WORLD SLAM MODEL: JOINT WORLD MODELING FOR SLAM AND NAVIGATION

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce **World SLAM Model (WSM)**, a unified framework that brings the SLAM paradigm directly into downstream navigation. Rather than treating SLAM merely as an upstream module that provides poses, maps or tokens, WSM adopts its core mechanisms, including incremental state updates with persistent memory and backend refinement of accumulated errors, to maintain a consistent world state during interaction. Given the current observation and a navigation goal, WSM predicts future visual states and jointly estimates their camera motion and dense geometry, grounding visual prediction in an evolving spatial world state. This spatial state is continuously updated as new observations arrive and provides the basis for action generation and closed-loop navigation. WSM is trained end-to-end with a joint navigation–SLAM objective, enabling downstream navigation to benefit directly from SLAM-style state maintenance and refinement while preserving accurate geometric estimation. Experiments demonstrate improved navigation performance together with strong SLAM accuracy, highlighting the potential of SLAM as an intrinsic mechanism for long-horizon world modeling and embodied interaction.

1 INTRODUCTION

SLAM has long served as a fundamental spatial module for robotics (Davison et al., 2007). Given a stream of observations, a SLAM system estimates camera motion and scene structure, producing poses and maps that are subsequently consumed by downstream tasks such as navigation and control (Weiss et al., 2011; Cadena et al., 2016). In a modular navigation pipeline, the SLAM module supplies poses and maps to downstream planning and control, while continuously maintaining and refining these estimates (Chaplot et al., 2020).

To reduce the complexity of this modular pipeline, recent navigation methods increasingly adopt end-to-end models that directly map visual observations to trajectories or actions (Cheng et al., 2025; Zhang et al., 2024; Shah et al., 2023). While this removes an explicit SLAM module from the navigation pipeline, it does not eliminate the dependence on spatial information traditionally provided by SLAM (Yin et al., 2025). StreamVLN (Wei et al., 2025) and Dynam3D (Wang et al., 2026b) still take depth as input, while NavDP (Cai et al., 2025) requires online robot poses or goal locations from external ground-truth localization. More recent approaches obtain such information from pretrained 3D reconstruction models (Peng et al., 2025; Zeng et al., 2025), but still treat geometry as an upstream estimate fed to the navigation module.

However, directly using SLAM outputs still leaves state estimation and navigation separately optimized, preventing navigation objectives from shaping how the world state is learned and maintained. We argue that navigation should instead benefit from *SLAM itself*, including its mechanisms for maintaining memory and correcting accumulated errors. Integrating SLAM into the navigation model benefits both training and inference. During training, geometric and navigation objectives jointly shape the shared world-state representation and its maintenance mechanisms. During inference, shared streaming memory and error correction continuously update the state used for planning and action prediction.

World models provide a natural framework for realizing this idea. A world model must continuously process observations, retain information from past interactions, predict future evolution, and update its state as new observations arrive. These closely match the core mechanisms of SLAM, making

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.

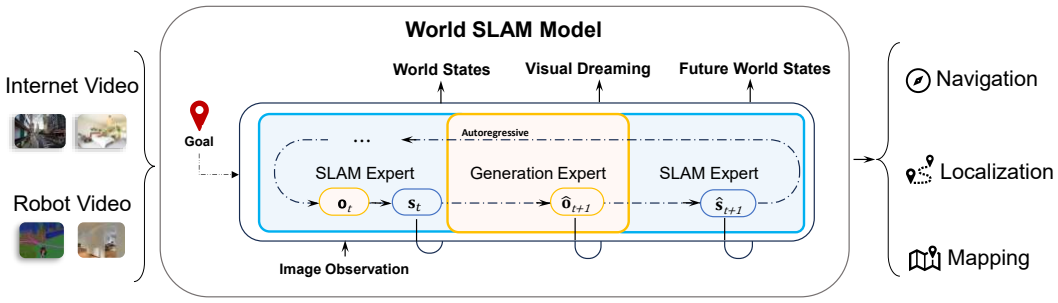

Figure 1: **Overview of the World SLAM Model.** WSM integrates a SLAM expert and a generation expert into a unified model that autoregressively estimates world states (i.e., depth and camera pose), dreams future observations, and infers future world states, supporting navigation, localization, and mapping.

SLAM a suitable computational structure for long-horizon world modeling rather than merely an external source of geometric estimates.

Based on this observation, we introduce the **World SLAM Model (WSM)**, which brings the SLAM mechanism directly into downstream navigation. Instead of following the conventional *SLAM* $\rightarrow$ *navigation* pipeline, WSM incorporates spatial estimation, memory, prediction, and action into a shared closed-loop model. In this way, navigation benefits not only from what SLAM estimates, but also from how SLAM continuously maintains and corrects the world state.

Specifically, the SLAM expert first estimates depth and camera poses from the current observation. Conditioned on the goal, the generation expert then dreams future observations from which the same SLAM expert, acting as an inverse dynamics model, infers the future world states and recovers the camera trajectory. After the trajectory is executed, new observations are fed back into the model, forming a continuous prediction–interaction–update loop. A persistent KV cache enables incremental processing across interaction steps, while the SLAM expert maintains spatial consistency as observations accumulate.

We evaluate WSM on both closed-loop navigation and SLAM. On the InternVLA-N1 (InternNav Team, 2025) start–goal navigation benchmark, WSM relies only on RGB observations and its own geometric estimates, yet achieves higher success rates than all evaluated baselines. On held-out InternVLN-N1 trajectories, we further evaluate its SLAM capability to verify that the model retains accurate spatial estimation while serving the downstream task.

Our contributions are summarized as follows:

- We revisit the role of SLAM in embodied systems and argue that downstream tasks can benefit more from incorporating the *SLAM mechanism itself* into the model than from only consuming the outputs of an external SLAM module, such as poses, maps or tokens.
- We introduce **World SLAM Model (WSM)**, which incorporates incremental processing, persistent memory, spatial estimation, and closed-loop interaction into a unified navigation model.
- We evaluate WSM on both navigation and SLAM, demonstrating the potential of using SLAM as an intrinsic computational mechanism for downstream embodied tasks.

2 Related Work

2.1 SLAM and Feed-Forward 3D Reconstruction

Optimization-based SLAM systems (Campos et al., 2021; Teed & Deng, 2021) estimate camera poses and scene geometry from sequential observations, combining front–end tracking with loop closure and backend optimization to maintain global consistency and correct accumulated drift.

Feed-forward reconstruction models instead regress geometry directly from images. VGGT (Wang et al., 2025), D4RT (Zhang et al., 2026) and π^3 (Wang et al., 2026a) jointly predict camera parameters, depth maps, and point maps. StreamVGGT (Zhuo et al., 2026) adapts it to streaming inputs through causal attention with cached historical tokens. LingBot-Map (Chen et al., 2026) maintains a compact geometric context for long-sequence streaming reconstruction, and SLAM-Former (Yuan et al., 2026b) integrates incremental tracking, mapping, and global refinement into a single transformer. Across these formulations, however, geometric state estimation remains the primary objective, while navigation is left as a separate downstream task.

WSM takes a complementary direction: rather than treating SLAM solely as a geometric estimation objective, it incorporates the underlying SLAM mechanisms into a world-modeling framework for navigation.

