DATA-CENTRIC RECURSIVE SELF-IMPROVEMENT FOR NAVIGATION VIA POLICY-SYNCHRONIZED LANGUAGE-ACTION DUAL BOOTSTRAPPING

Anonymous authors
Paper under double-blind review

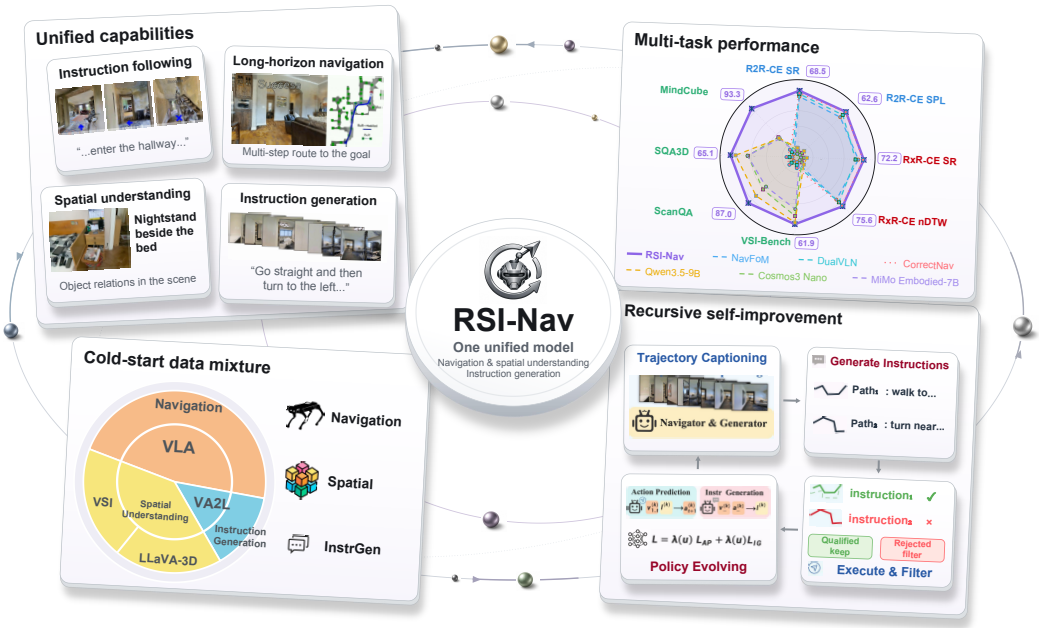


Figure 1: **RSI-Nav** unifies navigation, instruction generation, and spatial understanding in one model, and recursively improves through policy-synchronized language-action dual bootstrapping.

ABSTRACT

Vision-language navigation requires agents to follow natural-language instructions to reach target locations, making high-quality language supervision essential. Since human annotation is costly, prior work has explored scalable data synthesis methods. However, existing methods typically either use separate models for navigation and instruction generation or couple the two tasks through task-specific components; most also annotate the full trajectory corpus with a fixed model before updating, which can cause generated supervision to lag behind the evolving policy. Therefore, we introduce **RSI-Nav**, a data-centric recursive self-improvement framework for navigation via policy-synchronized language-action dual bootstrapping, which unifies vision-language-to-action instruction following and vision-action-to-language instruction generation within a shared model. Specifically, after supervised fine-tuning establishes fundamental capabilities, RSI-Nav progressively processes unlabeled trajectories in small shards. The current model generates instructions for these trajectories, executes them to evaluate goal completion and path efficiency, retains qualified samples, and immediately updates itself before processing the next shard. This shard-wise update keeps newly generated supervision aligned with the evolving policy. We also incorporate spatial understanding data, since stronger spatial reasoning benefits navigation. Extensive experiments show that RSI-Nav reaches 68.5% SR on R2R-CE and 72.2% SR on RxR-CE, while providing stronger instruction-generation utility and retaining strong spatial understanding after recursive evolution.

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.

1 INTRODUCTION

Vision-language navigation (VLN) requires an embodied agent to ground natural-language instructions in visual observations and execute sequential actions to reach a target. This task begins with R2R (Anderson et al., 2018) and has expanded to longer-range navigation in RxR (Ku et al., 2020), goal-oriented settings such as REVERIE (Qi et al., 2020) and SOON (Zhu et al., 2021), and continuous navigation in VLN-CE (Krantz et al., 2020). These increasingly challenging settings place greater demands on language grounding, spatial reasoning, and long-horizon decision making. Accordingly, research has progressed from improving navigation architectures and learning strategies (Chen et al., 2021; 2022a; Wang et al., 2023b; An et al., 2024) to scaling instruction-trajectory supervision through automatic data synthesis (Chen et al., 2022b; Wang et al., 2023c; 2025b).

