Figure 1 (b)). Because these policies differ in their pretrained backbones, we next examine generated-action-prefix conditioning through within-model interventions.

First, we evaluate each AR policy in open-loop under two prefix conditions. Under *Standard AR*, each action is conditioned on previously generated actions within the same target chunk. Under *Oracle AR*, we replace the generated prefix with the corresponding ground-truth actions (teacher forcing), keeping the trained model fixed. In the representative Llama3-V results in Figure 1 (b), Standard AR exhibits a net increase in error from the first to the final action position, whereas Oracle AR yields lower errors from the second position onward. The final-position gap is 0.06 on RECON and increases to 0.51 on out-of-distribution (OOD) HuRoN.

Second, we test whether this sensitivity also arises in a diffusion-pretrained backbone under AR-style generation. We train a separate AR-style LLaDA-V variant on RECON, providing preceding ground-truth actions during training while keeping the remaining target positions masked. After training, we evaluate this model under the same Standard AR and Oracle AR conditions, changing only the prefix source. As shown in Figure 2 (c), self-generated prefixes yield higher ATE than ground-truth prefixes by 0.19 on RECON, 1.02 on SCAND, and 0.44 on HuRoN. Section B provides the complete results across all four datasets.

Together, these interventions show that sensitivity to generated action prefixes is not specific to AR-pretrained backbones: it also occurs in LLaDA-V under AR-style generation. This shared sensitivity points to generated-prefix feedback as a vulnerability of AR-style action generation, with larger error gaps under distribution shift. Building on these findings, LLaDA-Nav fully masks the target action block and predicts all action tokens in a single forward pass.

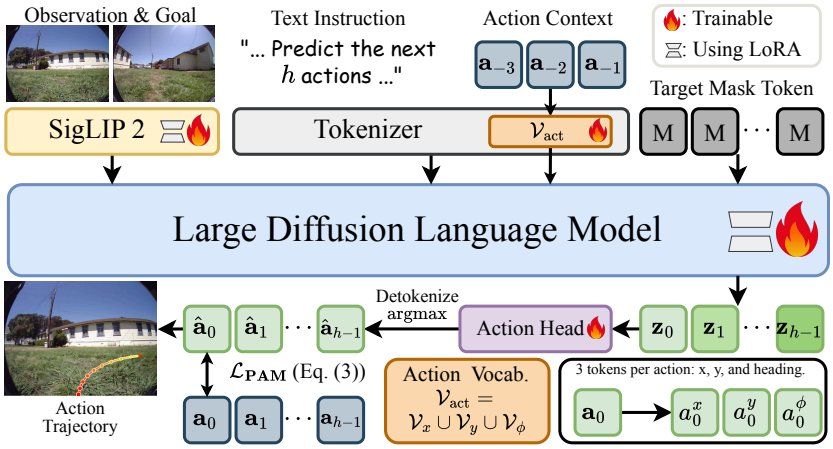


Figure 3: Overview of LLaDA-Nav. The model receives the current observation, goal image, and text instruction, together with recent executed actions supplied by ACC. PAM predicts all target action tokens from a fully masked block in a single forward pass. The predicted tokens are detokenized into body-frame pose increments and integrated into an h-step trajectory. Each displayed group z_j represents the component tokens (z_j^x, z_j^y, z_j^ϕ).

4 LLaDA-Nav: Diffusion Language Model for Navigation

Figure 3 illustrates the overall framework of LLaDA-Nav. Prefix-Free Action Block Modeling (PAM) predicts a fully masked action block in a single forward pass, while Action Context Conditioning (ACC) provides context from previously executed actions. We describe the details of our method in the following sections.

4.1 Architecture and Action Tokenization

Architecture. LLaDA-Nav builds on LLaDA-V (You et al., 2026a), which consists of a vision encoder, a multilayer perceptron (MLP) projector, and the masked diffusion language model LLaDA-8B-Instruct (Nie et al., 2025). We retain this pretrained multimodal backbone and adapt its masked-token interface to fixed-horizon action prediction.

Input construction. At each decision step, the base multimodal prompt $\boldsymbol{c}$ contains the current observation $\boldsymbol{I}_o$, the goal image $\boldsymbol{I}_g$, and a predefined text instruction. The two RGB images are encoded by the SigLIP 2 vision encoder (Tschannen et al., 2025), projected into the language-model embedding space, and presented in observation–goal order. The instruction specifies the image-goal navigation objective and the output format of an h-step body-frame action chunk. Appending a fully masked target block $\boldsymbol{M}$ to $\boldsymbol{c}$ yields the base model input $[\boldsymbol{c}, \boldsymbol{M}]$. Complete prompt templates and input examples are provided in Section D.

Action tokenization. Following prior work (Wen et al., 2025), we represent continuous robot actions using dedicated tokens in the language-model vocabulary. We normalize the translational components x and y and uniformly quantize each into $N_{\text{act,trans}}$ bins over the same normalized range. The heading increment is uniformly quantized over $[-\pi, \pi]$ into $N_{\text{act,head}}$ bins. We assign a distinct token to each quantization bin and extend the original LLaDA-V vocabulary $\mathcal{V}$ with the action vocabulary $\mathcal{V}_{\text{act}} = \mathcal{V}_x \cup \mathcal{V}_y \cup \mathcal{V}_\phi$. For each action $\boldsymbol{a}_j = (a_j^x, a_j^y, a_j^\phi)$, we denote its three discrete tokens as (z_j^x, z_j^y, z_j^ϕ), where $z_j^x \in \mathcal{V}_x$, $z_j^y \in \mathcal{V}_y$, and $z_j^\phi \in \mathcal{V}_\phi$. An h-step action chunk is therefore represented as a sequence of $3h$ action tokens. After detokenization, we rescale the translational components to physical units and integrate the resulting body-frame pose increments to obtain the predicted trajectory.

4.2 PREFIX-FREE ACTION BLOCK MODELING

The analysis in Section 3 reveals sensitivity to generated action prefixes. Although masked diffusion VLMs avoid a fixed left-to-right factorization, their standard iterative decoding also reuses recovered target tokens as context, retaining a potential pathway for within-chunk error propagation.

We introduce *Prefix-Free Action Block Modeling* (PAM), which excludes target action tokens from the conditioning context during both training and inference. Let $\boldsymbol{z} = (z_0^x, z_0^y, z_0^\phi, \ldots, z_{h-1}^x, z_{h-1}^y, z_{h-1}^\phi)$ denote the flattened sequence of $3h$ action tokens defined in the previous section. PAM uses the fully masked endpoint $t = 1$ of Equation (2), with $\boldsymbol{M} = [\mathrm{M}]^{3h}$.

To describe PAM independently of ACC, we first use the base multimodal condition $\boldsymbol{c}$. The complete model replaces $\boldsymbol{c}$ with $\boldsymbol{c}_{\mathrm{acc}}$, introduced in the following section. PAM minimizes the average cross-entropy over all target action positions, with the output distribution restricted to $\mathcal{V}_{\mathrm{act}}$:

$$\mathcal{L}_{\mathrm{PAM}} = -\mathbb{E}_{(\boldsymbol{c},\boldsymbol{z})\sim\mathcal{D}}\left[\frac{1}{3h}\sum_{i=0}^{3h-1}\log p_\theta^{\mathcal{V}_{\mathrm{act}}}\left(\boldsymbol{z}_i \mid \boldsymbol{c}, \boldsymbol{M}\right)\right], \tag{3}$$

where $\mathcal{D}$ is the training dataset and $p_\theta^{\mathcal{V}_{\mathrm{act}}}$ is the categorical distribution over the action vocabulary. Compared with Equation (2), all target positions are active in the loss because the complete action block is masked.

At inference, PAM predicts all $3h$ tokens in a single forward pass from $[\boldsymbol{c}, \boldsymbol{M}]$, selecting each token from its component-specific vocabulary:

$$\hat{z}_j^d = \arg\max_{z\in\mathcal{V}_d} p_{\theta,j,d}\left(z \mid \boldsymbol{c}, \boldsymbol{M}\right), \quad j = 0, \ldots, h-1,\ d \in \{x, y, \phi\}, \tag{4}$$

where $p_{\theta,j,d}$ denotes the action-vocabulary distribution at the slot for component d of step j.

The predicted tokens are grouped into $(\hat{z}_j^x, \hat{z}_j^y, \hat{z}_j^\phi)$ and detokenized into the continuous action $\hat{\boldsymbol{a}}_j$. Unlike standard masked-diffusion decoding, PAM performs no iterative refinement and feeds no predicted target token back to the model.

4.3 ACTION CONTEXT CONDITIONING

PAM removes conditioning on other actions within the current target chunk. However, recent motion history may help maintain continuity across successive action chunks. We therefore introduce *Action Context Conditioning* (ACC), which provides this history through the k most recently executed actions:

$$\boldsymbol{a}_{-k:-1} = \left(\boldsymbol{a}_{-k}, \ldots, \boldsymbol{a}_{-1}\right), \tag{5}$$

where $\boldsymbol{a}_{-1}$ denotes the most recently executed action.

Each executed action is computed from the measured pose change and retained in its original per-step body frame. We apply the same translational normalization and component-wise tokenization used for target actions. The resulting context tokens are ordered from oldest to newest. When fewer than k executed actions are available, missing history entries are filled with tokens encoding zero motion.

ACC appends the tokenized action history to the base multimodal condition:

$$\boldsymbol{c}_{\mathrm{acc}} = \left[\boldsymbol{c},\ \tau_{\mathcal{V}_{\mathrm{act}}}(\boldsymbol{a}_{-k:-1})\right], \tag{6}$$

where $\tau_{\mathcal{V}_{\mathrm{act}}}(\cdot)$ maps the action history to $3k$ action tokens. These context tokens remain unmasked and precede the fully masked target block.

The complete model uses $\boldsymbol{c}_{\mathrm{acc}}$ in place of $\boldsymbol{c}$ in both the training objective and inference rule (Equations (3) and (4)). Thus, predictions are conditioned on executed-action history, while no predicted token from the current target chunk is reused as input.