2.2 Visual Navigation

Classical navigation systems follow a modular pipeline in which SLAM provides localization and mapping to a downstream planner and controller (Marder-Eppstein et al., 2010). End-to-end methods (Yang et al., 2023; Cai et al., 2025; Roth et al., 2024) map observations directly to trajectories, reducing error propagation across module interfaces, but typically obtain the robot pose or goal position from external localization. Building on feed-forward 3D reconstruction models such as VGGT (Wang et al., 2025), LoGoPlanner (Peng et al., 2025) fine-tunes a pretrained visual-geometry backbone with depth-derived scale priors for localization and scene reconstruction without relying on external odometry. However, these approaches still depend on high-fidelity depth sensing, a separately pretrained 3D reconstruction model, or complex, system-specific SLAM optimization modules. The mechanisms for geometric state estimation and drift correction are not embedded in the navigation model.

In contrast, WSM brings the *SLAM mechanism itself* into the navigation model: incremental processing with persistent memory maintains spatial context over time, while backend refinement mitigates accumulated drift.

2.3 World Models

World models capture how a world appears across views and evolves over time, allowing unobserved states to be generated, reconstructed, or simulated. Cosmos (Agarwal et al., 2026) shows that world foundation models pretrained on large-scale video can be post-trained into customized world models for downstream physical AI applications, and Atlas (World Labs Team, 2026) indicates that grounding views at explicit camera poses allows a single model to support generation, reconstruction, and simulation. For downstream navigation and manipulation tasks, NWM (Bar et al., 2025) shows that an action-conditioned video model can serve as a navigation planner that incorporates new constraints at planning time, unlike supervised policies with fixed behavior. World action models such as LingBot-VA (Li et al., 2026a) and Fast-WAM (Yuan et al., 2026a) jointly model future visual observations and actions for embodied control. Existing world models have explored predictive visual dynamics and explicit geometric representations, but they generally do not use a shared, persistent metric state to connect visual prediction with downstream tasks.

WSM instead adopts the SLAM paradigm for world modeling, combining incremental state updates with backend refinement to maintain a spatially consistent world state over long-horizon interaction.

3 World SLAM Model

3.1 Overview

We formulate the navigation-oriented world SLAM model in terms of (1) observation-conditioned visual dynamics prediction, (2) observation-conditioned metric state estimation, and (3) dream-conditioned trajectory recovery. Observations are encoded by the Wan2.2 causal VAE (Wan et al., 2025) with a temporal compression ratio of $M = 4$, so each latent frame covers M raw frames. At latent timestep t, the model receives RGB observations $\mathbf{o}_t \in \mathcal{O}$ and represents their metric state as $\mathbf{s}_t = (\mathbf{d}_t, \{\mathbf{g}_{t,m}\}_{m=1}^{M})$, where $\mathbf{d}_t \in \mathcal{D}$ denotes scene depth and $\mathbf{g}_{t,m} \in \mathcal{G}$ denotes the camera pose

of the m-th raw frame. The model operates on chunks of K timesteps, and we use $\mathbf{O}_n$ and $\mathbf{S}_n$ to denote the observations and metric states of the n-th chunk.

Method overview. Figure 2 illustrates our framework. Section 3.2 introduces our navigation formulation with the video model and the SLAM model. Section 3.3 presents our unified architecture and its autoregressive modeling. Section 3.4 describes pretraining and post-training, and Section 3.5 describes our inference pipeline.

3.2 Video Model and SLAM Model for Navigation

For clarity, we formulate each step at the granularity of one latent frame; in practice, the model processes a chunk of K latent frames jointly, *e.g.*, dreaming $\hat{\mathbf{o}}_{t+1:t+K}$ at once.

Dream for visual states. Given RGB observations $\mathbf{o}_{\leq t}$ and their estimated metric states $\mathbf{s}_{\leq t}$, the visual dynamics model p_θ dreams the future visual state:

$$\hat{\mathbf{o}}_{t+1} \sim p_\theta(\cdot \mid \mathbf{o}_{\leq t}, \mathbf{s}_{\leq t}), \tag{1}$$

where the hat denotes dreamed quantities. The dreamed RGB sequence describes a visual transition but does not itself specify the corresponding metric motion. We recover that motion through the SLAM expert below.

SLAM for world states. The SLAM expert f_ψ recovers camera motion and scene geometry from temporal visual observations. Applied to a dreamed visual transition, it infers the corresponding camera motion, which amounts to inverse dynamics. The same f_ψ operates on observed and dreamed sequences:

$$\mathbf{s}_{t+1} = f_\psi(\mathbf{o}_{t+1} \mid \mathbf{o}_{\leq t}, \mathbf{s}_{\leq t}), \tag{2}$$

$$\hat{\mathbf{s}}_{t+1} = f_\psi(\hat{\mathbf{o}}_{t+1} \mid \mathbf{o}_{\leq t}, \mathbf{s}_{\leq t}). \tag{3}$$

To correct drift accumulated in incremental estimation, the SLAM expert runs a local backend:

$$\mathbf{s}_{t-k+1:t} = f_\psi(\mathbf{o}_{t-k+1:t} \mid \mathbf{o}_{\leq t-k}, \mathbf{s}_{\leq t-k}), \quad k \leq K, \tag{4}$$

which jointly re-estimates the latest k frames within a chunk of size K, and updates their states and KV cache.

Start-goal navigation. We additionally condition the dreaming process on the goal $\mathbf{p}_{\text{goal}} \in \mathbb{R}^3$, a position in the first-camera coordinate frame without orientation:

$$\hat{\mathbf{o}}_{t+1} \sim p_\theta(\cdot \mid \mathbf{o}_{\leq t}, \mathbf{s}_{\leq t}, \mathbf{p}_{\text{goal}}). \tag{5}$$

The pose sequence recovered from this goal-conditioned dream by eq. (3) serves as the metric plan.

3.3 Unified Architecture and Autoregressive Modeling

Architecture. We instantiate p_θ and f_ψ within a single Mixture-of-Transformers (MoT) (Liang et al., 2025) with three experts: a generation expert for RGB tokens, and two experts for metric states, depth and pose, which together form the SLAM expert. Each expert keeps its own parameters, while all tokens interact through joint self-attention at every layer. RGB observations and depth are encoded by the Wan2.2 VAE and patchified into tokens, and pose tokens are decoded into camera poses by a pose head. Since depth and pose tokens are mutually visible and regressed jointly, we refer to the depth and pose experts together as the SLAM expert f_ψ (Figure 2). The goal $\mathbf{p}_{\text{goal}}$ is embedded as a goal token that conditions only the generation expert via cross-attention.

Training sequence. Following the teacher-forcing scheme of LingBot-VA (Li et al., 2026a), we organize the input tokens of each chunk n into two groups (Figure 3a): clean tokens $\mathbf{O}_n$ and $\mathbf{S}_n$, which serve as context; and prediction tokens, which comprise noisy RGB tokens $\mathbf{O}_n^\tau$ for the generation expert and fixed state queries $\bar{\mathbf{S}}_n$ with zero depth and identity pose for the SLAM expert. All chunks of an episode are thus supervised in parallel within a single forward pass.