Scalable data synthesis allows navigation agents to learn from large amounts of unlabeled trajectories that would otherwise lack language supervision. Early methods use speaker models and back-translation to generate instructions for additional trajectories (Fried et al., 2018; Tan et al., 2019), while later work substantially scales this paradigm with larger environments and stronger instruction generators (Chen et al., 2022b; Wang et al., 2023c; 2025b;a). Despite this progress, current pipelines still leave substantial room for improvement. Most methods learn instruction generation and navigation separately, while existing dual-learning approaches typically rely on task-specific architectures (Wang et al., 2022; 2023a), limiting the benefit of bidirectional language-action learning. In addition, synthetic instructions are commonly generated by a fixed model, and the navigation policy is updated only after a large set of trajectories has been annotated. As the policy improves, the generated supervision can therefore become increasingly mismatched with the model being trained. These limitations call for a learning process that jointly exploits the two directions of language-action grounding while keeping newly generated supervision synchronized with the current policy.

This connection is particularly natural in navigation. More generally, complementary capabilities can benefit each other when one provides feedback for the other. An unlabeled trajectory can first be described as a route instruction and then executed again from its starting point, forming two complementary directions of language-action grounding. Instruction generation maps visual-action experience to language, while navigation maps language and visual observations back to actions. A faithful instruction should capture the navigation task well enough to guide the agent to the intended goal efficiently, making navigation execution a direct way to evaluate the generated supervision. When both capabilities are handled by the same evolving model, execution feedback can filter and improve language supervision, which in turn provides better data for subsequent navigation learning.

Based on this observation, we introduce **RSI-Nav**, a data-centric recursive self-improvement framework for navigation via policy-synchronized language-action dual bootstrapping. RSI-Nav uses a shared model to unify instruction following and generation for dual bootstrapping. We first apply supervised fine-tuning to establish basic capabilities for navigation and trajectory annotation. The model then progressively processes unlabeled trajectories in small shards. For each shard, the model generates instructions and executes them. The resulting rollouts assess whether generated instructions can guide the current model to the intended goal efficiently, and only qualified instruction-trajectory pairs are retained. The model is then immediately updated before the next shard. As a result, the updated model generates and evaluates the rollouts used in subsequent training, rather than relying on data produced by a fixed annotator at the beginning of the process. This shard-wise update keeps the generated supervision closely aligned with the evolving policy and forms a recursive process in which better models produce better training data for subsequent improvement.

Beyond language-action dual learning, we also incorporate spatial understanding into the same model, as spatial reasoning is crucial for interpreting routes and making navigation decisions (Ma et al., 2022; Cheng et al., 2024; Yang et al., 2025). Joint training with spatial-understanding data strengthens this capability and in turn benefits navigation. We evaluate RSI-Nav on continuous vision-language navigation, trajectory-to-instruction generation, and spatial understanding. On R2R-CE (Krantz et al., 2020) and RxR-CE (Ku et al., 2020), RSI-Nav consistently improves navigation performance over its supervised initialization. The evolved model substantially improves navigation while retaining strong spatial understanding, and its generated instructions provide stronger training utility for subsequent navigation. We further analyze the self-improvement process, examining policy synchronization, dual learning, action-conditioned instruction generation, and spatial

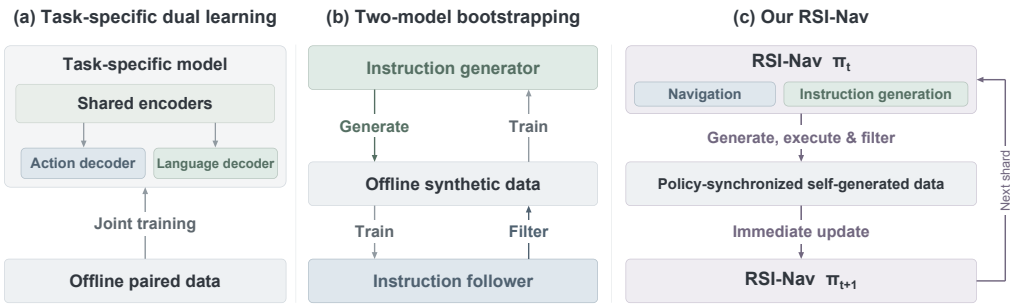


Figure 2: **Comparison of three VLN training paradigms.** (a) Dual-task learning relies on task-specific architectures and costly human annotation. (b) Data flywheel refines data at scale with separate navigation and instruction-generation models. (c) RSI-Nav generates and filters data with the current policy and updates immediately after each shard, forming a policy-synchronized language-action dual bootstrapping loop for data-centric recursive self-improvement.

