POSEVLN: LEARNING GEOMETRIC STATE EVOLUTION FOR VISION-LANGUAGE NAVIGATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-Language Navigation (VLN) requires an embodied agent to relate past experience, current spatial state, and future motion while following natural-language instructions. Recent approaches improve these capabilities through long-horizon memory, state-aware reasoning, and future prediction, yet these components are often modeled separately, leaving navigation state evolution insufficiently connected across time. We argue that camera pose provides a natural geometric representation for bridging past, present, and future states. Based on this insight, we propose PoseVLN, a pose-aware VLN framework that jointly learns navigation decisions and geometric state evolution. PoseVLN organizes visual memory and the current observation in an episode-relative coordinate frame, and jointly predicts the next K navigation actions together with a local ΔPose over the same horizon. The predicted ΔPose is composed with the current trajectory-level pose to obtain the future pose state, which is recursively propagated throughout navigation, forming a continuous geometric state evolution from past memory to future motion. Experiments on the R2R-CE and RxR-CE Val-Unseen splits demonstrate the effectiveness of explicitly modeling geometric state evolution, with PoseVLN outperforming the strongest compared baselines by 2.6 and 2.9 percentage points in success rate, respectively, using only single RGB observations.

1 INTRODUCTION

Vision-Language Navigation (VLN) requires an embodied agent to follow natural-language instructions and navigate to a target by interpreting visual observations and taking appropriate actions. Recent advances have extended VLN beyond direct instruction and observation matching, with increasing attention to memory, spatial reasoning, and predictive modeling. Memory-based approaches preserve information from previous observations and decisions (Wei et al., 2025; Song et al., 2025; Zhang et al., 2025a; Xue et al., 2026; Liu et al., 2026), while state-aware and geometric methods support navigation through progress reasoning, structured scene representations, or explicit spatial features (Wang et al., 2026b; 2025c; Li et al., 2026a; Zhang et al., 2025a; Xue et al., 2026). Predictive approaches further anticipate future actions, observations, landmarks, or latent dynamics before execution (Yao et al., 2025; Dai et al., 2026; Huang et al., 2026; Xu et al., 2025; Wang et al., 2026a; 2025b; Zhang et al., 2026). Together, these directions highlight the importance of relating past experience, the current navigation context, and future motion.

Despite these advances, two aspects of spatial modeling remain insufficiently addressed. First, past and current observations are not necessarily anchored in an explicit common spatial reference. Retrieving a relevant memory therefore does not by itself establish its viewpoint's position and orientation relative to the agent, leaving this relationship to be inferred for spatial decision-making. Second, future prediction does not necessarily specify an explicit geometric transition of the navigation state. Anticipated actions or visual content can guide action selection without providing an updated position and orientation for subsequent decisions. These gaps motivate a common geometric representation that explicitly relates past observations to the current state and describes how that state changes over the upcoming action horizon.

Camera pose provides a compact representation for addressing both needs. A trajectory-level pose specifies the position and orientation of the camera in a shared coordinate frame, associating each past and current observation with the spatial state from which it was acquired. Relative poses be-

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.

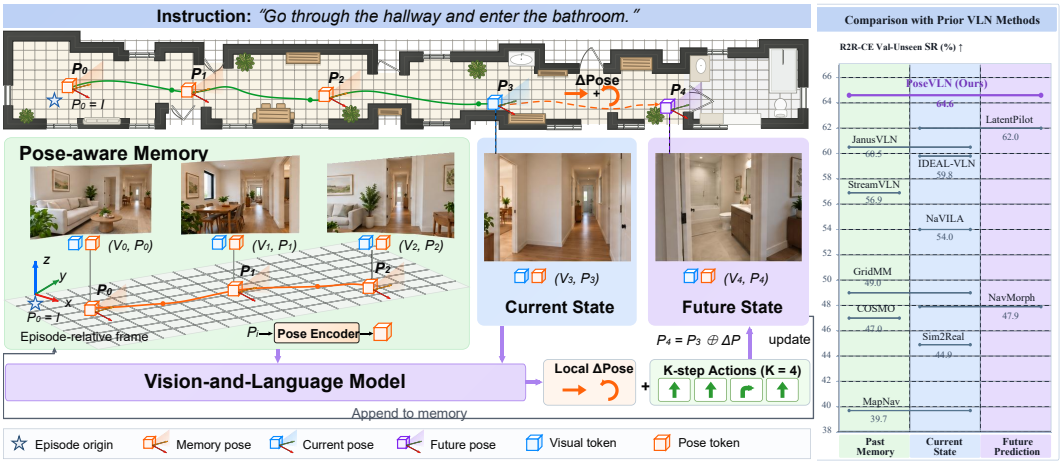

Figure 1: The PoseVLN framework and comparison with prior methods. Left: pose-aware memory and geometric state evolution. Right: comparison with prior VLN methods by navigation performance and temporal focus.

tween these observations then make their spatial relationships explicit. For future motion, a local pose change describes the translation and rotation over the upcoming action horizon. Composing this change with the current pose yields a future pose in the same coordinate frame, allowing the resulting state to be carried forward and associated with subsequent observations. Together, trajectory-level poses and local pose changes provide a common geometric description of past observations, the current viewpoint, and future motion. We refer to this continual maintenance and update of the agent's spatial state as geometric state evolution.

Based on this insight, we introduce PoseVLN, a pose-aware VLN framework that explicitly models navigation as geometric state evolution in a trajectory-level coordinate space. We define the first observation as the origin of an episode-relative coordinate system and associate each subsequent visual observation in memory with a trajectory-level pose state, allowing past observations and the current state to be organized within a common spatial reference. As illustrated in Figure 1, given the instruction, current observation, visual memory, and their associated pose states, PoseVLN jointly predicts the next K navigation actions and a local ΔPose that describes the geometric state transition over the same action horizon. The predicted ΔPose is composed with the current trajectory-level pose to obtain the future pose state, which is stored back into memory and reused in subsequent navigation. During training, ground-truth trajectory poses provide supervision for geometric state transitions. At inference, PoseVLN operates solely on a monocular RGB stream, without depth or external odometry. The first observation is assigned the identity pose $P_0 = I$ to define the episode-relative frame. No ground-truth pose is used, and all subsequent pose states are recursively predicted by the model. Our main contributions are three-fold:

- We propose PoseVLN, a pose-aware VLN framework that models geometric state evolution through joint action and local ΔPose prediction, recursively propagating poses without step-wise ground-truth pose or odometry at inference.
- We demonstrate strong performance on R2R-CE and RxR-CE, showing that geometric state evolution improves long-horizon navigation using only monocular RGB observations.
- We conduct comprehensive ablations and pose evaluations to analyze the relationship between pose estimation and navigation, providing new insights into their interplay.

2 Related Work

2.1 Memory and State-Aware Navigation

Video-based approaches such as NaVid (Zhang et al., 2024b) and Uni-NaVid (Zhang et al., 2024a) demonstrate the effectiveness of VLM/VLA models for continuous navigation. To support longer

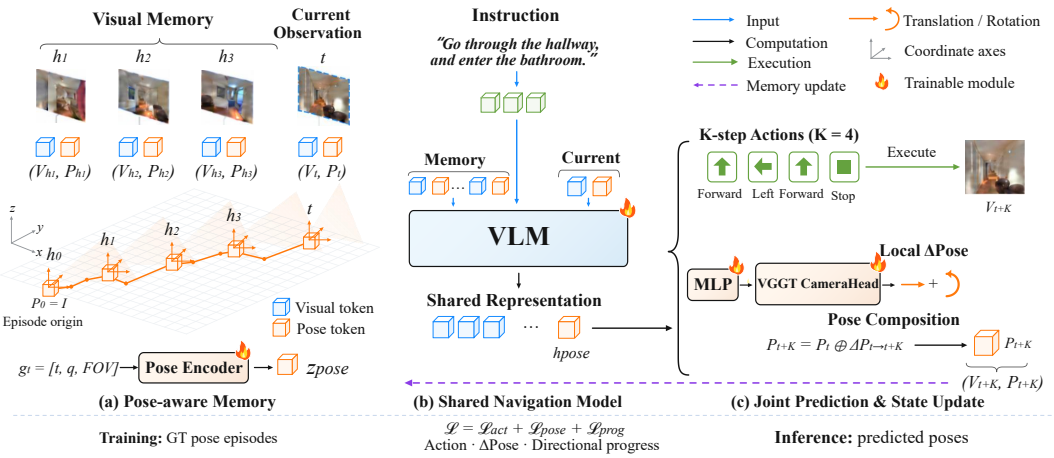

Figure 2: Overview of PoseVLN. Pose-aware memory associates observations with trajectory-level poses. A shared multimodal representation jointly predicts K-step actions and a local ΔPose, which is recursively propagated to update the geometric state.