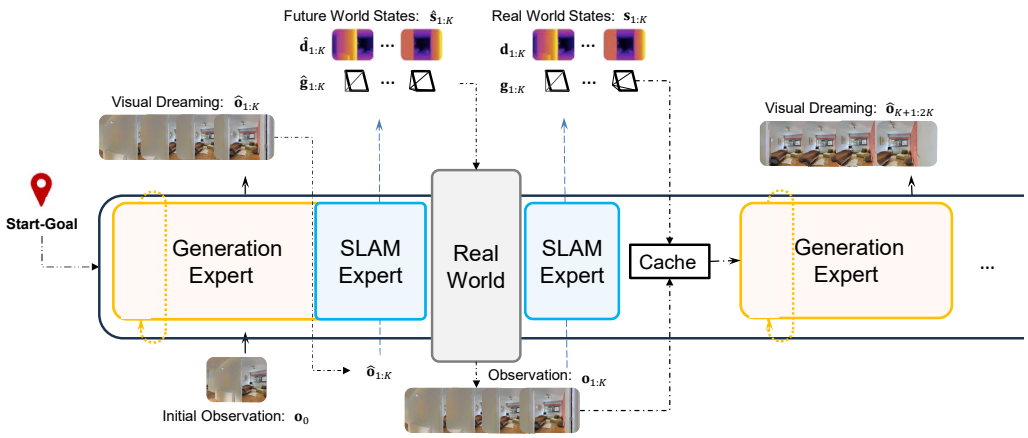

Figure 2: **Framework overview.** Our model unifies visual dreaming and SLAM within an autoregressive *Mixture-of-Transformers (MoT)* for start–goal navigation. Conditioned on the start–goal and the initial observation $\mathbf{o}_0$, the generation expert first *dreams* future RGB observations $\hat{\mathbf{o}}_{1:K}$, from which the SLAM expert recovers future world states $\hat{\mathbf{s}}_{1:K}$ as an *inverse dynamics model*. After execution, the SLAM expert estimates real world states $\mathbf{s}_{1:K}$ from the received observations $\mathbf{o}_{1:K}$, which are cached to condition the next dream $\hat{\mathbf{o}}_{K+1:2K}$.

Attention mask. The mask (Figure 3b) preserves temporal order across chunks while allowing joint prediction within a chunk. Clean tokens attend frame-causally to clean tokens of the same type (RGB or state), which keeps the RGB and state histories decoupled. In chunk n, noisy RGB tokens attend to the clean tokens of all earlier chunks and bidirectionally to one another; the past metric states help the generation expert imagine futures that are geometrically consistent with the observed scene. State queries share this visibility and additionally attend to the clean RGB tokens $\mathbf{O}_n$, since the metric states of a chunk are estimated from its own frames. Noisy RGB tokens and state queries do not attend to each other, which prevents information leakage.

3.4 Training

Pretraining: joint teacher-forcing training. We train RGB generation with flow matching (Lipman et al., 2022) and supervise depth and camera poses by regression. Given Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and a flow time $\tau \in [0, 1]$, we construct $\mathbf{O}_n^\tau = (1-\tau)\mathbf{O}_n + \tau\boldsymbol{\epsilon}$ and train the model to predict the corresponding velocity:

$$\mathcal{L}_{\mathrm{FM}} = \mathbb{E}_{\mathbf{O}_n, \boldsymbol{\epsilon}, \tau}\left[\left\| v_\theta(\mathbf{O}_n^\tau, \tau \mid \mathbf{c}_n) - (\boldsymbol{\epsilon} - \mathbf{O}_n)\right\|_2^2\right], \tag{6}$$

where $\mathbf{c}_n$ contains the goal token and the tokens visible to $\mathbf{O}_n^\tau$ under the attention mask. The SLAM expert instead regresses the clean depth latents and metric camera poses of chunk n directly from the state queries. All poses are expressed relative to the first camera. The overall objective is

$$\mathcal{L}_{\mathrm{pre}} = \mathcal{L}_{\mathrm{FM}} + \lambda_d \mathcal{L}_d + \lambda_p \left(\mathcal{L}_{\mathrm{abs}} + \mathcal{L}_{\mathrm{rel}}\right), \tag{7}$$

where $\mathcal{L}_d$ is the mean squared error on depth latents, $\mathcal{L}_{\mathrm{abs}}$ and $\mathcal{L}_{\mathrm{rel}}$ are smooth-L_1 losses on absolute and pairwise relative poses, and λ_d and λ_p balance the terms. Details are provided in section A.1.

Post-training: safety-prioritized reinforcement learning. Imitation pretraining suffers from covariate shift at deployment, and stochastic sampling can yield futures that collide with obstacles. Since the SLAM expert turns each generated video into a metric trajectory, generated futures can be scored against the occupancy maps and expert trajectories of the dataset, without a simulator. We therefore post-train the generation expert with DiffusionNFT (Zheng et al., 2026): for each history rolled out by the model itself and goal, a group of G RGB rollouts is sampled, and the SLAM expert recovers their trajectories $\boldsymbol{\xi}_{1:G}$. We adapt the reward of CompassNav (Li et al., 2026b) with a safety

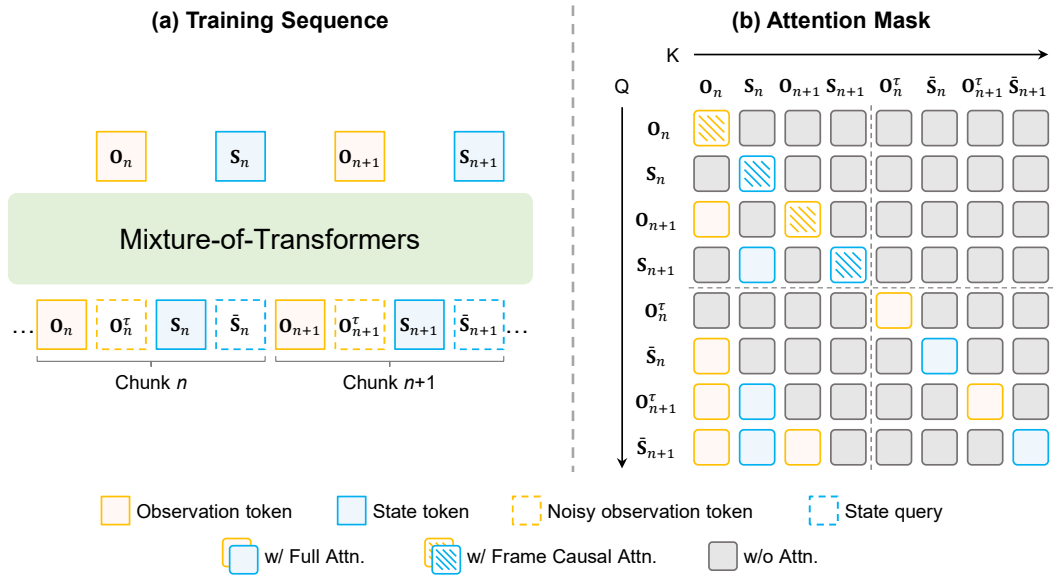

Figure 3: **Training sequence and attention mask.** (a) Each chunk n consists of clean tokens $\mathbf{O}_n, \mathbf{S}_n$ and prediction tokens $\mathbf{O}_n^\tau, \bar{\mathbf{S}}_n$, and all chunks of an episode are processed by the Mixture-of-Transformers in a single forward pass. (b) Attention mask. Temporal order is preserved across chunks, and observation and state tokens in the history context are invisible to each other. No prediction token can attend to its own target.

gate, so that safety takes strict priority:

$$r_i = \frac{s_i \exp(-C_i/T)}{\sum_{j=1}^{G} s_j \exp(-C_j/T)}, \tag{8}$$

where r_i is the reward of the i-th rollout, $s_i \in \{0, 1\}$ indicates whether $\boldsymbol{\xi}_i$ is collision-free and T is a temperature. The cost C_i is the mean squared error between the per-step geodesic progress toward the goal along $\boldsymbol{\xi}_i$ and along the expert trajectory. Measuring progress rather than spatial deviation rewards safe routes that approach the goal at the expert's pace, even when they depart from the expert path. Only the self-attention LoRA (Hu et al., 2021) adapters of the generation expert are updated; the SLAM expert stays frozen so the reward cannot be increased by updating the trajectory estimator itself. Details are provided in the Appendix A.2.