supervision. These analyses explain why the proposed process improves both generated supervision and the navigation policy. Generally, our contributions can be summarized as follows:

- We present **RSI-Nav**, a data-centric recursive self-improvement framework that unifies instruction following, instruction generation, and spatial understanding within a single model, where dual language-action learning provides complementary supervision to improve navigation.
- We obtain large-scale execution-verified instructions paired with source trajectories as well as a strong unified model. To our knowledge, RSI-Nav is the first navigation framework to combine unified language-action dual learning with policy-synchronized recursive self-improvement.
- We demonstrate strong performance across navigation and spatial-understanding tasks, achieving the best results among compared methods on VLN-CE benchmarks. Analyses examine key components of the self-improvement pipeline to explain how and why RSI-Nav improves.

2 Related Work

Vision-and-Language Navigation. Vision-and-Language Navigation (VLN) requires agents to follow natural-language instructions and navigate to target locations in visual environments. Existing scenarios include navigation with detailed action instructions (Anderson et al., 2018; Ku et al., 2020; Thomason et al., 2020), remote object navigation with coarse-grained instructions (Zhu et al., 2021; Qi et al., 2020), and navigation in continuous environments (Kolve et al., 2017; Krantz et al., 2020; Hong et al., 2022). Substantial progress has been made through vision-language pretraining and history modeling (Hao et al., 2020; Chen et al., 2021; Guhur et al., 2021), explicit map construction and planning (Chen et al., 2022a; Wang et al., 2023b; An et al., 2024), and modern vision-language models that directly predict navigation actions (Zhou et al., 2024; Zheng et al., 2024; Zhang et al., 2024). However, the high cost of human annotation remains a major bottleneck, motivating automatic data augmentation and synthesis. Early methods use speaker models and back-translation (Fried et al., 2018; Tan et al., 2019), while later work exploits large-scale environments and synthetic instructions to construct millions of instruction-trajectory pairs (Chen et al., 2022b; Wang et al., 2023c; 2025b; Li et al., 2025). Recent studies further highlight spatial understanding, including situated 3D reasoning and spatial memory (Yang et al., 2025; Yin et al., 2025).

Complementary Learning and Mutual Bootstrapping. Learning complementary capabilities together provides supervision and supports joint improvement across multiple abilities. In machine translation, dual learning couples forward and backward translation to provide richer mutual feedback (He et al., 2016). Similar ideas also appear in mathematical reasoning, where a language model acts as both solver and verifier (Weng et al., 2023; Lightman et al., 2023), with recent work jointly training answer generation and self-verification in a single model (Zhang et al., 2025a). These methods broadly share the principle that complementary capabilities can provide supervision or feedback for one another. In VLN, instruction following and generation form a similar bidirectional language-action relationship. Prior work realizes this through separate speaker and follower models, where

generated instructions mainly serve as augmentation or scoring signals, or through task-specific architectures and consistency objectives (Fried et al., 2018; Wang et al., 2022; 2023a).

Recursive Self-Improvement. Recursive self-improvement (RSI) broadly studies how an AI system can contribute to improving its subsequent versions. In the data-centric setting, RSIBench-Data (Meng et al., 2026) frames this as turning model failures and checkpoint feedback into better training data and, in turn, better models. Related approaches generate, filter, or verify model outputs for subsequent training in language reasoning (Zelikman et al., 2022; Dong et al., 2023; Xiong et al., 2025; Yuan et al., 2024) and embodied agents (Bousmalis et al., 2023; Lin et al., 2025). In VLN, SRDF (Wang et al., 2025b) iteratively refines synthetic instructions, CorrectNav (Yu et al., 2026) mines failure trajectories for self-correction, and SID-VLN (Li et al., 2025) reuses successful exploration trajectories. However, these methods separate data generation and model updates into relatively coarse rounds, allowing supervision to lag behind the evolving policy.

3 METHODOLOGY

RSI-Nav learns from unlabeled navigation trajectories through policy-synchronized language-action dual bootstrapping. As illustrated in Fig. 3, the framework begins with supervised initialization and recursively improves both the model and its generated supervision. We first introduce the continuous VLN setting, followed by a shared model that unifies instruction following, instruction generation, and spatial understanding in Sec. 3.1. Sec. 3.2 presents the core data-centric recursive self-improvement process, where the current policy generates instructions, executes them to assess goal completion and path efficiency, retains qualified samples, and updates itself before processing the next trajectory shard. Finally, Sec. 3.3 describes how we curate supervision for initialization and recursive evolution, including spatial-data replay and newly generated instruction-trajectory pairs.