horizons, StreamVLN (Wei et al., 2025) combines fast short-term context with slow-updating compressed memory, while Towards Long-Horizon VLN (Song et al., 2025) introduces multi-granularity dynamic memory. MapNav (Zhang et al., 2025a), ProFocus (Xue et al., 2026), MemVLN (Liu et al., 2026), and JanusVLN (Zeng et al., 2026) further improve memory utilization through semantic maps, selective reasoning, episodic/procedural memory, or decoupled semantic–spatial memory. Complementarily, Dynam3D (Wang et al., 2025c) constructs hierarchical 3D representations from posed RGB-D observations, GA-VLN (Yang et al., 2026a) uses depth and camera poses to build geometry-aware bird's-eye-view representations, and Progress-Think (Wang et al., 2026b) models semantic instruction progress from observation memory. Beyond VLN, Cambrian-P (Yang et al., 2026b) highlights camera pose as a geometric signal for relating observations across viewpoints.

2.2 Predictive and World-Model-Based Navigation

Predictive approaches anticipate future outcomes to support navigation decisions. NavMorph (Yao et al., 2025) models latent environmental dynamics for foresighted planning, while NavQ (Xu et al., 2025) estimates the future prospects of candidate actions. MonoDream (Wang et al., 2026a) predicts current and future panoramic cues from monocular observations, while SpaAct (Li et al., 2026b) learns action-conditioned visual transitions through future-frame selection. WNM-3D (Huang et al., 2026) further couples future-view prediction with navigation actions under geometry-aware scene representations. LatentPilot (Hao et al., 2026) incorporates latent visual reasoning to anticipate future scene information for navigation. These methods support navigation through future prediction and structured scene representations. PoseVLN focuses on explicitly propagating a camera-pose state: trajectory-level poses associate past and current observations within a shared coordinate frame, while local ΔPose specifies the state transition over the predicted action horizon.

3 Method

Framework Overview. We propose **PoseVLN**, a pose-aware VLN framework built upon StreamVLN under the single-RGB streaming setting. As illustrated in Figure 2, pose-aware memory associates observations with camera poses in a shared episode-relative coordinate frame, where poses are encoded into embeddings and inserted alongside the corresponding visual tokens. Conditioned on these inputs and the instruction, the VLM produces a shared representation for jointly predicting the next K navigation actions and a local ΔPose over the same horizon. The predicted transition is composed with the current trajectory-level pose, and the resulting future pose is paired with the newly acquired observation and written back into memory. At inference, this process proceeds recursively from $P_0 = I_4$, requiring neither step-wise ground-truth poses nor odometry.

3.1 PRELIMINARY

Navigation Task. Given an instruction I, the agent navigates from egocentric RGB observations $\{V_t\}_{t=0}^{T}$. At step t, it observes V_t and maintains pose-aware memory

$$\mathcal{M}_t = \{(V_{h_i}, P_{h_i})\}_{i=1}^{N_h}, \tag{1}$$

where P_{h_i} denotes the trajectory-level pose associated with memory observation V_{h_i}. The action space is $\mathcal{A} = \{\texttt{Forward}, \texttt{Left}, \texttt{Right}, \texttt{Stop}\}$, and the model predicts a K-step action chunk

$$A_{t:t+K-1} = (a_t, \ldots, a_{t+K-1}), \qquad a_{t+k} \in \mathcal{A}, \tag{2}$$

with $K = 4$ in our experiments.

Camera Pose Tokens. Let $P_i \in SE(3)$ denote the camera-to-episode transformation of V_i, with $P_0 = I_4$. For tokenization, we encode its inverse $E_i^{\mathrm{w2c}} = P_i^{-1}$:

$$g_i = [\mathbf{t}_i, \mathbf{q}_i, \bar{f}^h, \bar{f}^w] \in \mathbb{R}^9, \tag{3}$$

where $\mathbf{t}_i$ and $\mathbf{q}_i$ are the extrinsic translation and unit WXYZ rotation quaternion of E_i^{w2c}. The FoV entries are fixed sensor constants. A pose encoder maps g_i to $z_i^{\mathrm{pose}} = \mathrm{PoseEncoder}(g_i)$, which is concatenated with the corresponding visual tokens. Coordinate and parameterization details appear in Appendices A.1 and A.3.

3.2 POSEVLN

Joint Action and Pose Prediction. The instruction I, pose-aware memory $\mathcal{M}_t$, and current observation–pose pair (V_t, P_t) are jointly processed by the VLM to predict navigation actions and a local pose transition over the same K-step horizon:

$$(I, \mathcal{M}_t, (V_t, P_t)) \longrightarrow \left(A_{t:t+K-1}, \Delta\hat{P}_{t\to t+K}\right). \tag{4}$$

The shared representation generates the K action tokens, while the current pose-token hidden state passes through an MLP and a VGGT CameraHead (Wang et al., 2025a) for iterative local-pose regression. The predicted transition is then decoded and composed with the current pose to update the geometric state.

Pose-aware Geometric State Update. All memory and current poses share the same episode-relative frame. Using column-vector coordinates, the local transition is $\Delta P_{t\to t+K} = P_t^{-1} P_{t+K}$. The CameraHead parameterizes this transition through its inverse w2c encoding, $\Delta E_{t\to t+K}^{\mathrm{w2c}} = (\Delta P_{t\to t+K})^{-1}$. During decoding, raw quaternion outputs are normalized and converted to rotation matrices. The future trajectory-level pose is then obtained by

$$\hat{P}_{t+K} = P_t \oplus \Delta\hat{P}_{t\to t+K} = P_t \left(\Delta\hat{E}_{t\to t+K}^{\mathrm{w2c}}\right)^{-1}, \tag{5}$$

where $\oplus$ denotes rigid-body pose composition. After the action chunk is executed and V_{t+K} becomes available, the pair $(V_{t+K}, \hat{P}_{t+K})$ is written back into memory for subsequent predictions. At inference, this update is applied recursively from $P_0 = I_4$, with P_t given by the previously predicted state. Local-transform construction and recursive propagation are detailed in Appendices A.2 and A.4.

3.3 TRAINING OBJECTIVE

PoseVLN is trained with action prediction, pose regression, and directional progress objectives.

Action Loss. The predicted action chunk is supervised with an autoregressive cross-entropy loss:

$$\mathcal{L}_{\mathrm{act}} = -\frac{1}{K}\sum_{k=0}^{K-1} \log p_\theta\left(a_{t+k}^* \mid a_{t:t+k-1}^*, I, \mathcal{M}_t, V_t, P_t\right). \tag{6}$$

Pose Loss. Let $(\mathbf{t}^{\Delta}, \mathbf{q}^{\Delta})$ encode the ground-truth inverse local transform $\Delta E^{\mathrm{w2c},*}_{t\to t+K}$. GT rotations are deterministically converted to canonical WXYZ quaternions with $\|\mathbf{q}^{\Delta}\|_2 = 1$ and $q^{\Delta}_w \geq 0$. Following the component-wise regression of Cambrian-P (Yang et al., 2026b) and VGGT (Wang et al., 2025a), we optimize

$$\mathcal{L}_{\mathrm{pose}} = \sum_{s=1}^{S} w_s \left(\|\hat{\mathbf{t}}^{(s)} - \mathbf{t}^{\Delta}\|_1 + \|\hat{\mathbf{q}}^{(s)} - \mathbf{q}^{\Delta}\|_1 \right), \tag{7}$$

where w_s assigns larger weights to later refinement stages. The quaternion term regresses the canonical target from the raw output; it is not a geodesic or sign-invariant loss. Predicted quaternions are normalized only during decoding, not when computing the regression loss.