4.4 PARAMETER-EFFICIENT ADAPTATION

We adapt LLaDA-Nav using Low-Rank Adaptation (LoRA) (Hu et al., 2022), keeping the original LLaDA-V parameters frozen. For pretrained $\mathbf{W}_0 \in \mathbb{R}^{d_{\mathrm{out}}\times d_{\mathrm{in}}}$, the adapted weight is

$$\mathbf{W} = \mathbf{W}_0 + \frac{\alpha}{r}\mathbf{B}\mathbf{A}, \tag{7}$$

Table 1: Open-loop trajectory prediction. Absolute trajectory error (ATE) and relative pose error (RPE) on held-out trajectory segments from four navigation datasets; lower is better. The three best distinct values within each horizon are shaded green, light green, and yellow, respectively; ties receive the same color, and the best values are also in **bold**. Methods are grouped by their original action horizon $h \in \{5, 8\}$, while LLaDA-Nav is evaluated at both.

h	Model	RECON		SCAND		HuRoN		TartanDrive		Average	
		ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓
5	Forward	0.96	0.41	1.93	0.62	2.63	0.97	3.75	1.17	2.32	0.79
	GNM	0.52	0.20	1.43	0.47	1.20	0.53	1.17	0.42	1.08	0.40
	ViNT	0.53	0.20	1.53	0.49	1.08	**0.50**	1.54	0.52	1.17	0.43
	LLaDA-Nav (Ours)	**0.37**	**0.19**	**0.61**	**0.29**	**1.00**	0.51	**0.73**	**0.39**	**0.68**	**0.34**
8	Forward	1.92	0.54	2.97	0.62	4.35	1.06	5.75	1.19	3.75	0.85
	NoMaD	1.43	0.37	2.75	0.58	2.20	0.63	2.32	0.56	2.18	0.54
	OmniVLA	3.04	0.69	1.44	0.37	2.92	0.77	9.41	1.90	4.20	0.93
	NWM	1.37	0.35	**1.08**	**0.28**	4.96	1.08	5.54	1.16	3.24	0.72
	LLaDA-Nav (Ours)	**0.73**	**0.25**	1.31	0.41	**1.71**	**0.58**	**1.68**	**0.53**	**1.36**	**0.44**

where $\mathbf{A} \in \mathbb{R}^{r \times d_{\text{in}}}$ and $\mathbf{B} \in \mathbb{R}^{d_{\text{out}} \times r}$ are trainable matrices, r is the adaptation rank, and α controls the update scale. We initialize $\mathbf{A}$ using Kaiming initialization and $\mathbf{B}$ to zero.

As illustrated in Figure 3, LoRA is applied to the attention and feed-forward projections in every transformer layer of the diffusion language model, and to the query, key, and value projections of the SigLIP 2 vision encoder. The multimodal projector is kept frozen.

We additionally train the embeddings of the new action tokens and their corresponding rows in the extended language-model output head. Each new row is initialized to the mean of the pretrained vocabulary rows in its respective matrix, while the original vocabulary rows remain frozen. The action head in Figure 3 consists of these output rows, with no additional decoder.

5 Experiments

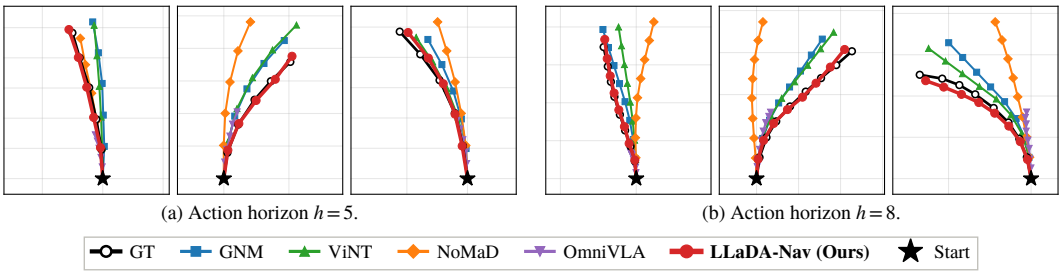
5.1 Experimental Setup

Open-loop trajectory prediction. Following NWM (Bar et al., 2025), we evaluate open-loop trajectory prediction on 100 held-out segments per dataset at horizons $h \in \{5, 8\}$. We report absolute trajectory error (ATE) and relative pose error (RPE), both measured in meters. Evaluation-set construction, horizon-specific goal selection, and metric definitions are provided in Section A.2.

Closed-loop evaluation. Following WIW (Zhang et al., 2026a), we evaluate 144 image-goal navigation episodes across 87 unseen HM3D scenes (Ramakrishnan et al., 2021) in Habitat-Sim (Savva et al., 2019). The first three actions of each predicted chunk are executed before replanning. We report SR, SPL (Anderson et al., 2018), and the mean number of executed actions; success criteria and the complete protocol are given in Section A.3.

Datasets. We train LLaDA-Nav on RECON (Shah et al., 2022), SCAND (Karnan et al., 2022), HuRoN (Hirose et al., 2024), and TartanDrive (Triest et al., 2022). We use the NWM splits except for HuRoN, for which we follow the NoMaD splitting protocol; details are provided in Section A.2. The RECON-only diagnostic experiments in Section 3 are reported separately from this four-dataset evaluation.

Baselines. We compare LLaDA-Nav against specialist navigation policies, including GNM (Shah et al., 2023a), ViNT (Shah et al., 2023b), and NoMaD (Sridhar et al., 2024), as well as the world-model-based NWM (Bar et al., 2025) and the VLM-based OmniVLA (Hirose et al., 2025). For open-loop evaluation, each baseline uses its native prediction horizon. For closed-loop evaluation, we additionally include the Qwen2.5-VL-72B base policy (Bai et al., 2025b) and the world-model-assisted configurations evaluated in WIW. Policy configurations and the action-proposal setup are detailed in Section A.3.



(a) Action horizon $h = 5$. (b) Action horizon $h = 8$.

Figure 4: Open-loop trajectory visualization on RECON. Representative trajectory predictions at action horizons $h = 5$ and $h = 8$. Black denotes the ground-truth trajectory, red denotes LLaDA-Nav, and the remaining colors denote baseline predictions. The star indicates the trajectory origin, and the circles represent successive predicted waypoints.

Table 3: Component ablation. PAM and ACC on open- and closed-loop evaluation.

PAM	ACC	Open-loop		Closed-loop	
		ATE ↓	RPE ↓	SR (%) ↑	SPL (%) ↑
		1.17	0.54	36.81	29.65
✓		1.13	0.49	44.44	36.64
	✓	1.23	0.58	38.89	30.48
✓	✓	**0.68**	**0.34**	**59.03**	**50.15**

Table 4: Masking and decoding ($h = 5$, $k = 3$). T denotes forward passes.

Mask	T	Average		Closed-loop	
		ATE ↓	RPE ↓	SR (%) ↑	SPL (%) ↑
Random	1	1.23	0.58	38.89	30.48
Random	5	**0.67**	**0.30**	38.89	33.03
PAM	1	0.68	0.34	**59.03**	50.15
PAM	5	**0.67**	0.31	58.33	**51.70**

Implementation details. We use 80 action bins per translational axis and 64 heading bins, LoRA rank $r = 64$, and ACC length $k = 3$, with $h = 5$ by default. The model has approximately 181M trainable parameters. Full architecture and optimization settings are listed in Table A1.

5.2 Experimental Results

Open-loop trajectory prediction. As shown in Table 1, LLaDA-Nav achieves the lowest average ATE and RPE among the compared methods at both action horizons. At $h = 5$, it reduces ATE and RPE relative to GNM, the strongest baseline at this horizon, by 37.0% and 15.0%, respectively. At $h = 8$, the corresponding reductions over NoMaD are 37.6% and 18.5%. Figure 4 illustrates representative predicted trajectories on RECON; additional examples across all four datasets and visualization details are provided in Section E.1.

Closed-loop evaluation. As shown in Table 2, LLaDA-Nav achieves 59.03% SR, 50.15% SPL, and an average of 41.5executed actions per episode. It improves SR by 15.28 percentage points over NWM, the strongest baseline in success rate, and SPL by 13.54 percentage points over ViNT, the strongest baseline in this metric. The higher SR and SPL show that the full policy improves both task completion and path efficiency when repeatedly executed and replanned in unseen environments. Representative closed-loop rollouts are provided in Section E.2.

Table 2: Closed-loop evaluation results. LLaDA-Nav substantially outperforms all baselines.

Methods	SR (%) ↑	Mean traj. ↓	SPL (%) ↑
Base policy	36.11	48.8	23.97
Hunyuan-13B	36.11	46.8	26.89
Wan2.2-A14B	43.05	45.8	31.46
NWM	43.75	45.0	32.62
GNM	35.42	49.1	31.33
ViNT	42.36	47.3	36.61
NoMaD	32.64	50.9	27.77
OmniVLA	36.11	47.5	34.42
NWM+NoMaD	27.78	53.8	22.08
LLaDA-Nav (Ours)	**59.03**	**41.5**	**50.15**

5.3 Ablation Studies

Table 3 evaluates the individual and combined effects of PAM and ACC, with single-pass decoding fixed across all settings. When PAM is disabled, we use random-mask training. PAM alone modestly

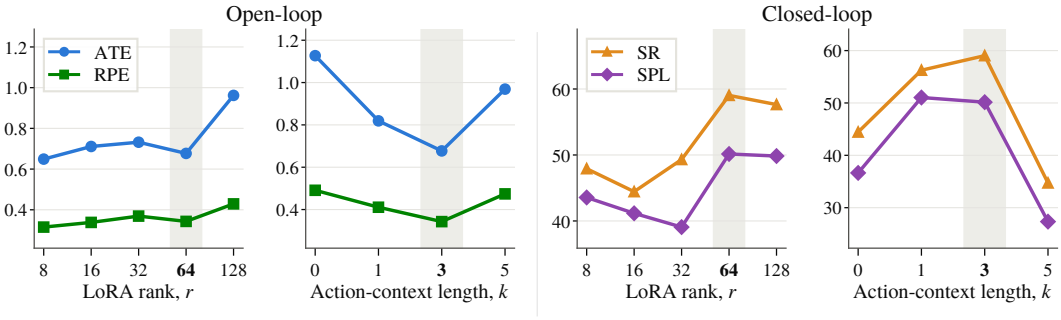


Figure 5: Hyperparameter ablations. Shading indicates our defaults, $r = 64$ and $k = 3$. The left panels show open-loop results, and the right panels show closed-loop results.

improves open-loop prediction, reducing ATE from 1.17 to 1.13 and RPE from 0.54 to 0.49, while increasing closed-loop SR from 36.81% to 44.44%. ACC alone yields a smaller SR gain, reaching 38.89%, while slightly degrading open-loop prediction. Combining the two components achieves the lowest open-loop errors and highest SR and SPL in this ablation.