Preliminary. We consider vision-language navigation in continuous environments, where an agent follows a natural-language instruction through low-level interaction with the environment. At step t, the agent receives an egocentric RGB observation o_t and selects an action $a_t \in \mathcal{A} = \{\texttt{FORWARD}, \texttt{LEFT}, \texttt{RIGHT}, \texttt{STOP}\}$. A navigation trajectory is represented as

$$\tau_i = (o_1, a_1, o_2, a_2, \ldots, o_T, a_T), \tag{1}$$

where T denotes the trajectory length. Each trajectory starts from an initial state s_i^{start} and terminates at a target location defined by its final state. Given a natural-language instruction x_i, the agent is required to reach the corresponding target efficiently. We use the term *policy-synchronized* rather than strict on-policy learning. The source trajectories remain offline, while instruction generation and execution-based evaluation are repeatedly refreshed by the current model, reducing the gap between generated supervision and the evolving policy, as detailed in Sec. 3.2.

3.1 A UNIFIED VISION-LANGUAGE-ACTION MODEL

RSI-Nav is built on Qwen3.5-9B and uses a single model for instruction following, instruction generation, and spatial understanding. All capabilities share the same model parameters and are distinguished by task-specific prompts, without separate generators, verifiers, or output heads. We formulate the two complementary language-action tasks and present spatial understanding below.

Instruction Following. At navigation step t, the agent maintains an observation history $H_t = \{o_1, \ldots, o_t\}$ and uniformly samples eight frames as visual context $\widetilde{H}_t$. Conditioned on instruction x, $\widetilde{H}_t$, and the task prompt, the model predicts a four-action chunk $c_t = (a_t, a_{t+1}, a_{t+2}, a_{t+3})$. We optimize the standard cross-entropy loss over the predicted action chunk:

$$\mathcal{L}_{\text{IF}} = -\log p_\theta(c_t \mid x, \widetilde{H}_t) = -\sum_{j=0}^{3} \log p_\theta(a_{t+j} \mid x, \widetilde{H}_t, a_{t:t+j-1}). \tag{2}$$

Instruction Generation. For the reverse direction, the model uniformly samples 32 frames $\widetilde{V}_{\tau_i}$ from the trajectory and uses the full action sequence $A_{\tau_i} = (a_1, \ldots, a_T)$ as input. Given these

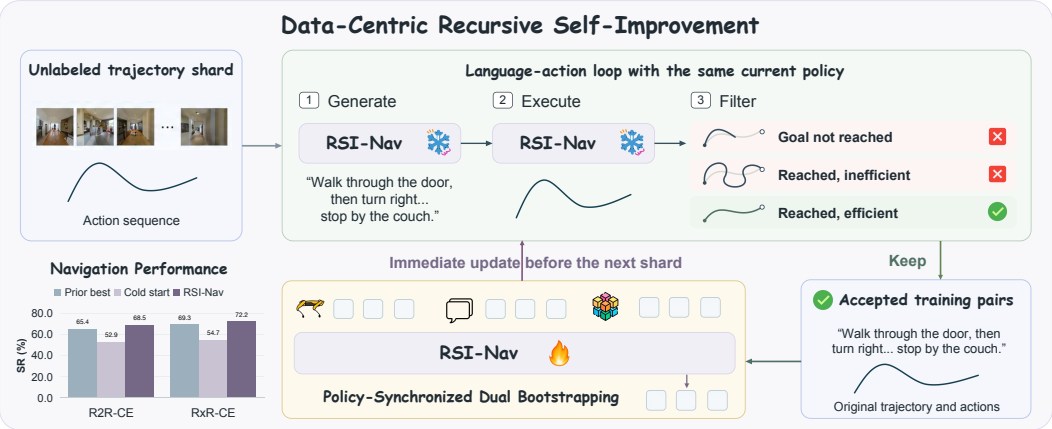


Figure 3: **Overview of RSI-Nav.** The current policy generates instructions for unlabeled trajectories and executes them to evaluate goal completion and efficiency. Qualified instructions are paired with the original trajectories to train instruction following and instruction generation jointly. The model is then updated with the accepted samples and spatial replay data before processing the next shard.

inputs and the task prompt, the model generates $x = (w_1, \ldots, w_M)$ with the objective:

$$\mathcal{L}_{\mathrm{IG}} = -\log p_\theta(x \mid \widetilde{V}_{\tau_i}, A_{\tau_i}) = -\sum_{m=1}^{M} \log p_\theta(w_m \mid \widetilde{V}_{\tau_i}, A_{\tau_i}, w_{<m}). \tag{3}$$

Spatial Understanding. We further incorporate spatial-understanding training data, which provide an important scaffold for embodied decision making and navigation (Gong et al., 2026; Team et al., 2026). These samples are trained under the same token-level objective, denoted as $\mathcal{L}_{\mathrm{spa}}$.