Directional Progress Loss. To discourage insufficient predicted translation along the target direction, we complement the pose regression loss with a directional progress constraint. Specifically, we constrain the predicted translation's projection onto the target direction in the inverse-extrinsic encoding. Let $d = \|\mathbf{t}^{\Delta}\|_2$; for $d > \tau$, we define

$$\mathcal{L}_{\mathrm{prog}} = \max\left(0,\ \rho d - \hat{\mathbf{t}}^{\top} \frac{\mathbf{t}^{\Delta}}{d}\right), \qquad d > \tau, \tag{8}$$

where $\hat{\mathbf{t}}$ is the final-stage translation, ρ specifies the required projection ratio, and τ filters negligible translations. The loss is zero for $d \leq \tau$. Both translation terms represent inverse-extrinsic components, rather than camera-center displacements. The complete formulation and hyperparameters appear in Appendix C.

The overall objective is

$$\mathcal{L} = \mathcal{L}_{\mathrm{act}} + \lambda_{\mathrm{pose}}\mathcal{L}_{\mathrm{pose}} + \lambda_{\mathrm{prog}}\mathcal{L}_{\mathrm{prog}}. \tag{9}$$

Here, λ_{pose} and λ_{prog} weight pose regression and directional progress supervision, respectively.

Training and Inference. Initialized from StreamVLN (Wei et al., 2025), PoseVLN undergoes supervised cold-start training followed by iterative closed-loop DAgger refinement (Wei et al., 2025; Hao et al., 2026), without introducing additional datasets. Following LatentPilot's (Hao et al., 2026) training pipeline, PoseVLN further incorporates pose-aware inputs and geometric supervision. During rollout collection, predicted pose states are recursively propagated, allowing pose errors to affect subsequent decisions and visited states. Simulator camera poses are recorded for pose context and local ΔPose supervision, and are not used as regular step-wise policy inputs except for re-anchoring during DAgger expert recovery. At inference, only the first observation defines the episode-relative origin ($P_0 = I$); all subsequent poses are recursively predicted without ground-truth poses or external odometry. Further details on pose propagation and implementation are provided in Appendices A.4 and E, respectively.

4 Experiments

4.1 Experimental Setup

Simulation Environments and Metrics. Following recent continuous Vision-Language Navigation methods (Krantz et al., 2020; Wei et al., 2025; Cheng et al., 2024; Zeng et al., 2026), we evaluate PoseVLN on two standard VLN-CE benchmarks, R2R-CE (Anderson et al., 2018) and RxR-CE English Val-Unseen (Ku et al., 2020). Both benchmarks are built upon Matterport3D environments and evaluated in the Habitat simulator (Savva et al., 2019). We report results on the Val-Unseen splits. For navigation evaluation, we report Navigation Error (NE), Oracle Success Rate (OS), Success Rate (SR), Success weighted by Path Length (SPL), and normalized Dynamic Time Warping (nDTW). For pose evaluation, we report translational and rotational Relative Pose Error (RPE) over different temporal horizons. All baseline results are taken from their respective papers, with observation settings indicated in the comparison tables.

Implementation Details. PoseVLN is initialized from StreamVLN (Wei et al., 2025), whose navigation backbone is built on LLaVA-Video 7B with Qwen2-7B as the language model. We follow

Table 1: Comparison with state-of-the-art methods on the R2R-CE Val-Unseen split. Pano., Odo., Depth, and S.RGB denote panoramic view, odometry, depth, and single RGB observations, respectively. All baseline results are reported from their respective papers.

Method	Observation				DAgger	R2R-CE Val-Unseen			
	Pano.	Odo.	Depth	S.RGB		NE↓	OS↑	SR↑	SPL↑
HPN+DN [ICCV21] (Krantz et al., 2021)	✓	✓	✓			6.31	40.0	36.0	34.0
CMA [CVPR22] (Hong et al., 2022)	✓	✓	✓			6.20	52.0	41.0	36.0
Sim2Sim [ECCV22] (Krantz & Lee, 2022)	✓	✓	✓			6.07	52.0	43.0	36.0
VLN↻BERT [CVPR22] (Hong et al., 2022)	✓	✓	✓			5.74	53.0	44.0	39.0
Ego^2-Map [ICCV23] (Hong et al., 2023)	✓	✓	✓			5.54	56.0	47.0	41.0
DreamWalker [ICCV23] (Wang et al., 2023a)	✓	✓	✓			5.53	59.0	49.0	44.0
GridMM [ICCV23] (Wang et al., 2023b)	✓	✓	✓			5.11	61.0	49.0	41.0
Reborn [CVPR22] (An et al., 2022)	✓	✓	✓			5.40	57.0	50.0	46.0
ScaleVLN [ICCV23] (Wang et al., 2023c)	✓	✓	✓			4.80	–	55.0	51.0
ETPNav [TPAMI24] (An et al., 2023)	✓	✓	✓		✓	4.71	65.0	57.0	49.0
Seq2Seq [ECCV20] (Krantz et al., 2020)			✓	✓	✓	7.77	37.0	25.0	22.0
R2R-CMTP [CVPR21] (Chen et al., 2021)	✓	✓	✓			7.90	38.0	26.4	22.7
InstructNav [CoRL24] (Long et al., 2024)	✓	✓	✓			6.89	–	31.0	24.0
CM2 [CVPR22] (Georgakis et al., 2022)		✓	✓	✓		7.02	41.5	34.3	27.6
LAW [EMNLP21] (Raychaudhuri et al., 2021)		✓	✓	✓	✓	6.83	44.0	35.0	31.0
WS-MGMap [NIPS22] (Chen et al., 2022)		✓	✓	✓		6.28	47.6	38.9	34.3
MapNav [ACL25] (Zhang et al., 2025a)		✓	✓	✓	✓	4.93	53.0	39.7	37.2
NaVid-4D [ICRA25] (Liu et al., 2025)			✓	✓		5.99	55.7	43.8	37.1
Sim2Real [CoRL24] (Wang et al., 2024)		✓	✓	✓		5.95	55.8	44.9	30.4
COSMO [ICCV25] (Zhang et al., 2025b)	✓					–	56.0	47.0	40.0
AO-Planner [AAAI25] (Chen et al., 2025)	✓		✓			5.55	59.0	47.0	33.0
g3D-LF [CVPR25] (Wang & Lee, 2025)		✓	✓	✓		5.70	59.5	47.2	34.6
NavMorph [ICCV25] (Yao et al., 2025)			✓	✓		5.75	56.9	47.9	33.2
GA-VLN [CVPR26] (Yang et al., 2026a)		✓	✓	✓		4.80	67.6	61.0	55.2
NaVid [RSS24] (Zhang et al., 2024b)				✓	✓	5.47	49.1	37.4	35.9
Uni-NaVid [RSS25] (Zhang et al., 2024a)				✓	✓	5.58	53.3	47.0	42.7
NaVILA [RSS25] (Cheng et al., 2024)				✓		5.22	62.5	54.0	49.0
StreamVLN [ICRA26] (Wei et al., 2025)				✓	✓	4.98	64.2	56.9	51.9
IDEAL-VLN [ICML26] (Xin et al., 2026)				✓		4.62	64.9	59.8	52.9
JanusVLN [ICLR26] (Zeng et al., 2026)				✓	✓	4.78	65.2	60.5	56.8
LatentPilot [ECCV26] (Hao et al., 2026)				✓	✓	4.41	66.3	62.0	58.0
PoseVLN (Ours)				✓	✓	**4.01**	**69.4**	**64.6**	**59.4**

the same single-RGB streaming setting and train on trajectories from R2R (Anderson et al., 2018), RxR (Ku et al., 2020), and ScaleVLN (Wang et al., 2023c), together with closed-loop DAgger rollouts (Wei et al., 2025; Hao et al., 2026), without introducing additional datasets. PoseVLN augments the backbone with pose-aware inputs and geometric supervision. Detailed implementation settings and training hyperparameters are provided in Appendix E.

4.2 Main Results

Results on R2R-CE. As shown in Table 1, we compare PoseVLN with a broad range of VLN methods on the R2R-CE Val-Unseen split under different observation settings. PoseVLN operates solely from a single RGB stream, without panoramic observations, depth, or step-wise odometry.