Adding ACC to PAM improves SR by 14.59 percentage points, compared with the 2.08-point gain from adding ACC to the baseline. This difference highlights the complementary effects of the training objective and action history. ACC encodes motion history through trainable action-token embeddings. With ACC enabled, full-mask training requires prediction from the visual-text condition and executed-action history, without access to ground-truth target tokens. These results suggest that full-mask training may help the model learn and use these representations more effectively, contributing to the larger combined gains. Additional results and visualizations appear in Section C.3.

Masking and decoding. Table 4 compares random-mask and PAM's full-mask training under single-pass ($T = 1$) and iterative ($T = 5$) decoding, with ACC fixed at $k = 3$. For random-mask training, iterative decoding reduces average ATE from 1.23 to 0.67, but SR remains at 38.89%, showing that improved trajectory prediction does not necessarily yield higher navigation success. With full-mask training, one and five passes achieve similar SR (59.03% and 58.33%), while iterative decoding improves SPL from 50.15% to 51.70%. Thus, full-mask training supports strong closed-loop performance under both decoders, and additional refinement offers mixed gains at a higher forward-pass cost. We use single-pass decoding by default. Per-dataset results and refinement-step sensitivity are provided in Table A4 and Section C.2.

Hyperparameter sensitivity. Figure 5 studies LoRA rank r and ACC length k. Among the tested ranks, $r = 64$ yields the highest SR and SPL, while $r = 8$ gives the lowest open-loop errors. For ACC, $k = 3$ achieves the lowest open-loop errors and highest SR, while $k = 1$ yields the highest SPL. We use $r = 64$ and $k = 3$ by default; detailed results are in Tables A6 and A7.

Additional results with MMaDA are provided in Section C.5.

6 Conclusion

We studied how action decoding affects direct VLM-based visual navigation. Our analysis shows that reusing generated actions within a chunk can propagate early errors to later predictions, especially under distribution shift. Motivated by this finding, we proposed LLaDA-Nav, which combines Prefix-Free Action Block Modeling with Action Context Conditioning. LLaDA-Nav predicts a fully masked action chunk in a single forward pass while preserving short-term action information through previously executed actions. Across four navigation datasets and unseen HM3D environments, LLaDA-Nav demonstrates substantial improvements in both open-loop trajectory prediction and closed-loop navigation over the evaluated baselines. Our ablations show complementary benefits of full-mask training and executed-action conditioning, with strong closed-loop performance achieved through single-pass prediction. These results suggest that prefix-free parallel action generation provides a simple and effective interface for robust VLM-based navigation.

AI USE STATEMENT

We used generative AI tools to polish the language and improve the clarity and readability of the author-written manuscript. All suggested edits were reviewed by the authors, who take full responsibility for the final content.

ETHICS STATEMENT

This work uses existing navigation datasets, and closed-loop evaluation is conducted exclusively in simulation. Any future deployment on physical robots would require additional safety validation, particularly for operation near people and obstacles.

REPRODUCIBILITY STATEMENT

We describe the training configuration, dataset splits, evaluation protocols, and metric definitions in Section A, with hyperparameters summarized in Table A1. Prompt construction and examples are provided in Section D, and detailed ablation results are reported in Section C.

REFERENCES

Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, and Amir R. Zamir. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018. URL https://arxiv.org/abs/1807.06757.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025a. URL https://arxiv.org/abs/2511.21631.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025b. URL https://arxiv.org/abs/2502.13923.

Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15791–15801, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Bar_Navigation_World_Models_CVPR_2025_paper.html.

Yuqi Chen, Junjie Gao, Yongzhou Pan, Siyuan Song, Zixuan Zhang, Jiaping Xiao, and Mir Feroskhan. GeniNav: Generative model driven image-goal navigation via imagination-guided consistency flow matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 996–1005, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Chen_GeniNav_Generative_Model_Driven_Image-Goal_Navigation_via_Imagination-Guided_Consistency_Flow_CVPR_2026_paper.html.

Yifei Dong, Fengyi Wu, Guangyu Chen, Zhi-Qi Cheng, Qiyu Hu, Yuxuan Zhou, Jingdong Sun, Jun-Yan He, Qi Dai, and Alexander G Hauptmann. Unified world models: Memory-augmented planning and foresight for visual navigation. *arXiv preprint arXiv:2510.08713*, 2025.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Tianxing He, Jingzhao Zhang, Zhiming Zhou, and James Glass. Exposure bias versus self-recovery: Are distortions really incremental for autoregressive text generation? In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5087–5102, Online and Punta Cana, Dominican Republic, 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.415. URL https://aclanthology.org/2021.emnlp-main.415/.

Noriaki Hirose, Dhruv Shah, Ajay Sridhar, and Sergey Levine. SACSoN: Scalable autonomous control for social navigation. *IEEE Robotics and Automation Letters*, 9(1):49–56, 2024. doi: 10.1109/LRA.2023.3329626. URL https://arxiv.org/abs/2306.01874.

Noriaki Hirose, Catherine Glossop, Dhruv Shah, and Sergey Levine. OmniVLA: An omni-modal vision-language-action model for robot navigation. *arXiv preprint arXiv:2509.19480*, 2025. URL https://arxiv.org/abs/2509.19480.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Haresh Karnan, Anirudh Nair, Xuesu Xiao, Garrett Warnell, Sören Pirk, Alexander Toshev, Justin Hart, Joydeep Biswas, and Peter Stone. Socially compliant navigation dataset (SCAND): A large-scale dataset of demonstrations for social navigation. *IEEE Robotics and Automation Letters*, 7(4):11807–11812, 2022. doi: 10.1109/LRA.2022.3184025. URL https://arxiv.org/abs/2203.15041.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P. Foster, Pannag R. Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model. In *Proceedings of the 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025. URL https://proceedings.mlr.press/v270/kim25c.html.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.

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, 2025.

Zhixuan Liang, Yizhuo Li, Tianshuo Yang, Chengyue Wu, Sitong Mao, Liuao Pei, Tian Nian, Shunbo Zhou, Xiaokang Yang, Jiangmiao Pang, Yao Mu, and Ping Luo. Discrete diffusion VLA: Bringing discrete diffusion to action decoding in vision-language-action policies. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=c3BVcHcSiR.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large language diffusion models. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/48b383b24230e0e6e649d9c98dae4d8c-Abstract-Conference.html.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.

Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John M Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. Habitat-matterport 3d dataset (HM3d): 1000 large-scale 3d environments for embodied AI. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL https://openreview.net/forum?id=-v4OuqNs5P.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. In *International Conference on Learning Representations*, 2015. URL https://arxiv.org/abs/1511.06732.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9339–9347, 2019.

Dhruv Shah, Benjamin Eysenbach, Nicholas Rhinehart, and Sergey Levine. Rapid exploration for open-world navigation with latent goal models. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 674–684. PMLR, 2022. URL https://proceedings.mlr.press/v164/shah22a.html.

Dhruv Shah, Ajay Sridhar, Arjun Bhorkar, Noriaki Hirose, and Sergey Levine. GNM: A general navigation model to drive any robot. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7226–7233. IEEE, 2023a. doi: 10.1109/ICRA48891.2023.10161227. URL https://arxiv.org/abs/2210.03370.

Dhruv Shah, Ajay Sridhar, Nitish Dashora, Kyle Stachowicz, Kevin Black, Noriaki Hirose, and Sergey Levine. ViNT: A foundation model for visual navigation. In *Proceedings of the 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 711–733. PMLR, 2023b. URL https://proceedings.mlr.press/v229/shah23a.html.

Ajay Sridhar, Dhruv Shah, Catherine Glossop, and Sergey Levine. NoMaD: Goal masked diffusion policies for navigation and exploration. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 63–70. IEEE, 2024. doi: 10.1109/ICRA57147.2024.10610665. URL https://arxiv.org/abs/2310.07896.

Samuel Triest, Matthew Sivaprakasam, Sean J. Wang, Wenshan Wang, Aaron M. Johnson, and Sebastian Scherer. TartanDrive: A large-scale dataset for learning off-road dynamics models. In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 2546–2552. IEEE, 2022. doi: 10.1109/ICRA46639.2022.9811648. URL https://arxiv.org/abs/2205.01791.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, et al. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Ningnan Wang, Weihuang Chen, Liming Chen, Haoxuan Ji, Zhongyu Guo, Xuchong Zhang, and Hongbin Sun. Expand your SCOPE: Semantic cognition over potential-based exploration for embodied visual navigation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40 (22):18620–18628, 2026. doi: 10.1609/aaai.v40i22.38929. URL https://ojs.aaai.org/index.php/AAAI/article/view/38929.

Yuqing Wen, Hebei Li, Kefan Gu, Yucheng Zhao, Tiancai Wang, and Xiaoyan Sun. LLaDA-VLA: Vision language diffusion action models. *arXiv preprint arXiv:2509.06932*, 2025. URL https://arxiv.org/abs/2509.06932.

Lijun Wu, Xu Tan, Di He, Fei Tian, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. Beyond error propagation in neural machine translation: Characteristics of language also matter. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 3602–3611, Brussels, Belgium, 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1396. URL https://aclanthology.org/D18-1396/.

Ling Yang, Ye Tian, Bowen Li, Xinchen Zhang, Ke Shen, Yunhai Tong, and Mengdi Wang. Mmada: Multimodal large diffusion language models. *Advances in Neural Information Processing Systems*, 38:138867–138907, 2025.