3.2 Policy-Synchronized Recursive Self-Improvement

With the shared instruction-following and instruction-generation model defined above, RSI-Nav recursively improves on a large corpus of unlabeled continuous trajectories. We partition the corpus into small, non-overlapping trajectory shards $\mathcal{U} = \{\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_K\}$ and process each trajectory once. After each shard, the model is immediately updated before generating new supervision for the next one. This fine-grained update keeps data generation and execution-based evaluation close to the current policy, rather than relying on a fixed model over the entire corpus.

Generate, Execute, and Filter. For each trajectory $\tau_i \in \mathcal{B}_k$, the model generates an instruction $\hat{x}_i$ from the visual-action trajectory and then follows $\hat{x}_i$ from the original start state. The endpoint of τ_i serves as the navigation target. This execution tests whether the generated instruction can guide the current model to the intended destination efficiently. We retain a sample based on goal completion and path efficiency. An execution is considered successful if its final position is within 3 m of the target. We further use SPL to measure navigation efficiency and define the acceptance indicator as

$$q_i = \mathbb{I}\left[d_i < 3\ \mathrm{m}\ \wedge\ \mathrm{SPL}_i \geq \tau_{\mathrm{SPL}}\right], \tag{4}$$

where d_i is the geodesic distance from the final position to the target and τ_{SPL} is a fixed efficiency threshold. Only samples satisfying both criteria are retained for subsequent training.

Dual Bootstrapping and Shard-Wise Update. The filtering stage determines which generated instructions are reliable enough for training. For each accepted pair $(\tau_i, \hat{x}_i)$, we retain the original trajectory τ_i. Since $\hat{x}_i$ is generated from τ_i, successful and efficient execution toward the same goal provides an execution-grounded check that the instruction is executable and consistent with the navigation task represented by the source trajectory. We therefore retain the original trajectory as the supervision target rather than the executed trajectory, avoiding the accumulation of execution errors and the cost of repeatedly re-rendering continuous trajectories in Habitat. Keeping the trajectory fixed also focuses recursive improvement on the generated language supervision.

Each accepted pair provides supervision for both complementary directions. For instruction following, $\hat{x}_i$ is paired with the observation histories and original actions along τ_i; for instruction generation, $(\widetilde{V}_{\tau_i}, A_{\tau_i})$ is paired with $\hat{x}_i$. For shard $\mathcal{B}_k$, these accepted samples form $\mathcal{D}_k^{\mathrm{IF}}$ and $\mathcal{D}_k^{\mathrm{IG}}$, respectively. During the shard-wise update, we mix them with spatial-understanding data replayed from supervised initialization. The overall training objective is

$$\mathcal{L}_k = \mathcal{L}_{\mathrm{IF}}\left(\mathcal{D}_k^{\mathrm{IF}}\right) + \mathcal{L}_{\mathrm{IG}}\left(\mathcal{D}_k^{\mathrm{IG}}\right) + \mathcal{L}_{\mathrm{spa}}\left(\mathcal{R}_k^{\mathrm{spa}}\right). \tag{5}$$

Here, $\mathcal{L}_{\mathrm{IF}}$ and $\mathcal{L}_{\mathrm{IG}}$ follow the objectives defined in Sec. 3.1, while $\mathcal{L}_{\mathrm{spa}}$ denotes the corresponding loss on spatial-understanding samples. This decomposition only distinguishes the three supervision sources; in practice, all samples are mixed and jointly optimized with a single SFT objective. The resulting θ_k is immediately used to process the next trajectory shard.

Recursive Self-Improvement. The updated model θ_k immediately processes the next shard $\mathcal{B}_{k+1}$. Refreshing instruction generation and execution-based filtering ensures that new supervision is produced by the latest model rather than a fixed policy. Repeating this cycle allows the model and its generated supervision to co-evolve across shards, while spatial-data replay preserves the reasoning acquired during SFT initialization. The full evolution yields a stronger unified model and a large collection of execution-verified instruction-trajectory pairs, with details in Sec. 4.