3.5 Autoregressive Closed-Loop Inference

The model replans as new observations arrive after execution. At each planning step, it generates a goal-conditioned RGB sequence, from which the SLAM expert directly regresses future depth and poses in a single forward pass per chunk, without iterative denoising. A low-level controller tracks a short prefix of the pose trajectory, after which the newly observed frames are appended to the history. The local backend then re-estimates the depth and poses of all observed frames in the current chunk and replaces their previous estimates, while their RGB tokens remain unchanged in the KV cache. Earlier estimates in a chunk are thus refined as more of its frames arrive, and remain fixed once the chunk is complete. Generated frames are discarded after each planning step, so the persistent history contains only observed frames and their estimated depth and poses. The next prediction uses this updated history and reuses the KV cache across planning steps.

4 Experimental Evaluation

We evaluate WSM on closed-loop start–goal navigation (Section 4.2) and on SLAM, measuring camera pose estimation and dense reconstruction on held-out trajectories (Section 4.3). Section 4.4

Table 1: **Navigation results on the InternVLA-N1 start–goal navigation benchmark.** SR and SPL are reported in percent. Checkmarks indicate the input modalities. Baseline results are taken from LoGoPlanner. † means our re-evaluation with its latest official code.

Method	Input		Home		Commercial	
	RGB	Depth	SR↑	SPL↑	SR↑	SPL↑
DD-PPO (Wijmans et al., 2019)	✓	✓	0.4	0.4	5.3	5.2
iPlanner (Yang et al., 2023)	–	✓	43.0	40.6	54.6	52.8
ViPlanner (Roth et al., 2024)	✓	✓	45.0	43.2	63.7	61.9
LoGoPlanner (Peng et al., 2025)	✓	✓	57.3	52.4	67.1	63.9
LoGoPlanner† (Peng et al., 2025)	✓	✓	59.6	53.6	65.8	62.0
World SLAM Model	✓	–	**70.3**	**66.3**	**73.9**	**71.8**

ablates two components: the SLAM expert, by comparing WSM with IDM (w/o SLAM), which decodes camera motion from dreamed frames without estimating depth or localizing observed frames, and the local backend. Appendix B.2 compares the model before and after post-training.

4.1 Experimental Setup

Training data and details. We train on InternVLN–N1 (InternNav Team, 2025) datasets and reserve 5% of the trajectories as the SLAM test split. These test trajectories are excluded from both pre-training and post-training. All experts are initialized from Wan2.2-5B (Wan et al., 2025). The generation expert directly inherits the pretrained DiT weights with a hidden width of 3072, while the depth and pose experts are initialized by linearly interpolating the pretrained weights to hidden widths of 1536 and 768, respectively LingBot-VA. All training is conducted on 48 A100 GPUs. Teacher-forcing pretraining runs for 10 epochs with a global batch size of 48, and the chunk size is randomly sampled from 1 to 4 latent frames. We use AdamW with a weight decay of 0.01, linearly warming up the learning rate to 3×10^{-5} over the first 5% of steps and then decaying it to 3×10^{-6} with a cosine schedule. Post-training runs for one epoch with the same learning rates, and a weight decay of 10^{-4}. More details are provided in Appendix B.

Inference details. We generate $K = 4$ future RGB latents at 320×192 resolution using 8 sampling steps. Each latent frame yields $M = 4$ poses, giving 16 predicted poses per planning step. The controller executes to the fourth predicted pose before replanning from four new RGB observations, and the KV cache retains up to 128 observed latent frames. All inference is performed on a single NVIDIA RTX 4090 GPU.

4.2 Navigation Evaluation

Following LoGoPlanner (Peng et al., 2025), we evaluate WSM on the InternVLA–N1 (InternNav Team, 2025) start–goal benchmark in a closed-loop setting. The benchmark comprises 40 InternScenes environments (20 Home and 20 Commercial) with 100 start–goal episodes each, all disjoint from training datasets.

The goal of each episode is specified as a position in the robot's initial frame and remains fixed as the robot moves. The model therefore has to track its own pose relative to the initial frame and decide when to stop. WSM takes only RGB observations as input and estimates camera poses and depth with its SLAM expert, without external odometry or post-processing. Following the benchmark protocol, an episode is successful if the robot decelerates or stops within 1 m of the goal. We report Success Rate (SR) and Success weighted by Path Length (SPL).

Table 1 compares WSM with DD-PPO (Wijmans et al., 2019), iPlanner (Yang et al., 2023), ViPlanner (Roth et al., 2024), and LoGoPlanner. For LoGoPlanner, we report both its published results and our re-evaluation with its latest official code.

WSM achieves the highest SR in both Home and Commercial environments, improving over LoGoPlanner by 10.7 and 8.1 percentage points, respectively. As shown in Figure 4, WSM reaches

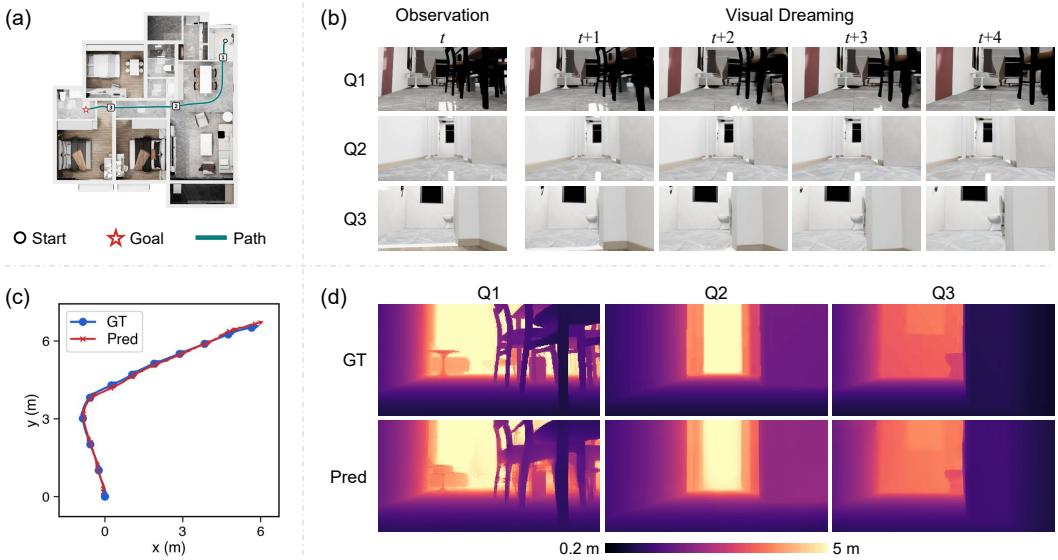

Figure 4: **Visualization of navigation results in simulation.** (a) Navigation result; Q1–Q3 mark the queries shown in (b) and (d). (b) RGB observation at time t and the frames dreamed by WSM for $t{+}1$ to $t{+}4$ at each query. (c) Camera trajectory estimated by the SLAM expert vs. ground truth. (d) Ground-truth and predicted depth at time t for each query, colored from 0.2 to 5 m.

Table 2: **SLAM evaluation on the 5% InternVLN-N1 test split.** † denotes the released checkpoint; ‡ denotes the model fine-tuned on the 95% InternVLN-N1 training split.

Method	ATE RMSE (m)↓	Acc. RMSE (m)↓
SLAM-Former† (Yuan et al., 2026b)	0.051	0.057
SLAM-Former‡ (Yuan et al., 2026b)	0.041	**0.041**
World SLAM Model	**0.021**	0.062

the goal in an unseen Home scene, and its estimated camera trajectory and predicted depth closely match the ground truth. We attribute this to jointly modeling visual dreaming and SLAM in a single model, where localization and mapping from RGB observations provide the pose and geometry required for goal reaching without external odometry.