Zebin You, Shen Nie, Xiaolu Zhang, Jun Zhou, Zhiwu Lu, Ji-Rong Wen, and Chongxuan Li. LLaDA-V: Large language diffusion models with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10093–10105, 2026a. URL https://openaccess.thecvf.com/content/CVPR2026/html/You_LLaDA-V_Large_Language_Diffusion_Models_with_Visual_Instruction_Tuning_CVPR_2026_paper.html.

Zebin You, Xiaolu Zhang, Jun Zhou, Chongxuan Li, and Ji-Rong Wen. Llada-o: An effective and length-adaptive omni diffusion model. *arXiv preprint arXiv:2603.01068*, 2026b.

Jiahan Zhang, Muqing Jiang, Nanru Dai, Taiming Lu, Arda Uzunoglu, Shunchi Zhang, Yana Wei, Jiahao Wang, Vishal M. Patel, Paul Pu Liang, Daniel Khashabi, Cheng Peng, Rama Chellappa, Tianmin Shu, Alan Yuille, Yilun Du, and Jieneng Chen. World-in-world: World models in a closed-loop world. In *International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=yDmb7xAfeb.

Jiazhao Zhang, Anqi Li, Yunpeng Qi, Minghan Li, Jiahang Liu, Shaoan Wang, Haoran Liu, Gengze Zhou, Yuze Wu, Xingxing Li, Yuxin Fan, Wenjun Li, Zhibo Chen, Fei Gao, Qi Wu, Zhizheng Zhang, and He Wang. Embodied navigation foundation model. In *International Conference on Learning Representations*, 2026b. URL https://openreview.net/forum?id=kkBOIsrCXh.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Proceedings of the 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183. PMLR, 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

This supplementary material provides implementation details, additional analyses, and qualitative results that complement the main paper. It is organized as follows:

- Section A details the training configuration, action representation and scaling, open- and closed-loop protocols, evaluation metrics, and Action Context Conditioning (ACC).
- Section B extends the generated-action-prefix analysis across autoregressive (AR) vision-language model (VLM) backbones and navigation datasets, including fixed-backbone interventions with LLaDA-V.
- Section C studies the interaction between training-time masking and inference-time decoding, iterative refinement, PAM and ACC, and the principal hyperparameters.
- Section D describes the multimodal prompt, including the navigation instruction and the executed-action context introduced by ACC.
- Section E presents qualitative open-loop trajectory predictions and closed-loop navigation rollouts.
- Section F discusses the scope of the framework and directions for future work.

A Experimental Details

Table A1 summarizes the model architecture, action representation, and training configuration used throughout our experiments. Unless otherwise specified, LLaDA-Nav predicts an action chunk with horizon $h = 5$, uses the $k = 3$ most recently executed actions for ACC, and is adapted with LoRA rank $r = 64$. The complete training and evaluation prompts are provided separately in Section D.

A.1 Training and Action Scaling

During training, trajectory segments are sampled with a frame stride $s \in \{1, 2, 3\}$ with probabilities $\{0.5, 0.3, 0.2\}$, respectively. Let $\boldsymbol{d}_i = (a_i^x, a_i^y)$ denote the translational displacement between two sampled frames. For dataset d, the displacement is normalized as

$$\boldsymbol{d}_i' = \frac{\boldsymbol{d}_i}{\alpha_d s}, \tag{A1}$$

where α_d is a dataset-specific normalization scale, following prior work (Shah et al., 2023a;b). We set α_d to 0.25, 0.38, 0.319, and 1.44 m for RECON (Shah et al., 2022), SCAND (Karnan et al., 2022), HuRoN (Hirose et al., 2024), and TartanDrive (Triest et al., 2022), respectively.

After token prediction, the normalized translation is mapped back to physical units by

$$\hat{\boldsymbol{d}}_i = \alpha_d s \, \hat{\boldsymbol{d}}_i', \tag{A2}$$

where s is 1 during evaluation. The heading component is treated separately and uniformly quantized over $[-\pi, \pi]$. The recovered translational and heading increments are subsequently integrated to form the predicted body-frame trajectory.

A.2 Open-Loop Trajectory Evaluation

Following Navigation World Models (NWM) (Bar et al., 2025), we evaluate one-shot trajectory prediction on RECON, SCAND, HuRoN, and TartanDrive. For RECON, SCAND, and TartanDrive, we directly adopt the train and test splits released by NWM. Because the HuRoN split used by NWM is private, we construct a random train–test split from the publicly available HuRoN data following the data-partitioning setting of NoMaD (Sridhar et al., 2024).

Forward motion is highly prevalent in these navigation datasets and can make randomly sampled evaluation trajectories disproportionately easy. Following NWM, we use a forward-only predictor to construct a hard evaluation set of 100 trajectories per dataset. The samples are selected to have forward-only prediction errors at a difficulty level comparable to that of the NWM evaluation set.

The original evaluation set is constructed with an action horizon of $h = 8$. The $h = 5$ evaluation is derived from the same underlying evaluation trajectories and starting observations as the $h = 8$

Table A1: Hyperparameters of LLaDA-Nav: model and action representation (top), training configuration (bottom). The domain-specific normalization scales α_d are ordered as RECON, SCAND, HuRoN, and TartanDrive.

Model and action representation		
Model	Backbone VLM	LLaDA-V
	Language model	LLaDA-8B
	Vision encoder	SigLIP 2
	Projector	2-layer MLP (GELU)
	Max sequence length	4096
Action	Action horizon h	5 (default; 8 variant)
	Action block	$3h$ tokens, fixed length
	Normalization param. α_d (m)	0.25 / 0.38 / 0.319 / 1.44
	$N_{\text{act,trans}}$	80
	$N_{\text{act,head}}$	64
	Added vocab rows	224
	Action-token row init.	Mean of pretrained vocab. rows
ACC	Context length k	3 executed actions
	Episode start	zero-motion padding
Training configuration		
LoRA	# of params	$\sim$181M (2.15%)
	LoRA r / α	64 / 128
	LoRA (LM, 32 layers)	q, k, v, o, gate, up, down
	LoRA (vision, 26 layers)	q, k, v
	LoRA init	B=0, Kaiming A
Optimization	Optimizer	AdamW (0.9, 0.999)
	LR / schedule	1×10^{-4} / cosine
	Warmup / wd	3% / 0
	Effective batch size	64
	Precision	bf16
	Training steps	20k ($h = 5$), 24k ($h = 8$)
Setup	Hardware	4$\times$ NVIDIA H200
	Training samples	611k ($h = 5$); 540k ($h = 8$) for 4 datasets
	Frame stride s (sampled)	$\{1, 2, 3\}$, ratio $\{0.5, 0.3, 0.2\}$

setting. For each sample, the goal image is taken from the frame five steps ahead, and the corresponding target trajectory consists of the first five actions. Thus, the two settings share the same trajectory origins but use horizon-specific goal images and action targets.

Given a current observation $\boldsymbol{I}_o$ and a future goal image $\boldsymbol{I}_g$, the policy predicts an h-step sequence of body-frame pose increments. The predicted translations are mapped back to physical units using Equation (A2) and integrated from the initial pose. We evaluate the recovered trajectory using ATE and RPE as defined below.

Metrics. Let $\hat{\boldsymbol{p}}_i \in \mathbb{R}^2$ and $\boldsymbol{p}_i \in \mathbb{R}^2$ denote the predicted and ground-truth positions at action step i, respectively, after integrating the body-frame action increments from the same initial pose. Following NWM (Bar et al., 2025), we compute Absolute Trajectory Error (ATE) as

$$\text{ATE} = \frac{1}{h}\sum_{i=1}^{h} \|\hat{\boldsymbol{p}}_i - \boldsymbol{p}_i\|_2 \,. \tag{A3}$$

ATE measures the average positional deviation over the complete predicted trajectory.

Relative Pose Error (RPE) compares the local displacement between consecutive trajectory points. Let $\Delta\hat{\boldsymbol{p}}_i = \hat{\boldsymbol{p}}_i - \hat{\boldsymbol{p}}_{i-1}$ and $\Delta\boldsymbol{p}_i = \boldsymbol{p}_i - \boldsymbol{p}_{i-1}$. We compute

$$\mathrm{RPE} = \frac{1}{h}\sum_{i=1}^{h} \left\| \Delta\hat{\boldsymbol{p}}_i - \Delta\boldsymbol{p}_i \right\|_2 . \tag{A4}$$

Lower ATE and RPE indicate better global trajectory accuracy and local motion consistency, respectively.

A.3 Closed-Loop Image-Goal Navigation

We conduct closed-loop evaluation using the ImageNav protocol of World-in-World (WIW) (Zhang et al., 2026a). The benchmark comprises 144 image-goal navigation episodes constructed from 87 validation scenes of HM3D (Ramakrishnan et al., 2021), which are executed in Habitat-Sim (Savva et al., 2019). Because HM3D is not included in our training datasets, all evaluated environments are unseen during training.

Each episode specifies the target using an RGB goal image associated with a position in the environment. At each decision step, the policy predicts an action sequence with horizon $h = 5$ from the current RGB observation and the goal image. Following WIW, the first three actions are executed in Habitat-Sim before replanning from the resulting observation. The procedure continues for at most 20 decision steps, with success defined as reaching within a geodesic distance of $0.5\,\mathrm{m}$ from the goal position.

For the world-model-assisted configurations reported by WIW, candidate action sequences are proposed by Qwen2.5-VL-72B-Instruct-AWQ (Bai et al., 2025b). This includes the base VLM policy and the configurations augmented with video or navigation world models, such as Wan (Wan et al., 2025), HunyuanVideo (Kong et al., 2024), and NWM (Bar et al., 2025). In contrast, GNM (Shah et al., 2023a), ViNT (Shah et al., 2023b), NoMaD (Sridhar et al., 2024), OmniVLA (Hirose et al., 2025), and LLaDA-Nav generate actions using their own learned navigation policies and do not use the Qwen proposal policy.

For LLaDA-Nav, ACC is constructed from the $k = 3$ most recently executed actions obtained from odometry. The actions retain their original per-step body-frame representation, are ordered from oldest to newest, and are inserted before the masked target token block. When fewer than three executed actions are available, the missing context positions are represented by zero-motion action tokens.