3.3 Training Data and Supervision

RSI-Nav is initialized with multi-source supervision for instruction following, instruction generation, and spatial understanding. We use continuous navigation data from SRDF (Wang et al., 2025b), EnvDrop (Tan et al., 2019), ScaleVLN (Wang et al., 2023c), R2R (Anderson et al., 2018), and RxR (Ku et al., 2020), and construct corresponding instruction-generation data by reversing the trajectory-instruction pairs. Since SRDF trajectories are originally discrete, we convert them into continuous trajectories for training, with details in Appendix D. We also incorporate spatial-understanding data from VSI-590K (Yang et al., 2026), 546K samples from LLaVA-3D (Zhu et al., 2024), and MindCube (Yin et al., 2025). All data are mixed for standard supervised fine-tuning, with the same spatial data replayed during recursive evolution. Starting from unlabeled continuous trajectories, RSI-Nav generates about 2.5M instruction-following samples and 1.1M instruction-generation samples through policy-synchronized recursive improvement.

4 Experiments

Implementation Details. RSI-Nav is built on Qwen3.5-9B and trained in two stages on 64 GPUs. First, we perform supervised fine-tuning for one epoch on the multi-source data described in Sec. 3.3, using a learning rate of 5×10^{-6}, a global batch size of 128, and AdamW optimization. We then conduct recursive self-improvement on the unlabeled trajectory corpus, partitioned into non-overlapping shards of 16,384 trajectories. After each shard, qualified instruction-trajectory pairs are immediately used for model updating together with spatial-understanding replay before processing the next shard.

Benchmarks and Evaluation Metrics. We evaluate navigation on R2R-CE (Krantz et al., 2020) and RxR-CE (Ku et al., 2020) using their validation-unseen splits. Following prior work, we report Navigation Error (NE), Oracle Success Rate (OSR), Success Rate (SR), and Success weighted by Path Length (SPL) on R2R-CE, and additionally nDTW on RxR-CE for trajectory alignment. For trajectory-to-instruction generation, we report BLEU-1, BLEU-4, METEOR, ROUGE, CIDEr, and SPICE. Since prior methods are mainly evaluated in discrete Matterport3D environments, whereas RSI-Nav operates in continuous VLN-CE with Habitat observations, these NLP scores serve as reference rather than strictly controlled comparisons. We further evaluate spatial understanding on VSI-Bench (Yang et al., 2025), ScanQA (Azuma et al., 2022), SQA3D (Ma et al., 2022), and MindCube (Yin et al., 2025), reporting VSI Overall, ScanQA CIDEr, SQA3D EM1-R, and MindCube Accuracy, respectively. In the main comparison, we distinguish single-view RGB, panoramic RGB, depth, and odometry inputs, and separately mark methods using waypoint predictors.

Real-World Evaluation Setup. For real-world evaluation, we deploy RSI-Nav on a DEEP Robotics M20 robot. RSI-Nav runs on a remote server equipped with an NVIDIA GeForce RTX

Table 1: **Comparison on R2R-CE and RxR-CE Val-Unseen and four spatial-understanding benchmarks.** * denotes methods using the waypoint predictor from Hong et al. (2022).