4.3 SLAM Evaluation

We evaluate camera pose estimation and dense 3D reconstruction on all RGB trajectories in the 5% InternVLN-N1 test split and compare WSM with SLAM-Former (Yuan et al., 2026b). For the comparison reported here, WSM is trained on the InternVLN-N1 navigation dataset alone, whereas SLAM-Former starts from a model pretrained on large-scale 3D datasets and is then fine-tuned on the same InternVLN-N1 training split. Both the released SLAM-Former checkpoint and its fine-tuned version are evaluated on the 5% test split.

The input resolutions are 320×192 for WSM and 518×518 for SLAM-Former. Camera pose estimation is measured by camera-center ATE RMSE after per-sequence $\mathrm{Sim}(3)$ alignment, while dense reconstruction is evaluated by accuracy (Acc.), the RMSE of the distances from predicted points to their nearest ground-truth points, after alignment and rigid ICP refinement.

As shown in Table 2, WSM achieves an ATE RMSE of 0.021 m, compared with 0.041 m for the fine-tuned SLAM-Former. For dense reconstruction, WSM obtains an Acc. RMSE of 0.062 m, which is worse than the 0.041 m of the fine-tuned SLAM-Former. This gap may result from encoding depth in the appearance-oriented VAE latent space.

Table 3: **SLAM IDM ablation.** Both models are evaluated after pretraining, without post-training. SR and SPL are reported in percent.

Method	Home		Commercial	
	SR↑	SPL↑	SR↑	SPL↑
IDM (w/o SLAM)	28.1	27.1	32.1	31.2
World SLAM Model (w/o post-training)	**64.6**	**61.5**	**68.5**	**66.9**

Table 4: **Local backend ablation.** Both variants are post-trained from the same pretrained model and differ only in whether the local backend is used in history rollout, during both post-training and closed-loop evaluation. SR and SPL are reported in percent.

Method	Home		Commercial	
	SR↑	SPL↑	SR↑	SPL↑
w/o local backend	68.3	65.2	72.6	70.5
w/ local backend	**70.3**	**66.3**	**73.9**	**71.8**

4.4 Ablation Studies

SLAM IDM. Existing world action models (Li et al., 2026a) dream future frames and decode actions from the dreamed transitions through inverse dynamics. IDM (w/o SLAM) follows this design: it removes the depth expert and all depth and pose estimation for observed frames, and its pose expert only performs inverse dynamics, generating future camera poses from the observed and dreamed RGB, conditioned on the goal and on previously issued poses. Both models share the same initialization and training data and are evaluated after pretraining.

As shown in Table 3, IDM (w/o SLAM) trails WSM without post-training by about 36 SR points in both Home and Commercial. This gap likely reflects IDM's less reliable localization and scene-geometry estimation, which hinder planning toward a goal fixed in the first-camera frame. WSM uses a shared SLAM expert to localize and map observed frames and perform inverse dynamics on dreamed frames in the same metric coordinates, keeping the plan anchored to the goal.

Local backend. The local backend in eq. (4) refines previous states. We compare it with incremental estimation alone, where each state is estimated once by eq. (2) and then kept fixed. Both variants are post-trained from the same pretrained model, each using its own state update in both post-training and evaluation. As shown in Table 4, the local backend improves SR/SPL by 2.0/1.1 on Home and by 1.3/1.3 on Commercial, indicating that refining recent states reduces error accumulation in the history and leads to more reliable closed-loop navigation.

5 Conclusion

We presented the World SLAM Model, in which the generation expert dreams goal-conditioned futures and the SLAM expert estimates world states from both observed and dreamed frames, maintaining one persistent world state in a closed loop. Using RGB input only, WSM achieves the highest SR and SPL on start-goal navigation among the evaluated methods while retaining accurate spatial estimation. These results indicate that navigation benefits not only from what SLAM estimates but also from how it maintains and refines a world state, highlighting SLAM as an intrinsic mechanism for long-horizon world modeling and embodied interaction.

WSM encodes depth in the Wan2.2 VAE latent space, which is designed for appearance rather than geometry, trading geometric fidelity for compatibility with the pretrained video generator. Its history KV cache is also kept without compression, so memory and computation grow with the number of observed frames, limiting long-horizon deployment. Future work includes geometry-native latent representations for depth and efficient history compression.

AI USE STATEMENT

In this work, we used generative AI tools for none of the required disclosure. We have not used generative AI tools for all of the required disclosure, and all of the required disclosure are not applicable to this work. Additionally, we used generative AI tools for polishing paper writing. We have reviewed all AI-assisted work. We use AI tools to correct the input sentence grammar and refine the writing to be formal and readable. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Niket Agarwal, Arslan Ali, Jon Allen, Martin Antolini, Adeline Aubame, Alisson Azzolini, Junjie Bai, Maciej Bala, Yogesh Balaji, Josh Bapst, et al. Cosmos 3: Omnimodal world models for physical ai. *arXiv preprint arXiv:2606.02800*, 2026.

Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15791–15801, 2025. doi: 10.1109/CVPR52734.2025.01472.

Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics*, 32(6):1309–1332, 2016.

Wenzhe Cai, Jiaqi Peng, Yuqiang Yang, Yujian Zhang, Meng Wei, Hanqing Wang, Yilun Chen, Tai Wang, and Jiangmiao Pang. NavDP: Learning sim-to-real navigation diffusion policy with privileged information guidance, 2025. URL https://arxiv.org/abs/2505.08712.

Carlos Campos, Richard Elvira, Juan J. Gómez Rodríguez, José M. M. Montiel, and Juan D. Tardós. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021. doi: 10.1109/TRO.2021.3075644.

Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. *arXiv preprint arXiv:2004.05155*, 2020.

Lin-Zhuo Chen, Jian Gao, Yihang Chen, Ka Leong Cheng, Yipengjing Sun, Liangxiao Hu, Nan Xue, Xing Zhu, Yujun Shen, Yao Yao, et al. Geometric context transformer for streaming 3d reconstruction. *arXiv preprint arXiv:2604.14141*, 2026.

An-Chieh Cheng, Yandong Ji, Zhaojing Yang, Xueyan Zou, Jan Kautz, Erdem Biyik, Hongxu Yin, Sifei Liu, and Xiaolong Wang. Navila: Legged robot vision-language-action model for navigation. In *RSS*, 2025.

Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. Monoslam: Real-time single camera slam. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):1052–1067, 2007.

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. *arXiv preprint arXiv:2106.09685*, 2021.

InternNav Team. InternVLA-N1: An open dual-system navigation foundation model with learned latent plans, 2025.

Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Fei Han, Mingrui Yu, Zelin Gao, Nan Xue, Xing Zhu, Yujun Shen, and Yinghao Xu. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026a.

LinFeng Li, Jian Zhao, Yuan Xie, Xin Tan, and Xuelong Li. Compassnav: Steering from path imitation to decision understanding in navigation. In *International Conference on Learning Representations*, volume 2026, pp. 44744–44768, 2026b.

Weixin Liang, LILI YU, Liang Luo, Srini Iyer, Ning Dong, Chunting Zhou, Gargi Ghosh, Mike Lewis, Wen tau Yih, Luke Zettlemoyer, and Xi Victoria Lin. Mixture-of-transformers: A sparse and scalable architecture for multi-modal foundation models. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=Nu6N69i8SB.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Eitan Marder-Eppstein, Eric Berger, Tully Foote, Brian Gerkey, and Kurt Konolige. The office marathon: Robust navigation in an indoor office environment. In *2010 IEEE international conference on robotics and automation*, pp. 300–307. IEEE, 2010.