All policies are evaluated on the same WIW episodes with the same discrete action interface, success criterion, and termination procedure. This shared protocol does not imply identical policy inputs; we isolate the contribution of ACC through the matched $k = 0$ comparison in Table A6. We report success rate (SR), success weighted by path length (SPL) (Anderson et al., 2018), and mean trajectory length.

Metrics. For each episode i, success is determined by

$$S_i = \mathbf{1}\left[d_{\mathrm{geo}}\left(\boldsymbol{p}_i^{\mathrm{final}}, \boldsymbol{p}_i^{\mathrm{goal}} \right) \le r_i^{\mathrm{goal}} \right], \tag{A5}$$

where $d_{\mathrm{geo}}(\cdot,\cdot)$ is the geodesic distance and r_i^{goal} is the goal radius specified for episode i. We used the default goal radius of $0.5\,\mathrm{m}$ for all episodes, as specified by WIW.

For N evaluation episodes, Success Rate (SR) is

$$\mathrm{SR} = \frac{100}{N}\sum_{i=1}^{N} S_i. \tag{A6}$$

Success weighted by Path Length (SPL) is computed as

$$\mathrm{SPL} = \frac{100}{N}\sum_{i=1}^{N} S_i \frac{L_i^*}{\max\left(L_i, L_i^*\right)}, \tag{A7}$$

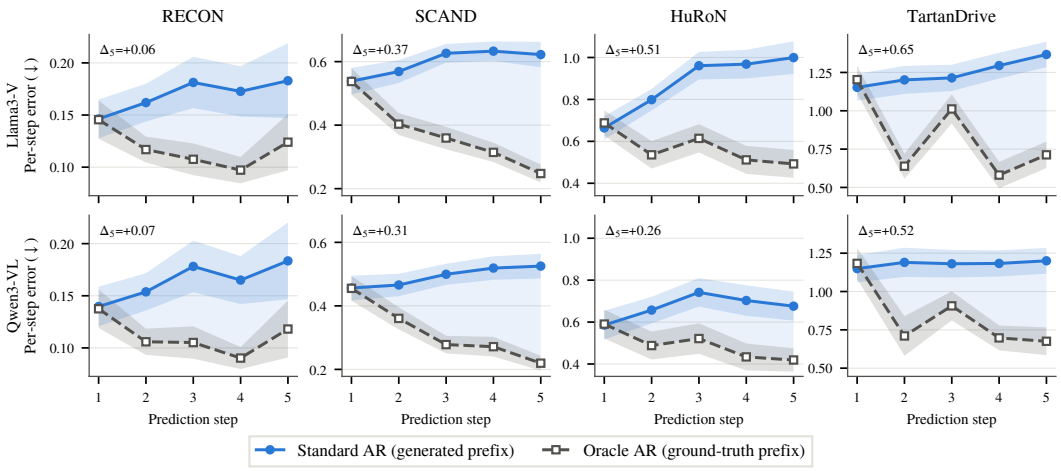


Figure A1: Extended oracle-prefix analysis across AR VLMs. Per-position prediction error for Llama3-V and Qwen3-VL on four navigation datasets at action horizon $h = 5$. Both policies are trained on RECON; therefore, RECON represents the in-distribution setting, while SCAND, HuRoN, and TartanDrive are evaluated as out-of-distribution datasets. Standard AR conditions on model-generated action prefixes, whereas Oracle AR replaces them with the corresponding ground-truth prefixes. Curves show the mean error, and shaded regions indicate 95% confidence intervals. Δ_5 denotes the error difference between Standard AR and Oracle AR at the final action position.

where L_i is the executed path length and L_i^* is the reference shortest-path length for episode i. Let E_i denote the number of actions executed before episode i terminates. We report mean trajectory length as

$$\text{Mean Traj.} = \frac{1}{N}\sum_{i=1}^{N} E_i. \tag{A8}$$

Higher SR and SPL indicate better task completion and path efficiency, respectively, while a lower mean trajectory length indicates fewer actions executed before episode termination.

B Additional Analysis of Generated-Action-Prefix Conditioning

Generated-action-prefix conditioning is analyzed through two controlled interventions (Figure 1 (b) and Figure 2 (c)). First, model-generated prefixes in AR VLM policies are replaced with ground-truth prefixes (*i.e.,* teacher forcing). Second, the same prefix-source intervention is applied to an AR-style LLaDA-V model. Here, we provide the extended results omitted from the main paper: the complete oracle-prefix analysis across two AR VLM backbones and four navigation datasets, full open-loop metrics for the fixed-backbone intervention, and a separate closed-loop comparison of generation protocols.

B.1 Oracle-Prefix Intervention across AR VLMs

In the main paper, we present representative oracle-prefix results on RECON and HuRoN. Figure A1 extends this analysis to both Llama3-V and Qwen3-VL on RECON, SCAND, HuRoN, and TartanDrive. We use the same intervention as in the main paper: Standard AR conditions later actions on the prefix generated by the model, whereas Oracle AR replaces this prefix with the corresponding ground-truth actions.

Both policies are trained on RECON. RECON therefore represents the in-distribution setting, while SCAND, HuRoN, and TartanDrive evaluate generalization under distribution shift. Across both backbones and all four datasets, Oracle AR produces lower error at the final action position. On RECON, the final-position gaps are relatively small, measuring 0.06 for Llama3-V and 0.07 for Qwen3-VL. The gaps become generally larger on the out-of-distribution (OOD) datasets, ranging

Table A2: Open-loop prefix-source intervention in AR-style LLaDA-V. We evaluate the same trained model at $h = 5$ using self-generated or ground-truth action prefixes. Δ denotes self-generated minus ground-truth.

Prefix source	RECON		SCAND		HuRoN		TartanDrive		Average	
	ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓
Self-generated	0.47	0.19	1.85	0.60	1.84	0.74	3.55	1.10	1.93	0.66
Ground truth	0.28	0.17	0.83	0.34	1.40	0.85	3.34	1.43	1.46	0.70
Δ (Self − GT)	$+0.19$	$+0.02$	$+1.02$	$+0.26$	$+0.44$	-0.11	$+0.21$	-0.33	$+0.46$	-0.04

Table A3: Fixed-backbone closed-loop generation-protocol comparison. Closed-loop comparison between two LLaDA-V policies with the same backbone architecture but different training and inference protocols. AR-style generation reduces both navigation success and path efficiency.

Generation protocol	SR (%) ↑	Mean traj. ↓	SPL (%) ↑
AR-style self-prefix	43.06	46.3	37.99
Masked diffusion	50.00	42.7	46.06
Δ (AR-style − diffusion)	-6.94	$+3.6$	-8.07

from 0.26 to 0.65 and reaching 0.65 and 0.52 on TartanDrive for Llama3-V and Qwen3-VL, respectively.

The per-position errors do not increase monotonically in every setting. Nevertheless, the error under Standard AR is consistently higher than that under Oracle AR at the final position. These complete results reinforce the observation in the main paper that later predictions are sensitive to errors in model-generated action prefixes, with a stronger effect generally observed under distribution shift.

B.2 Fixed-Backbone Interventions with LLaDA-V

We extend the open-loop prefix-source intervention in Figure 2 (c) to all four navigation datasets and separately compare action-generation protocols in closed-loop navigation.

For the prefix-source intervention, we train an AR-style LLaDA-V policy on RECON with prediction horizon $h = 5$. When predicting the j-th action, the preceding ground-truth actions $\boldsymbol{a}_{<j}$ are provided as input during training, while the remaining target positions stay masked.

Open-loop prefix-source intervention. Table A2 evaluates the same trained AR-style model using either self-generated or ground-truth action prefixes. Under the self-generated condition, each prediction uses the actions previously generated by the model. Under the ground-truth condition, these prefixes are replaced with the corresponding reference actions. The trained parameters remain fixed, and only the prefix source changes.

Replacing self-generated prefixes with ground-truth prefixes reduces ATE on all four datasets, lowering the average from 1.93 to 1.46. The reductions are 0.19 on RECON, 1.02 on SCAND, 0.44 on HuRoN, and 0.21 on TartanDrive, respectively.

RPE changes are mixed, decreasing on RECON and SCAND but increasing on HuRoN and TartanDrive. Under ground-truth-prefix conditioning, each prediction uses the ground-truth action prefix, whereas the evaluated trajectory is reconstructed from the model's predicted actions. The mismatch between the conditioning history and the predicted trajectory may affect local motion consistency even when the trajectory has lower ATE.

Closed-loop generation-protocol comparison. Table A3 reports a separate comparison between the AR-style policy and a full-mask single-pass policy. Both policies are trained on the same RECON data using the same backbone architecture, visual inputs, action vocabulary, and prediction horizon. The joint-prediction policy follows the full-mask training scheme used by PAM and pre-

Table A4: Effects of training-time masking and inference-time decoding. Open-loop ATE and RPE are reported on the four navigation datasets at action horizon $h = 5$, and closed-loop SR and SPL are evaluated on WIW ImageNav. All configurations use ACC with $k = 3$. Random-mask models benefit substantially from iterative decoding in open-loop prediction without improving closed-loop SR. In contrast, PAM achieves strong performance with a single forward pass, and iterative decoding provides only marginal additional gains.

Training	Inference	RECON		SCAND		HuRoN		TartanDrive		Average		Closed-loop	
		ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓	SR (%) ↑	SPL (%) ↑
Random mask	Single-pass	0.51	0.26	1.18	0.51	1.60	0.78	1.65	0.77	1.23	0.58	38.89	30.48
Random mask	Iterative ($T = 5$)	0.39	0.18	0.68	0.28	**0.95**	**0.43**	**0.68**	**0.33**	**0.67**	**0.30**	38.89	33.03
PAM	Single-pass	0.37	0.19	0.61	0.29	1.00	0.51	0.73	0.39	0.68	0.34	**59.03**	50.15
PAM	Iterative ($T = 5$)	**0.36**	**0.17**	**0.54**	**0.25**	0.99	0.46	0.78	0.37	**0.67**	0.31	58.33	**51.70**

dicts all action tokens simultaneously in a single forward pass. The AR-style policy instead generates actions sequentially, reusing its preceding predictions as context.