Method	Observation				Spatial Understanding				R2R-CE Val-Unseen				RxR-CE Val-Unseen			
	S.RGB	Pano.	Depth	Odo.	VSI-Bench↑	ScanQA↑	SQA3D↑	MindCube↑	NE↓	OSR↑	SR↑	SPL↑	NE↓	SR↑	SPL↑	nDTW↑
Waypoint-Predictor-Based VLN Models																
VLN⟳BERT* (Hong et al., 2022)		✓	✓	✓	-	-	-	-	5.74	53.0	44.0	39.0	8.98	27.0	22.6	46.7
Sim2Sim* (Krantz & Lee, 2022)		✓	✓	✓	-	-	-	-	6.07	52.0	43.0	36.0	-	-	-	-
GridMM* (Wang et al., 2023b)		✓	✓	✓	-	-	-	-	5.11	61.0	49.0	41.0	-	-	-	-
Reborn* (An et al., 2022)		✓	✓	✓	-	-	-	-	5.40	57.0	50.0	46.0	5.98	48.6	42.0	63.3
ETPNav* (An et al., 2024)		✓	✓	✓	-	-	-	-	4.71	65.0	57.0	49.0	5.64	54.7	44.8	61.9
HNR* (Wang et al., 2024)		✓	✓	✓	-	-	-	-	4.42	67.0	61.0	51.0	5.50	56.3	46.7	63.5
BEVBert* (An et al., 2023)		✓	✓	✓	-	-	-	-	4.57	67.0	59.0	50.0	-	-	-	-
ScaleVLN* (Wang et al., 2023c)		✓	✓	✓	-	-	-	-	4.80	-	55.0	51.0	-	-	-	-
RGB-Based Continuous VLN Models																
InstructNav (Long et al., 2024)		✓	✓	✓	-	-	-	-	6.89	-	31.0	24.0	-	-	-	-
LAW (Raychaudhuri et al., 2021)	✓		✓	✓	-	-	-	-	6.83	44.0	35.0	31.0	10.90	8.0	8.0	38.0
Dynam3D (Wang et al., 2026)	✓		✓	✓	-	-	-	-	5.34	62.1	52.9	45.7	-	-	-	-
AO-Planner (Chen et al., 2024)		✓	✓		-	-	-	-	5.55	59.0	47.0	33.0	7.06	43.3	30.5	50.1
Seq2Seq (Krantz et al., 2020)	✓		✓		-	-	-	-	7.77	37.0	25.0	22.0	12.10	13.9	11.9	30.8
CMA (Krantz et al., 2020)	✓		✓		-	-	-	-	7.37	40.0	32.0	30.0	-	-	-	-
Uni-NaVid (Zhang et al., 2025b)	✓				-	-	-	-	5.58	53.3	47.0	42.7	6.24	48.7	40.9	-
NaVILA (Cheng et al., 2025)	✓				-	-	-	-	5.22	62.5	54.0	49.0	6.77	49.3	44.0	58.8
StreamVLN (Wei et al., 2025)	✓				-	-	-	-	4.98	64.2	56.9	51.9	6.22	52.9	46.0	61.9
CorrectNav (Yu et al., 2026)	✓				-	-	-	-	4.24	67.5	65.1	62.3	4.09	69.3	63.3	75.2
NavFoM (Zhang et al., 2026)	✓				-	-	-	-	4.61	72.1	61.7	55.3	4.74	64.4	56.2	65.8
JanusVLN (Zeng et al., 2026)	✓				-	-	-	-	4.78	65.2	60.5	56.8	6.06	56.2	47.5	62.1
DualVLN (Wei et al., 2026)	✓				-	-	-	-	4.05	70.7	64.3	58.5	4.58	61.4	51.8	70.0
AwareVLN (Guo et al., 2026)	✓				-	-	-	-	4.02	73.5	65.4	55.1	3.95	67.6	56.1	65.7
Spatial Understanding Models																
RoboBrain2.5-8B (Tan et al., 2026)	✓				41.0	63.1	28.1	28.1	-	-	-	-	-	-	-	-
MiMo Embodied-7B (Hao et al., 2025)	✓				48.5	55.5	44.6	32.3	-	-	-	-	-	-	-	-
Qwen3.5-9B (Qwen Team, 2026)	✓				61.0	73.1	55.3	33.1	-	-	-	-	-	-	-	-
Embodied R1.5 (Yuan et al., 2026)	✓				56.1	67.1	48.2	34.8	-	-	-	-	-	-	-	-
Cosmos3 Nano (Agarwal et al., 2026)	✓				54.9	60.9	44.2	34.8	-	-	-	-	-	-	-	-
RSI-Nav Cold Start	✓				**64.6**	**93.2**	**68.1**	91.8	5.54	66.4	52.9	45.5	5.39	54.7	48.8	62.4
RSI-Nav (Ours)	✓				61.9	87.0	65.1	**93.3**	**3.93**	**75.6**	**68.5**	**62.6**	**3.85**	**72.2**	**63.5**	**75.6**

Table 2: **Ablation study of SFT data composition.** We examine the effects of dual learning, action conditioning, and spatial supervision.

Nav	V2L	VA2L	Spatial	NE↓	OSR↑	SR↑	SPL↑
✓				6.19	57.2	44.7	38.6
✓	✓			5.78	58.1	47.0	38.8
✓		✓		5.69	59.9	48.8	41.0
✓		✓	✓	**5.54**	**66.4**	**52.9**	**45.5**

Table 3: **Ablation on policy update granularity.** *Candidate* denotes the number of trajectories processed per GPU before one policy update.

Cold Start	NE↓	OSR↑	SR↑	SPL↑
Baseline	5.54	66.4	52.9	45.5
64	5.57	62.9	51.9	45.9
256	**4.91**	**68.7**	**57.2**	**48.7**
1024	5.03	68.1	55.1	46.3

4090 GPU and continuously processes egocentric RGB observations and language instructions to generate navigation actions for closed-loop execution. We evaluate the agent in both indoor and outdoor environments with tasks covering obstacle avoidance, narrow-space navigation, instruction following, open-vocabulary landmark grounding, and long-horizon navigation.