Jiaqi Peng, Wenzhe Cai, Yuqiang Yang, Tai Wang, Yuan Shen, and Jiangmiao Pang. LoGoPlanner: Localization grounded navigation policy with metric-aware visual geometry, 2025. URL https://arxiv.org/abs/2512.19629.

Pascal Roth, Julian Nubert, Fan Yang, Mayank Mittal, and Marco Hutter. Viplanner: Visual semantic imperative learning for local navigation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5243–5249. IEEE, 2024.

Dhruv Shah, Ajay Sridhar, Nitish Dashora, Kyle Stachowicz, Kevin Black, Noriaki Hirose, and Sergey Levine. Vint: A foundation model for visual navigation. *arXiv preprint arXiv:2306.14846*, 2023.

Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in neural information processing systems*, 34:16558–16569, 2021.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5294–5306. IEEE, 2025.

Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. π^3: Permutation-equivariant visual geometry learning. In *International Conference on Learning Representations (ICLR)*, 2026a.

Zihan Wang, Seungjun Lee, and Gim Hee Lee. Dynam3d: Dynamic layered 3d tokens empower vlm for vision-and-language navigation. *Advances in Neural Information Processing Systems*, 38:153522–153544, 2026b.

Meng Wei, Chenyang Wan, Xiqian Yu, Tai Wang, Yuqiang Yang, Xiaohan Mao, Chenming Zhu, Wenzhe Cai, Hanqing Wang, Yilun Chen, et al. Streamvln: Streaming vision-and-language navigation via slowfast context modeling. *arXiv preprint arXiv:2507.05240*, 2025.

Stephan Weiss, Davide Scaramuzza, and Roland Siegwart. Monocular-slam–based navigation for autonomous micro helicopters in gps-denied environments. *Journal of Field Robotics*, 28(6): 854–874, 2011.

Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. Dd-ppo: Learning near-perfect pointgoal navigators from 2.5 billion frames. *arXiv preprint arXiv:1911.00357*, 2019.

World Labs Team. Atlas: A world model for spatial intelligence. *World Labs Blog*, 2026. https://www.worldlabs.ai/blog/atlas.

Fan Yang, Chen Wang, Cesar Cadena, and Marco Hutter. iplanner: Imperative path planning. *arXiv preprint arXiv:2302.11434*, 2023.

Hang Yin, Xiuwei Xu, Linqing Zhao, Ziwei Wang, Jie Zhou, and Jiwen Lu. Unigoal: Towards universal zero-shot goal-oriented navigation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 19057–19066. IEEE, 2025.

Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-WAM: Do world action models need test-time future imagination?, 2026a. URL https://arxiv.org/abs/2603.16666.

Yijun Yuan, Zhuoguang Chen, Kenan Li, Weibang Wang, Minghui Qin, Zhijian Fang, Weicheng Zheng, and Hang Zhao. Slam-former: Putting slam into one transformer. In *European Conference on Computer Vision*, pp. 56–74. Springer, 2026b.

Shuang Zeng, Dekang Qi, Xinyuan Chang, Feng Xiong, Shichao Xie, Xiaolong Wu, Shiyi Liang, Mu Xu, and Xing Wei. Janusvln: Decoupling semantics and spatiality with dual implicit memory for vision-language navigation. *arXiv preprint arXiv:2509.22548*, 2025.

Chuhan Zhang, Guillaume Le Moing, Skanda Koppula, Ignacio Rocco, Liliane Momeni, Junyu Xie, Shuyang Sun, Rahul Sukthankar, Joëlle K. Barral, Raia Hadsell, Zoubin Ghahramani, Andrew Zisserman, Junlin Zhang, and Mehdi S. M. Sajjadi. Efficiently reconstructing dynamic scenes one d4rt at a time. In *CVPR*, 2026.

Jiazhao Zhang, Kunyu Wang, Rongtao Xu, Gengze Zhou, Yicong Hong, Xiaomeng Fang, Qi Wu, Zhizheng Zhang, and He Wang. Navid: Video-based vlm plans the next step for vision-and-language navigation. *Robotics: Science and Systems*, 2024.

Kaiwen Zheng, Huayu Chen, Haotian Ye, Haoxiang Wang, Qinsheng Zhang, Kai Jiang, Hang Su, Stefano Ermon, Jun Zhu, and Ming-Yu Liu. Diffusionnft: Online diffusion reinforcement with forward process. In *International Conference on Learning Representations*, volume 2026, pp. 134129–134150, 2026.

Dong Zhuo, Wenzhao Zheng, Jiahe Guo, Yuqi Wu, Jie Zhou, and Jiwen Lu. Streaming visual geometry transformer. In *International Conference on Learning Representations*, volume 2026, pp. 88055–88072, 2026.

A METHOD DETAILS

A.1 PRETRAINING OBJECTIVE

RGB generation. The RGB loss follows eq. (6). The flow time is sampled per chunk and shifted as $\tau = \gamma\sigma/(1 + (\gamma - 1)\sigma)$ with $\sigma \sim \mathcal{U}(0, 1)$, and the loss is weighted by a bell-shaped function of τ. The first chunk of each training sequence serves as a clean visual anchor and is excluded from the loss.

Depth. Metric depth d is clipped to $[d_{\min}, d_{\max}] = [0.2, 5]$ m and normalized in log space,

$$u = \frac{\log(d/d_{\min})}{\log(d_{\max}/d_{\min})} \in [0, 1]. \tag{9}$$

The map $2u - 1$ is replicated to three channels and encoded by the Wan2.2 VAE, and $\mathcal{L}_d$ is the mean squared error on the resulting depth latents after per-channel standardization. At inference, metric depth is recovered by decoding the predicted latents and inverting the log mapping.

Camera pose. Following SLAM-Former (Yuan et al., 2026b), we supervise camera poses with the smooth-L_1 (Huber) norm $\|\cdot\|_\delta$, applied to the translation and the unit-quaternion rotation of a pose difference. Since WSM predicts metric poses in the first-camera frame, no scale alignment is applied, and both absolute and pairwise relative poses are supervised:

$$\mathcal{L}_{\text{abs}} = \sum_j \big\|\hat{\mathbf{g}}_j - \mathbf{g}_j\big\|_\delta, \qquad \mathcal{L}_{\text{rel}} = \sum_{i,j} \big\|\hat{\mathbf{g}}_i^{-1}\hat{\mathbf{g}}_j - \mathbf{g}_i^{-1}\mathbf{g}_j\big\|_\delta, \tag{10}$$

where the sums run over all raw frames of a training sequence and are normalized by the number of terms.

A.2 POST-TRAINING

Reward. For each dream $\hat{\mathbf{O}}^{(i)}$, the SLAM expert recovers $L = KM$ camera positions $\mathbf{x}_{i,1:L}$, which are placed in the occupancy map through the dataset pose $\mathbf{x}_0$ of the query frame. Let $D(\mathbf{x})$ be the geodesic distance from $\mathbf{x}$ to the goal region of radius r_g, and let $\Delta_{i,l} = D(\mathbf{x}_{i,l-1}) - D(\mathbf{x}_{i,l})$ be the progress of step l, with $\mathbf{x}_{i,0} = \mathbf{x}_0$. The cost in eq. (8) is

$$C_i = \frac{1}{L}\sum_{l=1}^{L} \big(\Delta_{i,l} - \Delta_l^\star\big)^2, \tag{11}$$

where $\Delta_l^\star$ is the progress of the expert trajectory at the same time step. The safety indicator $s_i = 1$ if the robot footprint along the trajectory stays in free space. In addition, the safe dream with the lowest cost receives a bonus that grows with its cost margin over the second best.