Compared with the joint-prediction policy, the AR-style policy achieves lower SR (43.06% vs. 50.00%) and SPL (37.99% vs. 46.06%), with a higher mean number of executed actions (46.3 vs. 42.7).

Together with the oracle-prefix results in Figure A1, the open-loop intervention shows that sensitivity to generated action prefixes also arises in a diffusion-pretrained VLM adapted to AR-style generation. The separate closed-loop comparison shows higher navigation success and path efficiency for joint prediction.

C Additional Ablation Studies

This section provides additional detailed ablation results that complement the analyses in the main paper. We first examine the interaction between the training-time masking strategy and the inference-time decoding protocol, and then analyze the effect of the number of iterative refinement steps. Finally, we provide the complete per-dataset ablation results for PAM and ACC omitted from the main paper.

C.1 Training-Time Masking and Inference-Time Decoding

The main paper introduces Prefix-Free Action Block Modeling (PAM), which predicts all action tokens in a single forward pass from a fully masked target block. However, diffusion models can also be decoded iteratively. This raises the question of how training-time masking and inference-time decoding affect open-loop prediction and closed-loop navigation.

To answer this question, Table A4 compares four combinations obtained by crossing the training-time masking strategy (random masking versus PAM) with the inference-time decoding protocol (single-pass versus iterative decoding). All four configurations use ACC with the default context length $k = 3$. For iterative decoding, we use $T = 5$ refinement steps.

The results reveal different behaviors for the two training objectives. For the random-mask model, iterative decoding substantially improves open-loop prediction, reducing the average ATE from 1.23 to 0.67 and the average RPE from 0.58 to 0.30 compared to the single-pass inference. However, these improvements translate only marginally to closed-loop navigation, where SR remains unchanged (38.89%) and SPL increases only slightly from 30.48% to 33.03%.

With PAM, iterative decoding changes SR from 59.03% to 58.33% while increasing SPL from 50.15% to 51.70%. These results indicate a limited benefit from additional refinement, which requires five forward passes instead of one. We therefore use single-pass decoding as the default.

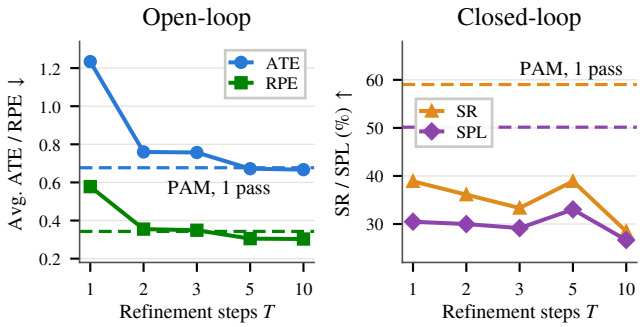


Figure A2: Sensitivity to the number of iterative refinement steps T. A model trained with random masking is decoded with $T \in \{1, 2, 3, 5, 10\}$ refinement steps and evaluated in open loop (left) and closed loop (right). Open-loop values are averaged over the four evaluation datasets. The dashed lines mark LLaDA-Nav decoded with a single forward pass. Iterative refinement recovers most of the open-loop gap by $T = 2$, but the closed-loop metrics remain well below the single-pass reference at every T.

Table A5: Per-dataset component ablation. Open-loop ATE and RPE on the four evaluation datasets at action horizon $h = 5$ for Prefix-Free Action Block Modeling (PAM) and Action Context Conditioning (ACC). Average denotes the mean across the four datasets, and lower is better for both metrics. This table reports the per-dataset breakdown of Table 3.

PAM	ACC	RECON		SCAND		HuRoN		TartanDrive		Average	
		ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓
		0.79	0.45	0.96	0.42	1.62	0.80	1.33	0.49	1.17	0.54
✓		0.46	0.21	1.19	0.45	1.33	0.73	1.53	0.57	1.13	0.49
	✓	0.51	0.26	1.18	0.51	1.60	0.78	1.65	0.77	1.23	0.58
✓	✓	**0.37**	**0.19**	**0.61**	**0.29**	**1.00**	**0.51**	**0.73**	**0.39**	**0.68**	**0.34**

C.2 Sensitivity to Iterative Refinement Steps

Although Table A4 evaluates iterative decoding using $T = 5$, diffusion models allow the refinement process to continue for an arbitrary number of iterations. We therefore investigate whether increasing the number of refinement steps can further improve navigation performance.

Figure A2 reports results for a random-mask model decoded with $T \in \{1, 2, 3, 5, 10\}$. For open-loop evaluation, ATE and RPE are averaged over the four navigation datasets. Closed-loop evaluation reports SR and SPL on WIW ImageNav.

Open-loop accuracy improves substantially between $T = 1$ and $T = 2$, after which additional refinement provides only marginal benefit. By $T = 5$, the average ATE approaches the performance of the single-pass PAM model, and increasing the refinement budget further produces little change.

Despite improved open-loop accuracy, the random-mask model remains below the single-pass PAM reference in both SR and SPL across all tested refinement budgets, with both metrics declining at $T = 10$. These results suggest that additional refinement alone is insufficient to close the closed-loop performance gap.

C.3 PAM and ACC across Navigation Datasets

We report the aggregate component ablation of PAM and ACC in the main paper. Table A5 provides the corresponding per-dataset results on RECON, SCAND, HuRoN, and TartanDrive.

PAM alone slightly improves the average ATE and RPE from $1.17/0.54$ to $1.13/0.49$, but its effect varies across datasets. ACC alone does not improve average open-loop accuracy, yielding an average

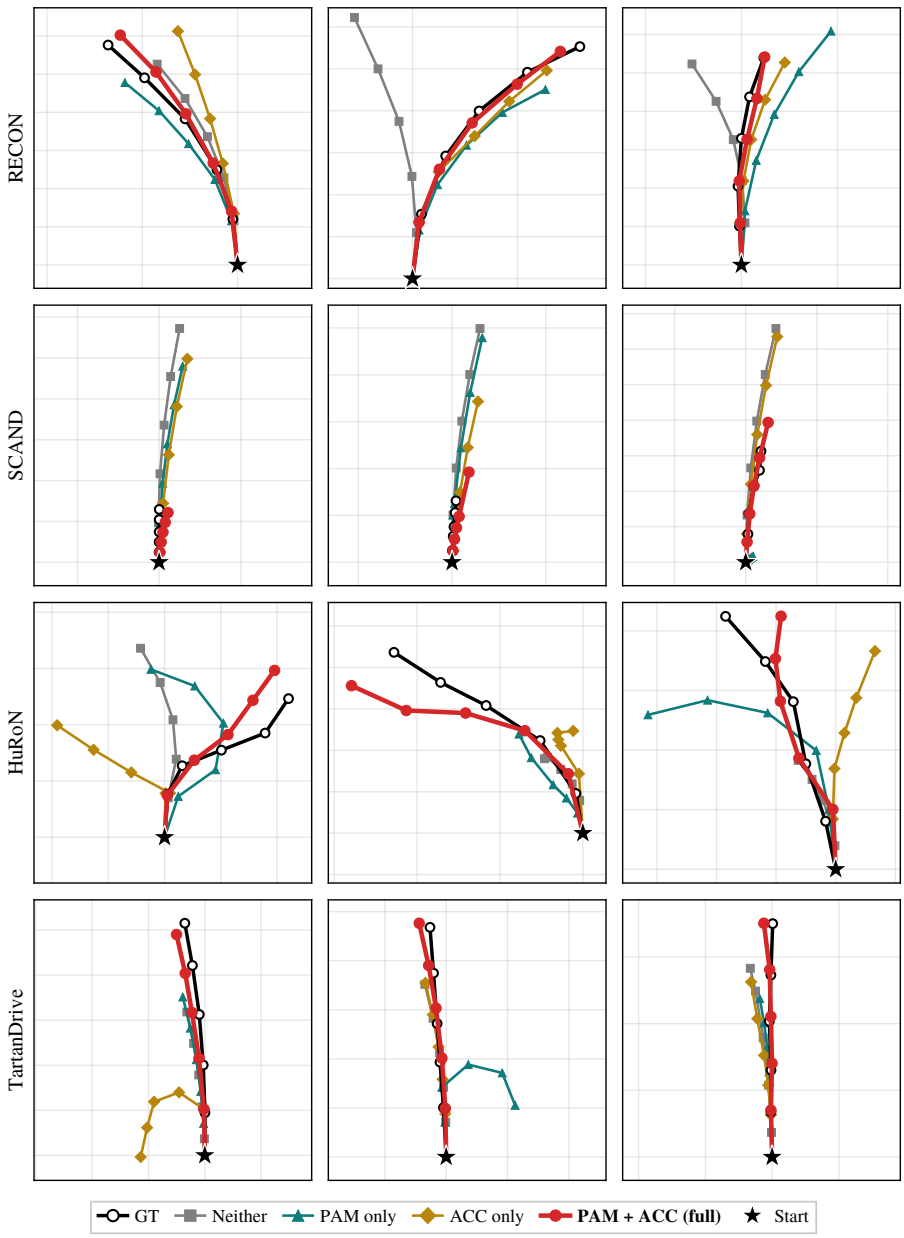


Figure A3: Qualitative component ablation of PAM and ACC. Each row shows three open-loop examples from one dataset. Black denotes ground truth, gray neither component, teal PAM only, ochre ACC only, and red both components; stars mark the starting poses. All configurations use single-pass decoding.

ATE and RPE of 1.23 and 0.58, respectively. Thus, under random-mask training and single-pass decoding, executed-action history alone does not consistently improve prediction across datasets.

Combining PAM and ACC reduces the average ATE from 1.17 to 0.68 and RPE from 0.54 to 0.34. The combined model also achieves the lowest ATE and RPE on every evaluated dataset. These results support the complementary effects of full-mask training and executed-action history under a fixed single-pass decoding protocol. As discussed in the main paper, full-mask training may help the model learn and use ACC's action-history representations more effectively.

Figure A3 provides qualitative examples of the four component configurations. The combined model better matches the ground-truth motion magnitude in the displayed SCAND and TartanDrive examples, although deviations in turning direction remain in some HuRoN cases.