PoseVLN achieves an NE of 4.01, an OS of 69.4, an SR of 64.6, and an SPL of 59.4, obtaining the best performance among all compared methods across all four metrics. Compared with StreamVLN (Wei et al., 2025), which serves as our navigation backbone, PoseVLN improves SR from 56.9 to 64.6 and SPL from 51.9 to 59.4, corresponding to absolute gains of 7.7 and 7.5 points, while reducing NE from 4.98 to 4.01 and increasing OS from 64.2 to 69.4. These gains indicate that explicitly modeling and propagating geometric state provides complementary spatial information beyond the visual-memory representation of the navigation backbone, improving both goal reaching and path efficiency without introducing additional sensing modalities. Notably, these gains are achieved without using ground-truth pose or odometry during inference, supporting our hypothesis that camera pose can be learned and propagated as an internal geometric state for navigation.

Table 2: Comparison with state-of-the-art methods on the RxR-CE Val-Unseen split.

Method	Observation				DAgger	RxR-CE Val-Unseen			
	Pano.	Odo.	Depth	S.RGB		NE↓	SR↑	SPL↑	nDTW↑
CMA [CVPR22] (Hong et al., 2022)	✓	✓	✓			8.76	26.5	22.1	47.0
VLN⟳BERT [CVPR22] (Hong et al., 2022)	✓	✓	✓			8.98	27.0	22.6	46.7
Reborn [CVPR22] (An et al., 2022)	✓	✓	✓			5.98	48.6	42.0	63.3
ETPNav [TPAMI24] (An et al., 2023)	✓	✓	✓		✓	5.64	54.7	44.8	61.9
LAW [EMNLP21] (Raychaudhuri et al., 2021)		✓	✓	✓	✓	10.90	8.0	8.0	38.0
Seq2Seq [ECCV20] (Krantz et al., 2020)			✓	✓	✓	12.10	13.9	11.9	30.8
Sim2Real [CoRL24] (Wang et al., 2024)		✓	✓	✓		8.79	25.5	18.1	–
NavMorph [ICCV25] (Yao et al., 2025)			✓	✓		8.85	30.8	22.8	44.2
AO-Planner [AAAI25] (Chen et al., 2025)	✓		✓			7.06	43.3	30.5	50.1
GA-VLN [CVPR26] (Yang et al., 2026a)		✓	✓	✓		5.88	55.4	45.2	–
Uni-NaVid [RSS25] (Zhang et al., 2024a)				✓	✓	6.24	48.7	40.9	–
NaVILA [RSS25] (Cheng et al., 2024)				✓		6.77	49.3	44.0	58.8
StreamVLN [ICRA26] (Wei et al., 2025)				✓	✓	6.22	52.9	46.0	61.9
JanusVLN [ICLR26] (Zeng et al., 2026)				✓	✓	6.06	56.2	47.5	62.1
LatentPilot [ECCV26] (Hao et al., 2026)				✓	✓	5.19	58.2	49.9	67.5
IDEAL-VLN [ICML26] (Xin et al., 2026)				✓		5.23	59.1	50.7	64.5
PoseVLN (Ours)				✓	✓	**4.76**	**62.0**	**53.5**	**70.5**

Results on RxR-CE. As shown in Table 2, PoseVLN achieves 62.0% SR, 53.5% SPL, and 70.5 nDTW, with an NE of 4.76 m on RxR-CE Val-Unseen. Compared with StreamVLN, PoseVLN improves SR, SPL, and nDTW by 9.1, 7.5, and 8.6 points, respectively. These improvements indicate higher navigation success, better path efficiency, and closer agreement with the reference trajectories on this challenging benchmark. Nevertheless, the remaining unsuccessful episodes motivate further analysis of accumulated pose errors and stopping behavior over long navigation horizons.

4.3 Ablation and Analysis

We conduct a series of ablation and diagnostic experiments to better understand how pose-aware learning affects navigation performance and geometric state estimation.

Effect of Training Objectives. We perform cumulative ablations during cold-start training, where the cold-start data consist of R2R, RxR, and ScaleVLN. Starting from our reproduced StreamVLN baseline with 56.17% SR, $+\mathcal{L}_{\text{act}}$ denotes action-only replay training, $+\mathcal{L}_{\text{pose}}$ adds pose regression, and $+\mathcal{L}_{\text{prog}}$ further introduces directional progress supervision. As shown in Table 3, action-only replay substantially degrades the pretrained policy. This is expected because StreamVLN is further optimized on additional data after training on R2R, RxR, and ScaleVLN; replaying only the cold-start data therefore shifts the model away from its later optimized state. Adding $\mathcal{L}_{\text{pose}}$ largely recovers this degradation, indicating that pose supervision provides complementary geometric signals beyond action prediction. However, roughly one quarter of the training segments contain pure rotations with little translation, which may partly explain the tendency of ΔPose predictions toward near-zero motion under pose regression alone. We therefore introduce $\mathcal{L}_{\text{prog}}$ to encourage sufficient translation along the target direction, further improving performance.

We further apply DAgger refinement to both the action-only baseline and PoseVLN using the same aggregated closed-loop rollout data. DAgger largely recovers the action-only baseline, whereas PoseVLN achieves a substantially larger improvement. This suggests that closed-loop training alone cannot explain the final gain, and that pose-aware geometric supervision plays an important role.

Pose prediction compared with VGGT. We compare PoseVLN with VGGT (Wang et al., 2025a) over horizons $H \in \{4, 8, 12, 16\}$. PoseVLN recursively composes successive $K = 4$ predictions, incorporating newly observed RGB frames, updated memory, and the previously predicted pose at each prediction step. VGGT instead performs direct two-view inference from the endpoint image pair $[I_i, I_{i+H}]$. Since VGGT's monocular two-view reconstruction is subject to global scale ambiguity, translation errors are evaluated after scale alignment. For each valid prediction–GT pair, both outputs are converted to local $c2w$ displacements. A non-negative translation scale is independently

Table 3: Cumulative training ablation on R2R-CE Val-Unseen.

Method	SR (%)↑	SPL (%)↑	NE (m)↓
StreamVLN	56.17	50.02	5.100
$\mathcal{L}_{\text{act}}$	53.51	48.03	5.250
+ DAgger	56.99	53.38	4.862
$\mathcal{L}_{\text{act}} + \mathcal{L}_{\text{pose}}$	56.33	51.39	4.883
$\mathcal{L}_{\text{act}} + \mathcal{L}_{\text{pose}} + \mathcal{L}_{\text{prog}}$	57.21	52.24	4.705
+ DAgger	**64.63**	**59.35**	**4.007**

Table 4: Pose prediction comparison across horizons. PoseVLN composes predictions, while pretrained VGGT observes endpoint images.

	#Pairs	Position RPE (m)		Rotation RPE (°)	
H	Moving / All	PoseVLN	VGGT	PoseVLN	VGGT
4	20,836 / 27,912	0.071	**0.015**	7.68	**0.70**
8	21,277 / 26,069	0.145	**0.065**	**11.67**	11.98
12	20,253 / 24,228	0.242	**0.215**	**14.73**	23.24
16	18,660 / 22,387	**0.362**	0.453	**17.40**	31.35

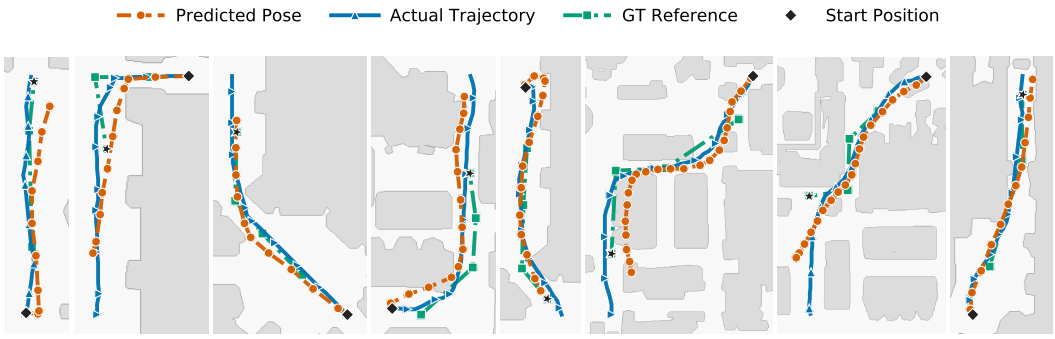

Figure 3: Qualitative comparison between predicted pose trajectories and actual navigation trajectories on R2R-CE Val-Unseen.

estimated as $s^* = \max(0, \widehat{\mathbf{t}}^\top \mathbf{t}^{\text{GT}} / \|\widehat{\mathbf{t}}\|_2^2)$, with the same per-pair alignment procedure applied to both methods. This evaluation measures translation agreement up to scale rather than metric displacement accuracy. Position RPE is evaluated on pairs with GT displacement greater than $0.05\,\text{m}$, while Rotation RPE is computed on all valid pairs. All errors are micro-averaged over the corresponding valid pairs. As shown in Table 4, VGGT achieves lower Position RPE for $H = 4, 8, 12$, whereas PoseVLN performs better at $H = 16$. For Rotation RPE, VGGT achieves lower error at $H = 4$, while PoseVLN is comparable or better at longer horizons.