Objective. Within each group, rewards are standardized into advantages A_i and mapped to optimality probabilities

$$\rho_i = \frac{1}{2} + \frac{1}{2}\,\mathrm{clip}\big(A_i/A_{\max}, -1, 1\big). \tag{12}$$

Following DiffusionNFT (Zheng et al., 2026), let v^{old} be the old policy that samples the dreams, an exponential moving average of the trained policy v_θ, and define the implicit positive and negative policies $v^+ = (1-\beta)v^{\text{old}} + \beta v_\theta$ and $v^- = (1+\beta)v^{\text{old}} - \beta v_\theta$. For a dream noised as $(1-\tau)\hat{\mathbf{O}}^{(i)} + \tau\epsilon$, we minimize

$$\mathcal{L}_{\text{post}} = \mathbb{E}\Big[\rho_i\big\|v^+ - (\epsilon - \hat{\mathbf{O}}^{(i)})\big\|_2^2 + (1-\rho_i)\big\|v^- - (\epsilon - \hat{\mathbf{O}}^{(i)})\big\|_2^2\Big] + \lambda_{\text{ref}}\big\|v_\theta - v^{\text{ref}}\big\|_2^2, \tag{13}$$

where v^{ref} is the pretrained model, and the two error terms use the adaptive normalization of DiffusionNFT. High-reward dreams pull v_θ toward their flow-matching targets, and low-reward dreams push it away. The loss is defined on the forward process and uses only the clean dreams, so it requires neither likelihoods nor the sampling trajectory. Algorithm 1 summarizes the procedure.

Algorithm 1 Post-training of WSM

Require: pretrained generation expert v_θ and SLAM expert f_ψ, queries $\mathcal{Q}$, group size G
1: Add LoRA adapters to the self-attention layers of v_θ and freeze all other parameters
2: $v^{\text{old}} \leftarrow v_\theta$
3: **for** each query in $\mathcal{Q}$ **do**
4: Sample G dreams $\hat{\mathbf{O}}^{(1:G)}$ from v^{old} given the history and the goal
5: Recover their trajectories with f_ψ
6: Compute s_i, C_i, r_i, and ρ_i for each dream
7: Update the LoRA adapters by minimizing $\mathcal{L}_{\text{post}}$ in eq. (13)
8: Update v^{old} as a moving average of v_θ
9: **end for**

B Experimental Details

B.1 Implementation Details

Architecture. The three experts share 30 transformer layers with joint self-attention. The generation expert keeps the width of Wan2.2-5B, with hidden size 3072, FFN size 14336, and 24 attention heads. The depth and pose experts have hidden sizes 1536 and 768 and FFN sizes 7168 and 3584, and are initialized by linearly interpolating the pretrained weights to these widths. RGB and depth latents are patchified with patch size $(1, 2, 2)$.

Pretraining. Table 5 lists the pretraining hyperparameters. We set $\lambda_d = 1$, $\lambda_p = 0.5$ in eq. (7). The generation expert and the goal embedding use 0.3 times the base learning rate.

Post-training. Table 6 lists the post-training hyperparameters. We post-train on about 10% of the training trajectories, selected as hard examples. Starting from the pretrained model, we add LoRA adapters with rank 32 and $\alpha = 64$ to the query, key, value, and output projections of the self-attention layers in the generation expert. For each query, we sample $G = 16$ dreams of $K = 4$ latent frames with 8 sampling steps and draw 4 flow times per dream. The reward uses $T = 0.01\,\text{m}^2$ and $r_g = 0.2\,\text{m}$, and the objective uses $\beta = 0.1$, $A_{\max} = 5$, and $\lambda_{\text{ref}} = 10^{-4}$.

Table 5: **Pretraining hyperparameters.**

Hyperparameter	Value
Initialization	Wan2.2-5B
Input resolution	320×192
Chunk size (latent frames)	uniform in $\{1, \ldots, 4\}$
Timestep shift γ	5
λ_d / λ_p / δ	1 / 0.5 / 0.1
Optimizer	AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.95$, $\epsilon = 10^{-8}$)
Weight decay	0.01
Peak / final learning rate	3×10^{-5} / 3×10^{-6}
LR multiplier (generation expert, goal embedding)	0.3
Schedule	5% linear warmup, cosine decay
Gradient clipping	2.0
Epochs	10
Global batch size	48
Precision	bfloat16
Hardware	48 A100 GPUs

B.2 Two-Stage Ablation

Figure 5 compares the pretrained model with the final model after post-training under the same inference settings. Post-training improves SR by 5.7 and 5.4 points and SPL by 4.8 and 4.9 points on Home and Commercial, respectively. Since only the generation expert is updated, the gains mainly reflect a change in which futures are dreamed: the reward favors dreams whose recovered trajectories

Table 6: **Post-training hyperparameters.**

Hyperparameter	Value
Initialization	pretrained model
Trainable parameters	self-attention LoRA of the generation expert
LoRA rank / α	32 / 64
Group size G	16
Dreamed latent frames K	4
Sampling steps	8
Flow times per dream	4
Timestep shift γ	5
Temperature T / goal radius r_g	0.01 m^2 / 0.2 m
β / $A_{\max}$ / λ_{ref}	0.1 / 5 / 10^{-4}
Optimizer	AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$)
Weight decay	10^{-4}
Peak / final learning rate	3×10^{-5} / 3×10^{-6}
Schedule	3% linear warmup, cosine decay
Gradient clipping	1.0
Epochs	1
Precision	bfloat16
Hardware	48 A100 GPUs

are collision-free and approach the goal at the expert's pace, which teacher-forcing pretraining does not optimize explicitly.

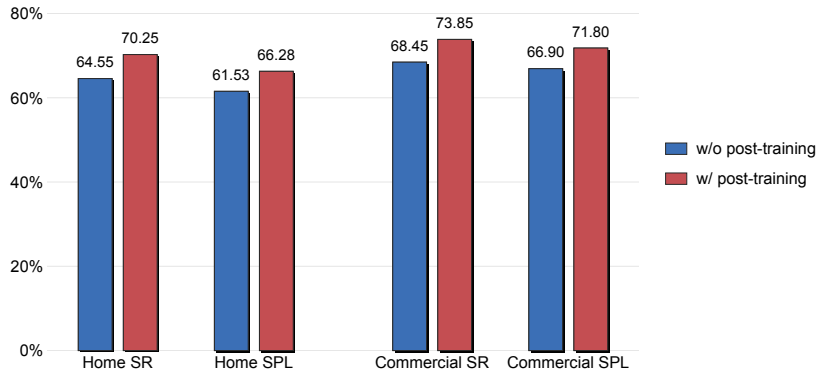

Figure 5: **Two-stage ablation.** SR and SPL are reported in percent.

B.3 Additional Qualitative Results

Figures 6–9 show additional closed-loop navigation results in simulation, with panels as in Figure 4. The four episodes are taken from two Home scenes of the InternVLA-N1 start–goal benchmark.