Table A6: Detailed ablation results for the executed-action context length k. Open-loop ATE and RPE on the four evaluation datasets at action horizon $h = 5$, together with closed-loop SR and SPL. Average denotes the mean across the four datasets. Lower is better for ATE and RPE, and higher is better for SR and SPL. This table reports the per-dataset breakdown of Figure 5.

k	RECON		SCAND		HuRoN		TartanDrive		Average		Closed-loop	
	ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓	SR (%) ↑	SPL (%) ↑
0	0.46	0.21	1.19	0.45	1.33	0.73	1.53	0.57	1.13	0.49	44.44	36.64
1	0.43	0.22	0.68	0.34	1.28	0.66	0.88	0.42	0.82	0.41	56.25	**51.04**
3	**0.37**	**0.19**	**0.61**	**0.29**	**1.00**	**0.51**	**0.73**	**0.39**	**0.68**	**0.34**	**59.03**	50.15
5	0.59	0.30	0.65	0.30	1.48	0.73	1.15	0.58	0.97	0.47	34.72	27.34

Table A7: Per-dataset results for the LoRA rank r. Open-loop ATE and RPE on the four evaluation datasets at action horizon $h = 5$, together with closed-loop SR and SPL. Average denotes the mean across the four datasets. Lower is better for ATE and RPE, and higher is better for SR and SPL. This table reports the per-dataset breakdown of Figure 5.

r	RECON		SCAND		HuRoN		TartanDrive		Average		Closed-loop	
	ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓	SR (%) ↑	SPL (%) ↑
8	**0.33**	**0.16**	0.62	**0.28**	**0.94**	**0.48**	**0.71**	**0.34**	**0.65**	**0.31**	47.92	43.56
16	0.34	0.17	0.75	0.34	1.00	**0.48**	0.75	0.37	0.71	0.34	44.44	41.16
32	0.36	0.18	0.64	0.30	1.15	0.61	0.78	0.39	0.73	0.37	49.31	39.07
64	0.37	0.19	**0.61**	0.29	1.00	0.51	0.73	0.39	0.68	0.34	**59.03**	**50.15**
128	0.53	0.25	0.67	**0.28**	1.35	0.64	1.30	0.54	0.96	0.43	57.64	49.84

C.4 Exact Hyperparameter Results

We summarize the effects of the ACC context length k and the LoRA rank r in Figure 5. Here, we report the exact per-dataset open-loop results and corresponding closed-loop SR and SPL.

Executed-action context length. Table A6 varies the number of previously executed actions provided by ACC. Using one executed action ($k = 1$) improves the average ATE from 1.13 to 0.82 and increases SR from 44.44% to 56.25% relative to $k = 0$. Increasing the context to $k = 3$ further reduces the average ATE and RPE to 0.68 and 0.34 and yields the highest SR of 59.03%.

Extending the context to $k = 5$ degrades both open- and closed-loop performance, increasing the average ATE to 0.97 and reducing SR and SPL to 34.72% and 27.34%. This suggests that ACC benefits from a short and recent motion history, whereas a longer context does not necessarily provide additional useful information. We therefore use $k = 3$ by default.

LoRA rank. Table A7 reports the effect of the LoRA rank. A small rank of $r = 8$ produces the lowest average open-loop error, with 0.65 ATE and 0.31 RPE, but reaches only 47.92% closed-loop SR. In contrast, $r = 64$ achieves the highest closed-loop performance, with 59.03% SR and 50.15% SPL, while retaining competitive open-loop accuracy.

Increasing the rank to $r = 128$ does not provide further gains and increases the average ATE to 0.96. These results show that the setting with the lowest open-loop error does not necessarily yield the highest closed-loop success. Because closed-loop navigation is the primary deployment criterion, we use $r = 64$ as the default.

C.5 Application to MMaDA

To examine the applicability of PAM and ACC beyond LLaDA-V, we apply both components to MMaDA (Yang et al., 2025). For MMaDA, we use LoRA rank $r = 16$, train for 28,000 steps, and apply LoRA only to the language model. All other training and evaluation settings match the default LLaDA-V configuration. As shown in Table A8, MMaDA-Nav achieves lower ATE and RPE on all four datasets than the default LLaDA-V-based policy, with 70.14% closed-loop SR

Table A8: Application of PAM and ACC to MMaDA.

Policy	RECON		SCAND		HuRoN		TartanDrive		Average		Closed-loop	
	ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓	ATE ↓	RPE ↓	SR (%) ↑	SPL (%) ↑
LLaDA-Nav	0.37	0.19	0.61	0.29	1.00	0.51	0.73	0.39	0.68	0.34	59.03	50.15
MMaDA-Nav	**0.27**	**0.14**	**0.47**	**0.22**	**0.70**	**0.38**	**0.67**	**0.32**	**0.53**	**0.26**	**70.14**	**64.51**

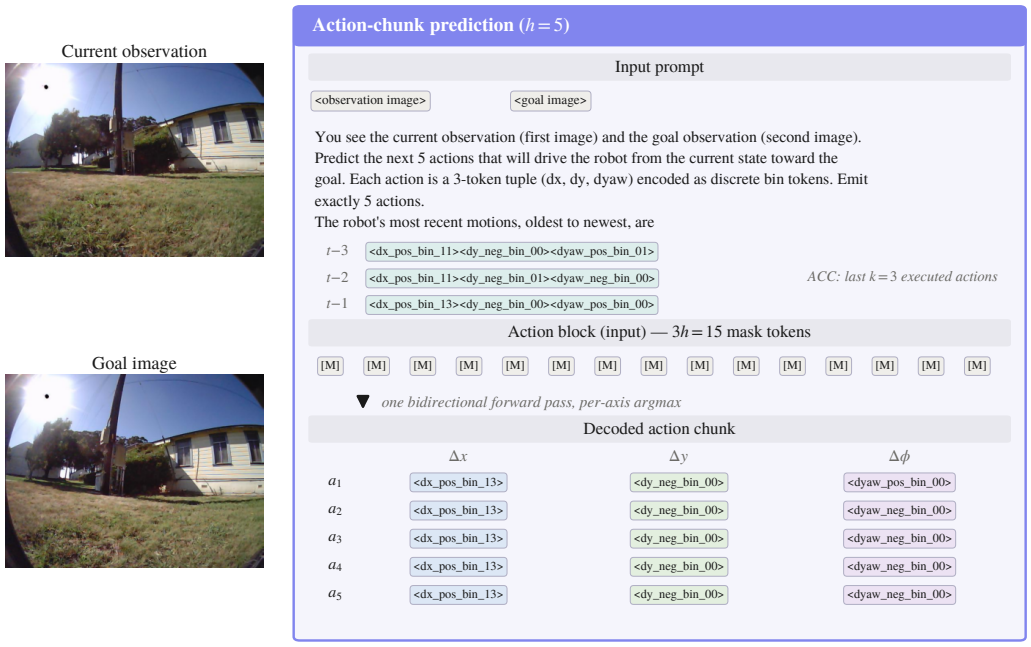


Figure A4: Prompt construction and decoding example for LLaDA-Nav at action horizon $h = 5$. The base multimodal prompt presents the current observation and the goal image in order, followed by the text instruction that states the image-goal objective and the required output format. The action-context conditioning block appends the last $k = 3$ executed actions as action tokens. The target block is then appended as $3h = 15$ mask tokens, and a single forward pass with component-wise argmax decodes all target positions at once. The decoded action chunk is listed as one row per action $\boldsymbol{a}_j$, with the discrete bin tokens for the two translational components and the heading component.

and 64.51% SPL. These results support the applicability of PAM and ACC beyond LLaDA-V as a shared approach to action generation across multimodal diffusion backbones. This comparison demonstrates transferability of the approach.

D Prompt Design and Examples

We provide the complete input construction used by LLaDA-Nav during training and inference. The prompt consists of four components: an ordered pair of visual inputs, a navigation instruction, an optional executed-action context, and a fully masked target block. The same construction is used across all four training datasets and in both open- and closed-loop evaluation.

D.1 Multimodal Prompt Construction

The visual input contains the current observation followed by the goal image. A fixed text instruction then specifies the image-goal navigation objective, the prediction horizon h, and the required

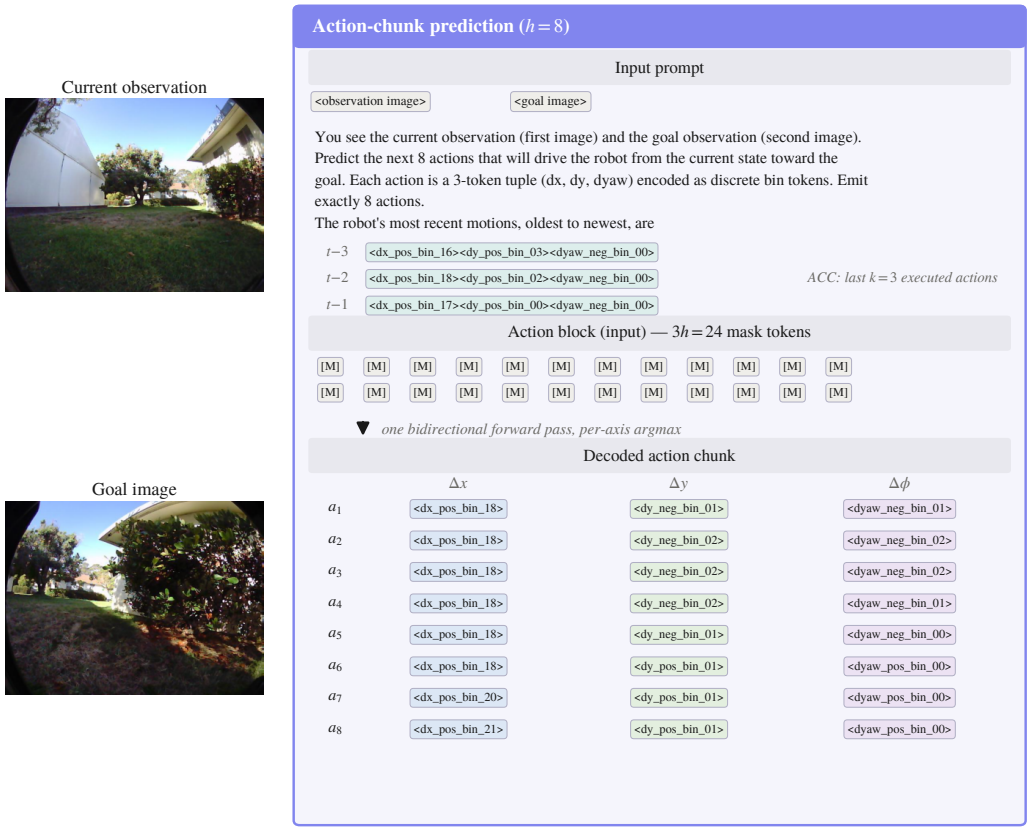


Figure A5: Prompt construction and decoding example for LLaDA-Nav at action horizon $h = 8$. The layout matches Figure A4. Only the instructed number of actions and the size of the target block change with the horizon, which here yields $3h = 24$ mask tokens and an eight-step decoded action chunk. The prompt template, the action vocabulary, and the single-pass decoding procedure are identical across the two horizons.

action representation. Each action is represented by three discrete tokens corresponding to the two translational components and the heading component.