Comparison between predicted and actual trajectories. Figure 3 qualitatively compares the recursively predicted pose trajectories with the actual navigation trajectories on R2R-CE Val-Unseen. The predicted trajectories generally capture the overall motion direction, major turns, and trajectory shape. In some long-horizon examples, the overall structure remains recognizable despite noticeable endpoint drift, illustrating that coarse trajectory consistency can coexist with accumulated localization errors. These examples suggest that PoseVLN captures meaningful geometric structure over multiple navigation steps, while accurate long-horizon pose propagation remains a challenge.

Pose accuracy in successful and failed navigation episodes. The relationship between geometric state estimation and navigation success is examined by grouping R2R-CE Val-Unseen episodes according to their final outcome (SR $= 1/0$). Here, $H = 4$ denotes the direct $K = 4$ prediction, while $H = 8, 12, 16$ are obtained by autoregressively composing successive $K = 4$ pose transitions. As shown on the left side of Table 5, successful episodes consistently exhibit lower pose errors across all horizons, with larger absolute gaps under longer-horizon composition. At $H = 16$, the mean raw position error reaches $1.660\,\text{m}$ for failed episodes, nearly $3\times$ that of successful episodes ($0.554\,\text{m}$). Similarly, translation-direction errors are lower in successful episodes than in failed ones: $7.67°$ versus $12.54°$ at $H = 4$, and $12.29°$ versus $22.01°$ at $H = 16$. These results indicate an association between navigation success and more accurate geometric state estimation, particularly in translation position and direction over long-horizon propagation.

Effect of pose inputs. We further study the contribution of memory and current pose information through an inference-time input ablation. All variants use the same trained PoseVLN checkpoint,

Table 5: Pose analysis on R2R-CE Val-Unseen. Left: comparison of pose errors between successful and failed navigation episodes. Right: ablation of memory and current pose inputs.

Each entry: Successful / Failed (S/F)			
H	**Raw Pos. (m)** ↓	**Direction (°)** ↓	**Rotation (°)** ↓
4	**0.145** / 0.383	**7.67** / 12.54	**7.43** / 7.96
8	**0.262** / 0.769	**8.78** / 17.72	**11.09** / 12.29
12	**0.396** / 1.193	**10.67** / 19.29	**13.85** / 15.64
16	**0.554** / 1.660	**12.29** / 22.01	**16.25** / 18.55

Method	**Pose Input**		**R2R-CE Val-Unseen**			
	Hist.	**Curr.**	**NE**↓	**OS**↑	**SR**↑	**SPL**↑
PoseVLN	✓	✓	**4.01**	**69.44**	**64.63**	**59.35**
w/o Memory Pose		✓	4.05	68.08	63.32	58.05
w/o Current Pose	✓		4.15	68.08	62.94	57.85
w/o Mem./Curr. Pose			4.11	67.65	62.89	57.73

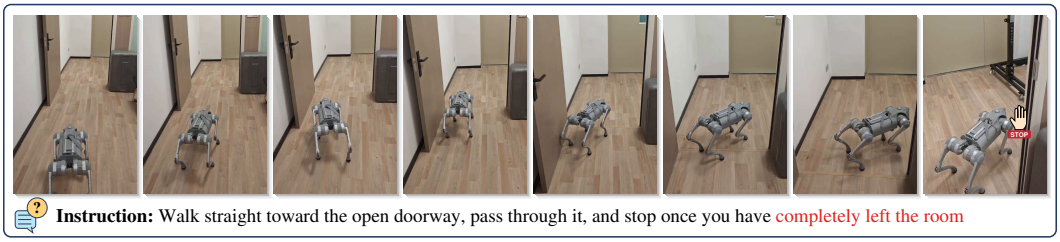

Figure 4: Real-world navigation of PoseVLN on a Unitree Go2.

with the corresponding pose inputs masked only during evaluation. As shown on the right of Table 5, using both memory and current poses achieves the best performance, with 64.63% SR and 59.35% SPL. Removing either pose source consistently degrades navigation performance, while removing both yields the lowest SR and SPL. Masking the current pose leads to a slightly larger degradation than masking memory poses, while the full model performs best across all reported metrics. These results indicate that both current and memory pose states provide useful geometric context under the jointly trained model.

4.4 Real-World Deployment

We further conduct a preliminary real-world deployment of PoseVLN on a Unitree Go2 equipped with an onboard RGB camera, with inference performed on a remote RTX 4090 server. As shown in Figure 4, the robot successfully follows the instruction, traverses the doorway, and stops after leaving the room. This result demonstrates the applicability of PoseVLN to physical robot navigation under real visual observations, while systematic real-world evaluation is left for future work.

5 Limitations

PoseVLN still has several limitations. First, recursive pose composition can accumulate estimation errors and lead to long-horizon drift. Second, the current experiments do not fully isolate the respective contributions of auxiliary geometric supervision and recursively propagated pose inputs. Finally, the complementarity between learned ΔPose and action-derived pose estimates remains to be further investigated. Future work may explore more robust long-horizon pose propagation, better disentanglement of geometric supervision from state representation, and hybrid geometric estimation that combines learned pose transitions with action-derived motion cues.

6 Conclusion

We introduce PoseVLN, a pose-aware VLN framework that connects past memory, current spatial state, and future motion through geometric state evolution. PoseVLN organizes observations in an episode-relative coordinate frame and jointly predicts navigation actions and local ΔPose, whose recursive propagation maintains an evolving spatial state throughout navigation. Using only single-RGB observations without step-wise ground-truth pose or odometry at inference, PoseVLN achieves strong performance on R2R-CE and RxR-CE. These results demonstrate the effectiveness of camera pose as a compact geometric bridge for temporal state reasoning in VLN.

AI USE STATEMENT

In this work, we did not use generative AI tools for tasks requiring mandatory disclosure under the ICLR 2027 AI Policy for Authors. Tasks involving mathematical proofs and qualitative or thematic data analysis are not applicable to this work. Additionally, we used ChatGPT to improve the language and readability of the manuscript. We have reviewed all AI-assisted work. Specifically, the authors checked and revised the suggested edits to ensure that they preserved the intended technical meaning and accurately reflected the methods and experimental results. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we describe the architecture, pose representation, training objectives, and inference procedure of PoseVLN in Section 3. The datasets, implementation details, training settings, and evaluation protocols are provided in Section 4 and Appendices. The ablation studies and pose-error analyses in Section 4 further document the contributions of the main components and the evaluation of pose prediction.

REFERENCES

Dong An, Zun Wang, Yangguang Li, Yi Wang, Yicong Hong, Yan Huang, Liang Wang, and Jing Shao. 1st place solutions for rxr-habitat vision-and-language navigation competition (cvpr 2022). *arXiv preprint arXiv:2206.11610*, 2022.

Dong An, Hanqing Wang, Wenguan Wang, Zun Wang, Yan Huang, Keji He, and Liang Wang. Etpnav: Evolving topological planning for vision-language navigation in continuous environments. *arXiv preprint arXiv:2304.03047*, 2023.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Jiaqi Chen, Bingqian Lin, Xinmin Liu, Lin Ma, Xiaodan Liang, and Kwan-Yee K Wong. Affordances-oriented planning using foundation models for continuous vision-language navigation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 23568–23576, 2025.

Kevin Chen, Junshen K. Chen, Jo Chuang, Marynel Vazquez, and Silvio Savarese. Topological planning with transformers for vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

Peihao Chen, Dongyu Ji, Kunyang Lin, Runhao Zeng, Thomas H. Li, Mingkui Tan, and Chuang Gan. Weakly-supervised multi-granularity map learning for vision-and-language navigation. *arXiv preprint arXiv:2210.07506*, 2022.