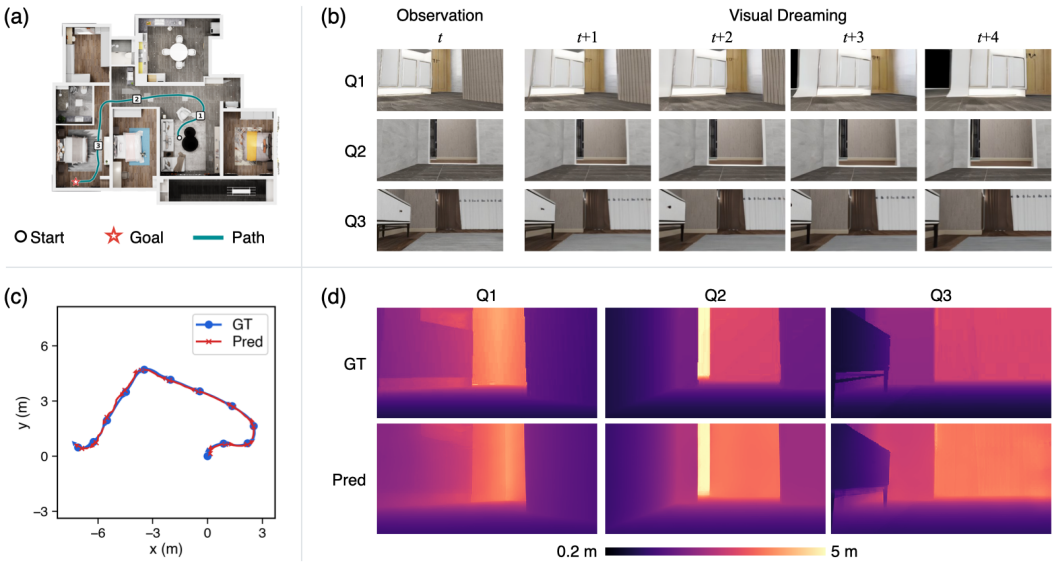

Figure 6: **Navigation result in Home scene 15, episode 27.**

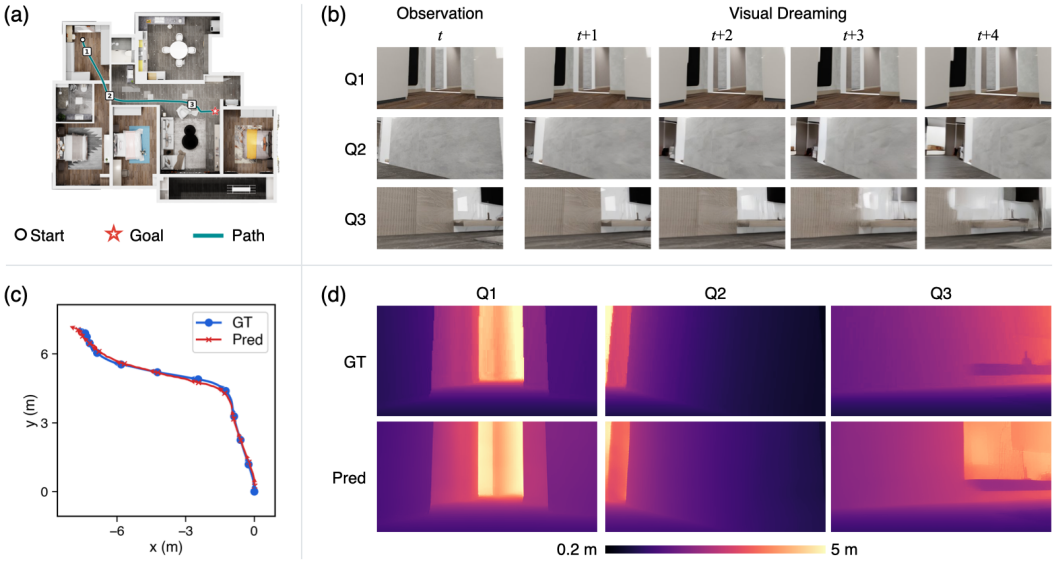

Figure 7: **Navigation result in Home scene 15, episode 79.**

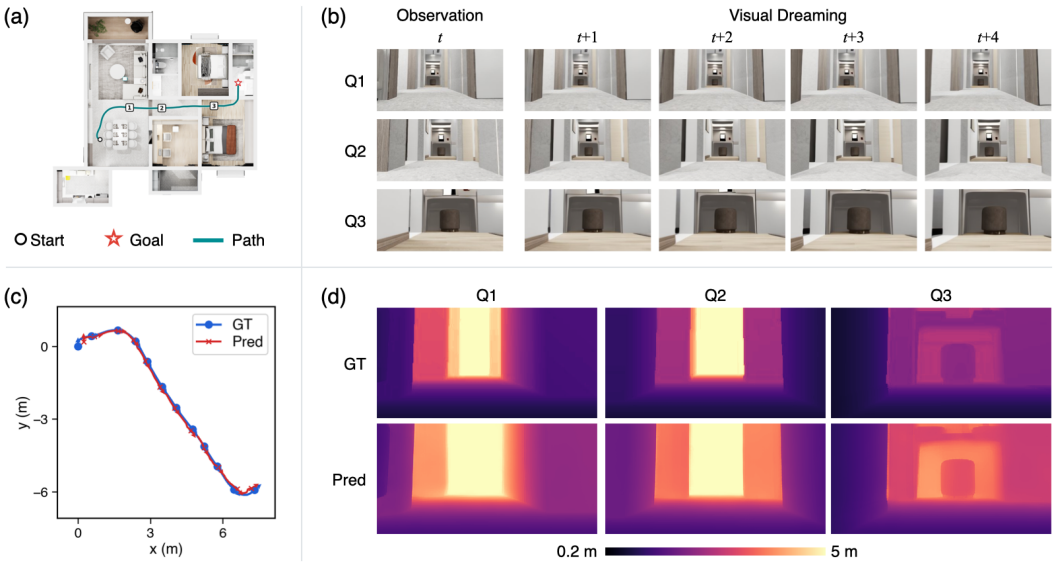

Figure 8: **Navigation result in Home scene 16, episode 94.**

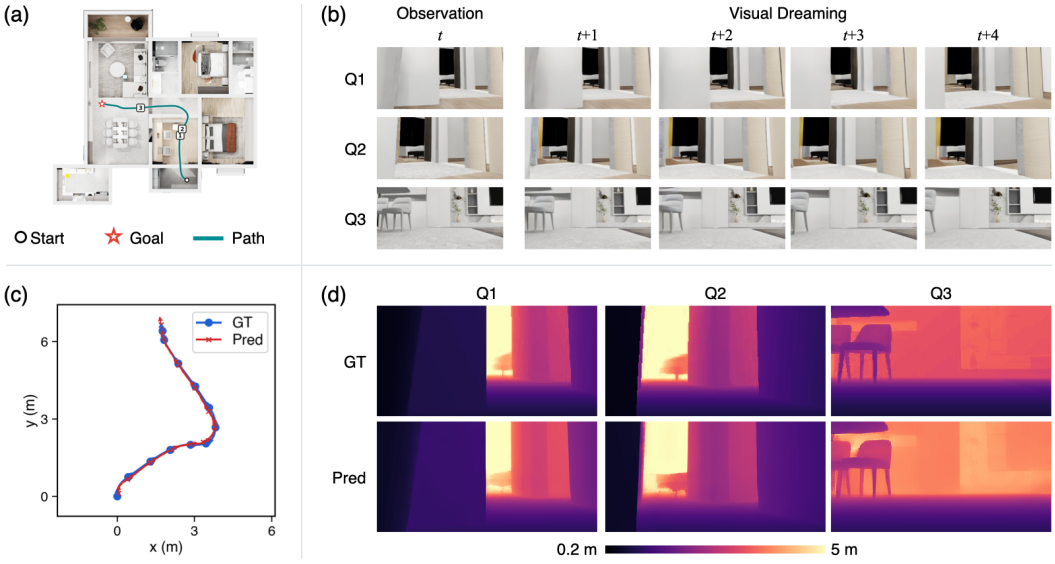

Figure 9: **Navigation result in Home scene 16, episode 96.**