When ACC is enabled, the instruction is followed by the k most recently executed actions, ordered from oldest to newest. Each executed action is represented using the same action vocabulary as the target action chunk. At the beginning of an episode, unavailable context positions are filled with zero-motion tokens, as described in Section A.

The target block is appended after the executed-action context and contains exactly $3h$ mask tokens. LLaDA-Nav predicts an action-token distribution at every target position from this fully masked block in a single forward pass. The resulting logits are restricted to the corresponding action-token subvocabulary for each component, and the token with the highest probability is selected independently at every target position.

For ACC variants at a fixed prediction horizon, the visual-input order, navigation instruction, action vocabulary, and target-block structure remain unchanged. The ACC variants differ only by the insertion of the executed-action description and its corresponding action tokens immediately before the target block.

D.2 Examples at Different Action Horizons

Figure A4 shows the complete prompt and decoded output for the default action horizon $h = 5$. The instruction requests five actions, and the appended target block therefore contains $3h = 15$ mask tokens. The example also shows the $k = 3$ executed actions supplied by ACC and the five decoded action tuples obtained in a single forward pass.

Figure A5 presents the corresponding construction for $h = 8$. The visual-input order, instruction template, action vocabulary, ACC construction, and decoding procedure remain unchanged. Only the requested number of actions and the length of the target block are modified, resulting in $3h = 24$ mask tokens and an eight-action output chunk. These examples illustrate that the same prompt interface supports different prediction horizons without changing the model architecture or decoding procedure.

E Additional Qualitative Results

We provide additional qualitative results for both open-loop trajectory prediction and closed-loop image-goal navigation. These visualizations complement the aggregate metrics in the main paper by showing how the predicted action chunks and executed trajectories differ across datasets, environments, and policy families.

E.1 Open-Loop Trajectory Predictions

Figure A6 compares LLaDA-Nav with GNM, ViNT, NoMaD, and OmniVLA on representative held-out trajectory segments from RECON, SCAND, HuRoN, and TartanDrive. Each row corresponds to one dataset, and each column shows a different evaluation sample. All trajectories are expressed relative to the same initial robot pose, marked by a star.

For a common visualization span, we show only the first five predicted waypoints from each method. All methods share the same starting observation and five-step ground-truth reference, but baselines with a native prediction horizon of $h = 8$ receive a goal image farther along the same underlying trajectory. The black trajectory represents the ground truth, while the red trajectory denotes LLaDA-Nav.

In the displayed examples, LLaDA-Nav generally preserves the direction and curvature of the ground-truth trajectory across held-out samples from all four training domains. The qualitative differences become particularly visible when a baseline overestimates forward motion (especially in SCAND and TartanDrive), predicts an incorrect turning direction (especially NoMaD in RECON and HuRoN), or deviates progressively from the target trajectory. These examples qualitatively illustrate the prediction errors of the baselines, which also have higher average ATE and RPE in the main evaluation.

E.2 Closed-Loop Navigation Rollouts

Figure A7 shows representative closed-loop rollouts from the WIW ImageNav evaluation on unseen HM3D scenes. Each panel overlays the trajectories executed by all policies on a top-down map of the corresponding environment. The star and diamond indicate the starting and goal positions, respectively.

Unlike open-loop prediction, each trajectory in this figure results from repeated observation, action-chunk prediction, partial execution, and replanning. The visualization therefore reflects not only the accuracy of an individual action chunk but also how prediction errors affect subsequent observations and decisions.

Across the shown episodes, LLaDA-Nav generally follows a more direct path toward the goal, whereas several baseline policies take long detours, repeatedly change direction, or move away from the goal. These qualitative examples are also consistent with its higher SR and SPL and fewer executed actions on average in the quantitative evaluation.

F Limitations

Our experiments primarily use LLaDA-V, with an additional evaluation on MMaDA; closed-loop evaluation is limited to Habitat-Sim. Future work will extend PAM and ACC to a broader range of diffusion-based VLMs to examine their generality across different model architectures. We also plan to deploy LLaDA-Nav on physical robots and evaluate its robustness under real-world closed-loop conditions.

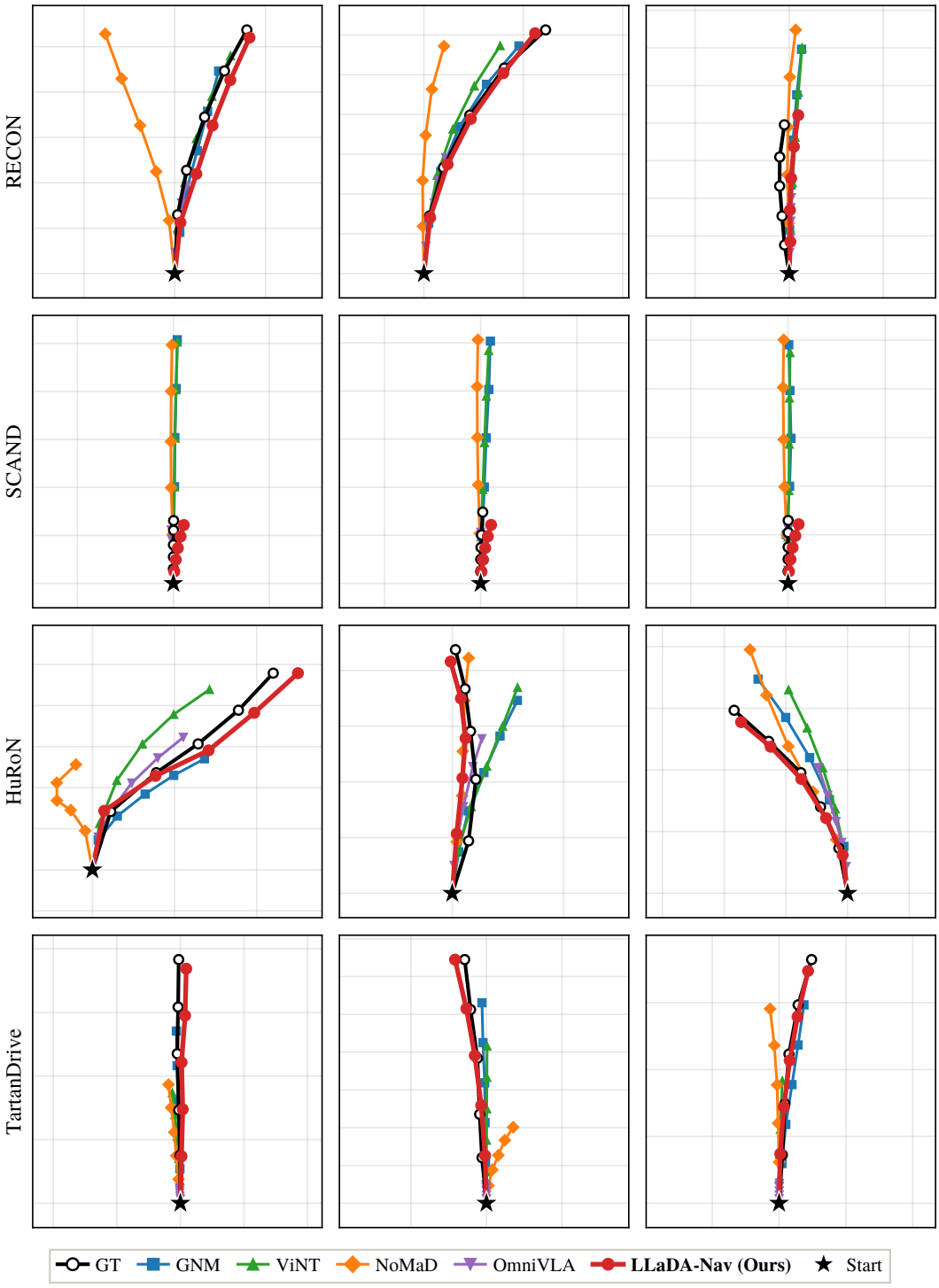


Figure A6: Open-loop trajectory predictions on all four evaluation datasets. Each row corresponds to one dataset (RECON, SCAND, HuRoN, and TartanDrive), and the three columns show different held-out trajectory segments from that dataset. Black denotes the ground-truth trajectory and red denotes LLaDA-Nav, while the remaining colors denote the baseline predictions. The star marks the segment origin, and the markers along each curve represent predicted waypoints. Every curve is plotted over the first five predicted waypoints. For the baselines that predict at $h = 8$, only the first five actions of the predicted chunk are shown, so that all methods are compared over the same prediction span; these baselines receive a goal image farther along the same underlying trajectory.



Figure A7: Closed-loop image-goal navigation rollouts on unseen HM3D scenes. Each panel shows one evaluation episode, with the executed trajectories of all policies overlaid on the top-down map of the scene. The star marks the episode start and the diamond marks the goal. Red denotes LLaDA-Nav, and the remaining colors denote the baselines.