An-Chieh Cheng, Yandong Ji, Zhaojing Yang, Zaitian Gongye, Xueyan Zou, Jan Kautz, Erdem Bıyık, Hongxu Yin, Sifei Liu, and Xiaolong Wang. Navila: Legged robot vision-language-action model for navigation. *arXiv preprint arXiv:2412.04453*, 2024.

Guangzhao Dai, Shuo Wang, Zihan Wang, Guo-Sen Xie, Yang Yang, Jinshan Pan, Qianru Sun, and Xiangbo Shu. History to future: Evolving agent with experience and thought for zero-shot vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15177–15187, 2026.

Georgios Georgakis, Karl Schmeckpeper, Karan Wanchoo, Soham Dan, Eleni Miltsakaki, Dan Roth, and Kostas Daniilidis. Cross-modal map learning for vision and language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Haihong Hao, Lei Chen, Mingfei Han, Changlin Li, Dong An, Yuqiang Yang, Zhihui Li, and Xiaojun Chang. Latentpilot: Scene-aware vision-and-language navigation by dreaming ahead with latent visual reasoning. *arXiv preprint arXiv:2603.29165*, 2026.

Yicong Hong, Zun Wang, Qi Wu, and Stephen Gould. Bridging the gap between learning in discrete and continuous environments for vision-and-language navigation. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15418–15428. IEEE, 2022.

Yicong Hong, Yang Zhou, Ruiyi Zhang, Franck Dernoncourt, Trung Bui, Stephen Gould, and Hao Tan. Learning navigational visual representations with semantic map supervision. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3032–3044. IEEE, 2023.

Yuehao Huang, Yunzi Wu, Xiaotao Zhang, Xinhai Li, Jiankun Dong, Jiajun Lv, Chi Zhang, Chenjia Bai, Yong Liu, and Xuelong Li. Wnm-3d: A world navigation model with 3d scene conditioning for closed-loop vln. *arXiv preprint arXiv:2608.07267*, 2026.

Jacob Krantz and Stefan Lee. Sim-2-sim transfer for vision-and-language navigation in continuous environments. In *European conference on computer vision*, pp. 588–603. Springer, 2022.

Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *European Conference on Computer Vision*, pp. 104–120. Springer, 2020.

Jacob Krantz, Aaron Gokaslan, Dhruv Batra, Stefan Lee, and Oleksandr Maksymets. Waypoint models for instruction-guided navigation in continuous environments. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15142–15151. IEEE, 2021.

Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

Kailing Li, Tianwen Qian, Lijin Yang, Yuqian Fu, Jingyu Gong, Xiaoling Wang, and Liang He. Bridging the 2d-3d gap: A hierarchical semantic-geometric map for vision language navigation. *arXiv preprint arXiv:2606.00095*, 2026a.

Pengna Li, Kangyi Wu, Shaoqing Xu, Fang Li, Hanbing Li, Lin Zhao, Kailin Lyu, Long Chen, Zhi-Xin Yang, and Nanning Zheng. Spaact: Spatially-activated transition learning with curriculum adaptation for vision-language navigation. *arXiv preprint arXiv:2604.27620*, 2026b.

Haoran Liu, Weikang Wan, Xiqian Yu, Minghan Li, Jiazhao Zhang, Bo Zhao, Zhibo Chen, Zhongyuan Wang, Zhizheng Zhang, and He Wang. Na vid-4d: Unleashing spatial intelligence in egocentric rgb-d videos for vision-and-language navigation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 10607–10615. IEEE, 2025.

Yuqi Liu, Shengju Qian, Tianyuan Qu, Mingxian Lin, Zixuan Wang, Xin Wang, Bei Yu, and Jiaya Jia. Memvln: Episodic and procedural memory for vision-and-language navigation. *arXiv preprint arXiv:2607.23504*, 2026.

Yuxing Long, Wenzhe Cai, Hongcheng Wang, Guanqi Zhan, and Hao Dong. Instructnav: Zero-shot system for generic instruction navigation in unexplored environment. *arXiv preprint arXiv:2406.04882*, 2024.

Sonia Raychaudhuri, Saim Wani, Shivansh Patel, Unnat Jain, and Angel Chang. Language-aligned waypoint (law) supervision for vision-and-language navigation in continuous environments. In *Proceedings of the 2021 conference on empirical methods in natural language processing*, pp. 4018–4028, 2021.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

Xinshuai Song, Weixing Chen, Yang Liu, Weikai Chen, Guanbin Li, and Liang Lin. Towards long-horizon vision-language navigation: Platform, benchmark and method. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12078–12088. IEEE, 2025.

Hanqing Wang, Wei Liang, Luc Van Gool, and Wenguan Wang. Dreamwalker: Mental planning for continuous vision-language navigation. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 10839–10849. IEEE, 2023a.

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, 2025a.

Shuo Wang, Yongcai Wang, Wanting Li, Xudong Cai, Yucheng Wang, Maiyue Chen, Kaihui Wang, Zhizhong Su, Deying Li, and Zhaoxin Fan. Aux-Think: Exploring reasoning strategies for data-efficient vision-language navigation. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 30892–30916. Curran Associates, Inc., 2025b. doi: 10.52202/085713-1040.

Shuo Wang, Yongcai Wang, Zhaoxin Fan, Yucheng Wang, Maiyue Chen, Kaihui Wang, Zhizhong Su, Wanting Li, Xudong Cai, Yeying Jin, et al. Monodream: Monocular vision-language navigation with panoramic dreaming. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 10074–10082, 2026a.

Shuo Wang, Yucheng Wang, Guoxin Lian, Yongcai Wang, Maiyue Chen, Kaihui Wang, Bo Zhang, Zhizhong Su, Yutian Zhou, Wanting Li, et al. Progress-think: Semantic progress reasoning for vision-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4076–4086, 2026b.

Zihan Wang and Gim Hee Lee. g3d-lf: Generalizable 3d-language feature fields for embodied tasks. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14191–14202. IEEE, 2025.

Zihan Wang, Xiangyang Li, Jiahao Yang, Yeqi Liu, and Shuqiang Jiang. Gridmm: Grid memory map for vision-and-language navigation. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15579–15590. IEEE, 2023b.

Zihan Wang, Xiangyang Li, Jiahao Yang, Yeqi Liu, and Shuqiang Jiang. Sim-to-real transfer via 3d feature fields for vision-and-language navigation. *arXiv preprint arXiv:2406.09798*, 2024.

Zihan Wang, Seungjun Lee, and Gim Hee Lee. Dynam3D: Dynamic Layered 3D Tokens Empower VLM for Vision-and-Language Navigation. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 153522–153544. Curran Associates, Inc., 2025c. doi: 10.52202/085713-5133. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/e1a1847e39a7b79b41199176b152f0e6-Paper-Conference.pdf.

Zun Wang, Jialu Li, Yicong Hong, Yi Wang, Qi Wu, Mohit Bansal, Stephen Gould, Hao Tan, and Yu Qiao. Scaling data generation in vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023c.

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.

Zihao Xin, Wentong Li, Yixuan Jiang, Bin Wang, Piji Li, Jianke Zhu, Jie Qin, and Sheng-Jun Huang. Instruction decomposition and action alignment for vision-language navigation. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Peiran Xu, Xicheng Gong, and Yadong Mu. Navq: Learning a q-model for foresighted vision-and-language navigation. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 6327–6341. IEEE, 2025.

Wei Xue, Mingcheng Li, Xuecheng Wu, Jingqun Tang, Dingkang Yang, and Lihua Zhang. Profocus: Proactive perception and focused reasoning in vision-and-language navigation. *arXiv preprint arXiv:2603.05530*, 2026.

Jiahao Yang, Zihan Wang, Xiangyang Li, Xing Zhu, Yujun Shen, Yinghao Xu, and Shuqiang Jiang. Ga-vln: Geometry-aware bev representation for efficient vision-language navigation. *arXiv preprint arXiv:2605.22036*, 2026a.

Jihan Yang, Zifan Zhao, Xichen Pan, Shusheng Yang, Junyi Zhang, Bingyi Kang, Hu Xu, Shang-Wen Li, and Saining Xie. Cambrian-p: Pose-grounded video understanding. *arXiv preprint arXiv:2605.22819*, 2026b.

Xuan Yao, Junyu Gao, and Changsheng Xu. Navmorph: A self-evolving world model for vision-and-language navigation in continuous environments. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5536–5546. IEEE, 2025.

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. In *International Conference on Learning Representations*, volume 2026, pp. 33001–33026, 2026.

Jiazhao Zhang, Kunyu Wang, Shaoan Wang, Minghan Li, Haoran Liu, Songlin Wei, Zhongyuan Wang, Zhizheng Zhang, and He Wang. Uni-navid: A video-based vision-language-action model for unifying embodied navigation tasks. *arXiv preprint arXiv:2412.06224*, 2024a.

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. *arXiv preprint arXiv:2402.15852*, 2024b.

Lingfeng Zhang, Xiaoshuai Hao, Qinwen Xu, Qiang Zhang, Xinyao Zhang, Pengwei Wang, Jing Zhang, Zhongyuan Wang, Shanghang Zhang, and Renjing Xu. Mapnav: A novel memory representation via annotated semantic maps for vlm-based vision-and-language navigation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13032–13056, 2025a.

Lingfeng Zhang, Zeying Gong, Xiaoshuai Hao, Haoxiang Fu, Qiang Zhang, Mingliang Zhou, Hangjun Ye, Xiaojun Liang, Junwei Liang, and Wenbo Ding. Futurenav: Unified world-action modeling for vision-and-language navigation. *arXiv preprint arXiv:2606.30367*, 2026.

Siqi Zhang, Yanyuan Qiao, Qunbo Wang, Zike Yan, Qi Wu, Zhihua Wei, and Jing Liu. Cosmo: Combination of selective memorization for low-cost vision-and-language navigation. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5511–5522. IEEE, 2025b.

Appendix

This appendix provides additional details of PoseVLN, including the pose representation and recursive geometric state propagation, closed-loop DAgger collection, the Directional Progress Loss, training sample construction, and implementation details. We further clarify how geometric supervision is used during training. Simulator-recorded camera poses are used to construct pose-aware training context and local ΔPose targets, whereas at inference only the first observation defines the episode-relative origin and all subsequent pose states are recursively propagated from model predictions without ground-truth pose or external odometry.

A Pose Representation and Geometric State Propagation

A.1 Episode-relative Camera Coordinates

Using column-vector coordinates, let the simulator camera-to-world transformation of frame t be

$$T_t^{\mathrm{c2w}} = \begin{bmatrix} R_t & p_t \\ 0^\top & 1 \end{bmatrix} \in SE(3), \tag{10}$$

where R_t and p_t denote the camera orientation and center in the simulator global coordinate system.

We use the first camera frame as the episode reference:

$$\bar{T}_t^{\mathrm{c2w}} = \left(T_0^{\mathrm{c2w}}\right)^{-1} T_t^{\mathrm{c2w}}. \tag{11}$$

By construction, $\bar{T}_0^{\mathrm{c2w}} = I_4$. This rigid coordinate change preserves metric scale; no additional trajectory-scale normalization is applied. The main-text trajectory-level state is

$$P_t \equiv \bar{T}_t^{\mathrm{c2w}}, \tag{12}$$

which maps camera coordinates to the episode frame.

For pose tokenization, we use the inverse world-to-camera representation:

$$E_t^{\mathrm{w2c}} = P_t^{-1} = \left(\bar{T}_t^{\mathrm{c2w}}\right)^{-1}. \tag{13}$$

Thus, the trajectory state is stored in c2w form, while its pose token encodes the corresponding w2c rotation and translation.

A.2 Local ΔPose Supervision

For a current timestep i and a future timestep $j = i + K$, the local camera transformation is

$$\Delta T_{i\to j}^{\mathrm{c2w}} = \left(\bar{T}_i^{\mathrm{c2w}}\right)^{-1} \bar{T}_j^{\mathrm{c2w}}. \tag{14}$$

Writing

$$\bar{T}_i^{\mathrm{c2w}} = \begin{bmatrix} \bar{R}_i & \bar{p}_i \\ 0^\top & 1 \end{bmatrix}, \qquad \bar{T}_j^{\mathrm{c2w}} = \begin{bmatrix} \bar{R}_j & \bar{p}_j \\ 0^\top & 1 \end{bmatrix}, \tag{15}$$

gives

$$R_{ij}^{\mathrm{rel}} = \bar{R}_i^\top \bar{R}_j, \qquad p_{ij}^{\mathrm{rel}} = \bar{R}_i^\top (\bar{p}_j - \bar{p}_i). \tag{16}$$

Here, p_{ij}^{rel} is the displacement of the future camera center expressed in the current camera frame. The main-text transition is $\Delta P_{i\to j} = \Delta T_{i\to j}^{\mathrm{c2w}}$.

Substituting Eq. 11 into Eq. 14 yields

$$\Delta T_{i\to j}^{\mathrm{c2w}} = \left(T_i^{\mathrm{c2w}}\right)^{-1} T_j^{\mathrm{c2w}}, \tag{17}$$

showing that the local transformation is independent of the arbitrary global origin and orientation.

We supervise the CameraHead using the inverse local transform:

$$\Delta E_{i\to j}^{\mathrm{w2c}} = \left(\Delta T_{i\to j}^{\mathrm{c2w}}\right)^{-1} = E_j^{\mathrm{w2c}} \left(E_i^{\mathrm{w2c}}\right)^{-1}. \tag{18}$$

This transform maps current-camera coordinates to future-camera coordinates. Its rotation and translation satisfy

$$R_{ij}^{\mathrm{w2c}} = (R_{ij}^{\mathrm{rel}})^\top, \qquad \mathbf{t}_{ij}^{\Delta} = -(R_{ij}^{\mathrm{rel}})^\top p_{ij}^{\mathrm{rel}}. \tag{19}$$

Accordingly, the translation supervised by the pose and Directional Progress objectives is an inverse-extrinsic component, rather than the camera-center displacement p_{ij}^{rel}.

A.3 9D CAMERA PARAMETERIZATION

We reuse the VGGT CameraHead architecture and follow the component-wise pose regression of Cambrian-P and VGGT (Yang et al., 2026b; Wang et al., 2025a). Our encoding explicitly uses WXYZ quaternion ordering:

$$g_i = [\mathbf{t}_i, \mathbf{q}_i, \bar{f}^h, \bar{f}^w] = [t_x, t_y, t_z, q_w, q_x, q_y, q_z, \bar{f}^h, \bar{f}^w] \in \mathbb{R}^9. \tag{20}$$

For trajectory-level tokens, $\mathbf{t}_i$ and $\mathbf{q}_i$ encode the translation and rotation of $E_i^{\mathrm{w2c}} = P_i^{-1}$. Local targets use the same encoding layout for $\Delta E_{i\to j}^{\mathrm{w2c}}$.

Canonical Quaternion Regression. For a GT rotation R^*, we deterministically construct the target

$$\mathbf{q}^* = \mathrm{Canon}\left(\mathrm{Quat}_{\mathrm{WXYZ}}(R^*)\right), \qquad \|\mathbf{q}^*\|_2 = 1, \quad q_w^* \geq 0, \tag{21}$$

where canonicalization standardizes the quaternion sign. The CameraHead outputs an unconstrained vector $\hat{\mathbf{q}} \in \mathbb{R}^4$, and the rotation term directly minimizes $\|\hat{\mathbf{q}} - \mathbf{q}^*\|_1$ at each refinement stage. This is component-wise regression against a canonical target, rather than a geodesic or sign-invariant rotation loss. Predicted quaternions are normalized during geometric decoding, not before computing this regression loss.

Fixed Field of View. The two FoV entries are retained for compatibility with the 9D CameraHead interface. They are overwritten with known sensor constants during training and inference, contribute no effective FoV loss, and are excluded from geometric state propagation.

A.4 RECURSIVE POSE PROPAGATION

At inference, PoseVLN initializes the episode coordinate system with $\hat{P}_0 = P_0 = I_4$, without querying a GT camera pose. For the final-stage quaternion output, geometric decoding applies

$$\bar{\mathbf{q}} = \frac{\hat{\mathbf{q}}}{\max(\|\hat{\mathbf{q}}\|_2, \epsilon)}, \qquad \hat{R} = \mathrm{QuatToMat}_{\mathrm{WXYZ}}(\bar{\mathbf{q}}). \tag{22}$$

The decoded rotation and predicted translation form $\Delta\hat{E}_{i\to j}^{\mathrm{w2c}}$. We invert this transform before state composition:

$$\Delta\hat{T}_{i\to j}^{\mathrm{c2w}} = \left(\Delta\hat{E}_{i\to j}^{\mathrm{w2c}}\right)^{-1}. \tag{23}$$

For decoded w2c components $(\hat{R}, \hat{\mathbf{t}})$, the inverse has rotation $\hat{R}^\top$ and translation $-\hat{R}^\top\hat{\mathbf{t}}$. The trajectory-level state is updated by

$$\hat{P}_j = \hat{P}_i \Delta\hat{T}_{i\to j}^{\mathrm{c2w}}. \tag{24}$$

Equivalently, under the w2c convention,

$$\hat{E}_j^{\mathrm{w2c}} = \Delta\hat{E}_{i\to j}^{\mathrm{w2c}} \hat{E}_i^{\mathrm{w2c}}. \tag{25}$$

The propagated state $\hat{P}_j$ is paired with the new observation V_j and stored in memory:

$$\mathcal{M}_t = \{(V_{h_1}, \hat{P}_{h_1}), \ldots, (V_{h_{N_h}}, \hat{P}_{h_{N_h}})\}. \tag{26}$$

Its pose token is constructed from $\hat{E}_j^{\mathrm{w2c}} = \hat{P}_j^{-1}$. Thus, all non-initial current and memory states are model-predicted.

The resulting state evolution is

$$\hat{P}_0 \xrightarrow{\Delta\hat{P}_{0\to K}} \hat{P}_K \xrightarrow{\Delta\hat{P}_{K\to 2K}} \hat{P}_{2K} \to \cdots. \tag{27}$$

Pose errors therefore accumulate under the same recursive propagation used for navigation evaluation.

B CLOSED-LOOP DAGGER AND POSE COLLECTION

We collect additional training trajectories through closed-loop policy rollouts in the Habitat simulator. During collection, the policy executes predicted action chunks while its trajectory-level pose state is recursively propagated from predicted local ΔPose according to Appendix A.4. Consequently, accumulated pose errors can affect subsequent navigation decisions during data collection.

For each simulator state used to produce an RGB observation, we record the corresponding ground-truth camera pose T_t^{c2w}. Local pose targets are subsequently constructed from these temporally aligned camera poses following Appendix A.2. Each executed action is logged together with the RGB observation and camera pose at the simulator state from which that action is issued, ensuring temporal alignment among observations, actions, and pose supervision.

Expert Intervention. During a rollout, we compute the minimum distance d_t between the current agent position and the reference path. The Habitat shortest-path expert takes control when $d_t > 1.0\,\mathrm{m}$ and returns control to the policy when $d_t < 0.5\,\mathrm{m}$. Expert takeover is also triggered when the normalized action-error score exceeds 0.5, the accumulated action errors exceed 8, or the policy produces an invalid `Stop` action.

Pose Re-anchoring. When expert correction is triggered, the internally propagated pose state is re-anchored to the current simulator camera pose:

$$\hat{P}_t \leftarrow \left(T_0^{\mathrm{c2w}}\right)^{-1} T_t^{\mathrm{c2w}}. \tag{28}$$

Before control is returned to the policy, unexecuted action predictions and stale recurrent states are discarded, and closed-loop action and pose prediction resumes from the corrected state.

This re-anchoring is used only during training-data collection and is never applied during evaluation or inference. A corrected rollout therefore contains a policy-generated segment, an expert recovery segment, and a subsequent policy-generated segment. The simulator camera trajectory is recorded independently throughout the rollout and is later used to construct ground-truth pose context and ΔPose supervision for training.

C DIRECTIONAL PROGRESS LOSS

The basic pose objective supervises translation and rotation independently. We additionally introduce a directional constraint to discourage local translation predictions with insufficient progress along the ground-truth translation direction.

For the m-th valid local-motion target, let $\mathbf{t}_m^{\Delta}$ denote the ground-truth local w2c translation and $d_m = \|\mathbf{t}_m^{\Delta}\|_2$ its magnitude. For samples satisfying $d_m > \tau$, the normalized local translation direction is

$$\mathbf{u}_m^{\Delta} = \frac{\mathbf{t}_m^{\Delta}}{d_m}. \tag{29}$$

Let $\hat{\mathbf{t}}_m$ denote the translation predicted by the final CameraHead refinement. Its signed progress along the ground-truth direction is

$$s_m^{\mathrm{prog}} = \hat{\mathbf{t}}_m^{\top} \mathbf{u}_m^{\Delta}. \tag{30}$$

The complete Directional Progress Loss is

$$\mathcal{L}_{\mathrm{prog}} = \frac{\sum_m \mathbb{I}(d_m > \tau) \max\left(0,\, \rho d_m - s_m^{\mathrm{prog}}\right)}{\sum_m \mathbb{I}(d_m > \tau) + \epsilon}. \tag{31}$$

We use $\tau = 0.1\,\mathrm{m}$ and $\rho = 0.8$.

This auxiliary objective penalizes predictions with insufficient progress along the ground-truth local translation direction, including near-zero or oppositely directed predictions. Importantly, $\mathcal{L}_{\mathrm{prog}}$ is a local geometric constraint on the pose branch and should not be interpreted as task-level progress toward the navigation goal.

D TRAINING SAMPLE CONSTRUCTION

We construct training samples from trajectories in R2R, RxR, ScaleVLN, and the additional closed-loop rollouts described in Appendix B. For each trajectory, we sample an action window of length $N = 16$ and predict an action chunk every $K = 4$ steps, resulting in $M = N/K = 4$ prediction segments per complete window.

For a window starting at timestep s, the m-th current state is

$$i_m = s + mK, \qquad m \in \{0, \dots, M-1\}, \tag{32}$$

with future target

$$j_m = \min(i_m + K, T_{\text{end}}). \tag{33}$$

Each valid segment provides

$$V_{i_m} \longrightarrow (A_{i_m:j_m-1}, \Delta P_{i_m \to j_m}). \tag{34}$$

The local transition is encoded as its inverse world-to-camera transform for CameraHead supervision.

The pose annotations associated with the current observation and historical memory observations are expressed in the episode-relative coordinate system and encoded into pose latents. For non-initial windows, we additionally sample at most $N_h = 8$ historical observations:

$$\mathcal{M}_{i_m} = \{(V_{h_n}, P_{h_n})\}_{n=1}^{N_h}. \tag{35}$$

These historical observation-pose pairs provide navigation context for the current prediction segment but do not receive independent action or local-pose supervision.

The closed-loop rollouts expose the policy to observation and action sequences induced by its own predictions. The synchronized camera trajectory provides geometrically aligned pose annotations and local ΔPose targets for constructing the corresponding training samples.

E IMPLEMENTATION DETAILS

PoseVLN is built upon StreamVLN under the single-RGB streaming setting. The navigation backbone is initialized from the corresponding StreamVLN checkpoint, while the pose encoder and VGGT CameraHead are initialized from scratch.

We use trajectory windows of length $N = 16$, at most $N_h = 8$ memory observations, and an action-prediction horizon of $K = 4$. The pose encoder maps the 9D camera representation to the 3,584-dimensional language-model embedding space. The VGGT CameraHead operates on a 2,048-dimensional representation with four transformer blocks, 16 attention heads, and four iterative refinements.

Training is performed on four GPUs using AdamW with a learning rate of 5×10^{-6}, zero weight decay, a cosine schedule with 3% warmup, BF16 precision, gradient checkpointing, and DeepSpeed ZeRO-2. The per-device batch size is 1 with 32 gradient-accumulation steps, giving an effective batch size of 128.

The action and pose loss weights are both 1.0. Within the pose objective, translation and quaternion regression each use unit weight. The inherited FoV weight is 0.5, but its loss is zero because the predicted FoV entries are overwritten with the target constants. For the Directional Progress Loss, we use $\tau = 0.1\,\text{m}$ and $\rho = 0.8$. Its coefficient is linearly increased from 0.05 to 0.5 during the first $1{,}920$ supervised-training steps and fixed to 0.2 during subsequent DAgger refinement.

Unless otherwise specified, the remaining settings follow StreamVLN (Wei et al., 2025).