4.1 Main Results

Comparison with Prior Methods. Tab. 1 compares RSI-Nav with previous methods on R2R-CE, RxR-CE, and four spatial-understanding benchmarks. On navigation, RSI-Nav achieves the best results among the compared methods across all metrics, reaching 68.5% SR and 62.6% SPL on R2R-CE, and 72.2% SR, 63.5% SPL, and 75.6 nDTW on RxR-CE. Compared with the supervised initialization, recursive evolution consistently improves navigation success and trajectory alignment, showing that self-generated and execution-verified instructions provide supervision beyond the initial SFT stage. The gain in nDTW indicates that the model better follows the instructed route rather than merely reaching the goal through alternative paths, which is particularly important for long-horizon instruction following. For spatial understanding, both the supervised initialization and evolved model outperform the original Qwen3.5-9B and recent spatial models across all four benchmarks. After recursive evolution, RSI-Nav retains strong spatial understanding, with some decline on three benchmarks but an improvement on MindCube from 91.8 to 93.3. This shows that spatial replay largely preserves spatial capabilities while achieving substantial navigation gains.

Real-World Qualitative Results. Fig. 4 presents qualitative results of RSI-Nav in real-world environments. RSI-Nav successfully handles diverse navigation scenarios, including obstacle avoid-

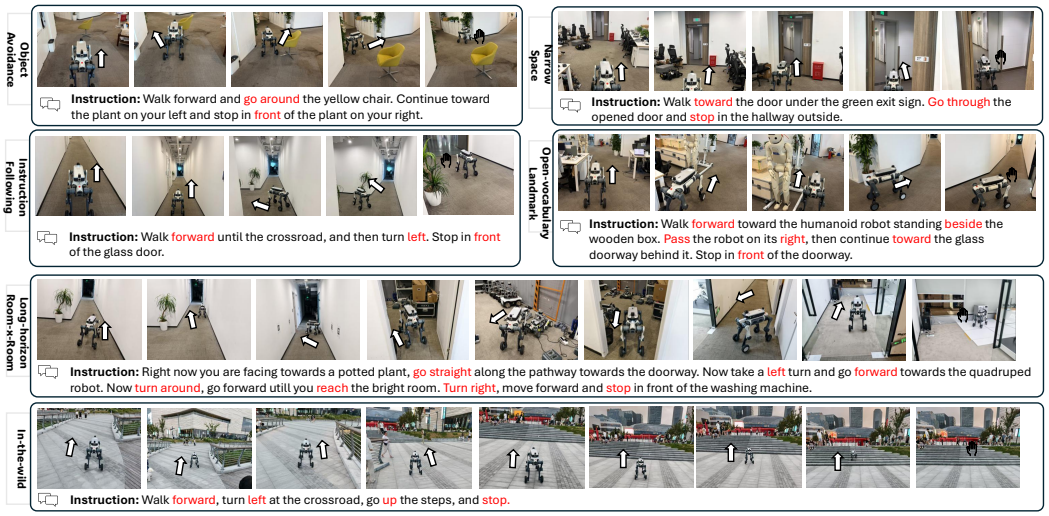


Figure 4: Qualitative results of RSI-Nav in real-world environments.

ance, narrow-space traversal, multi-step instruction following, open-vocabulary landmark grounding, long-horizon room-x-room navigation, and in-the-wild outdoor navigation. These examples demonstrate successful closed-loop execution across several real-world scenarios.

4.2 Ablation Studies

How Does Policy Synchronization Affect Learning? We study the update granularity on a fixed subset of 64K unlabeled trajectories using 64 GPUs, where *candidate* denotes the number of trajectories processed per GPU before each policy update. As shown in Tab. 3, 256 candidates perform best, improving SR and SPL most significantly. With only 64 candidates, performance also drops, suggesting that each update changes the policy too little for successive policies and their generated data to differ meaningfully. In contrast, 1,024 candidates process the entire subset before updating, approaching the conventional offline setting and enlarging the policy-data gap. The intermediate granularity therefore provides a better balance between meaningful policy evolution and timely supervision refresh. Based on this ablation, we use 256 candidates per GPU, corresponding to a global shard size of 16,384 trajectories on 64 GPUs, for full-scale recursive evolution.

Does Dual Learning Benefit Navigation? We examine whether instruction generation provides useful supervision for instruction following. As shown in Tab. 2, adding VA2L data improves SR from 44.7% to 48.8% and SPL from 38.6% to 41.0% over navigation-only training, confirming that trajectory annotation benefits navigation. Tab. 5 further shows that joint instruction following and generation substantially improves navigation while causing minor changes in instruction-generation metrics. This indicates that the two tasks can be jointly optimized with little loss in annotation quality, while the complementary trajectory-to-language supervision provides clear gains for navigation.

Are Actions Important for Dual Learning? Under the same visual input, VA2L consistently outperforms V2L, improving SR from 47.0% to 48.8% and SPL from 38.8% to 41.0%. Although instruction generation can rely on visual observations alone, incorporating actions establishes a stronger dual relation between action-to-language generation and language-to-action navigation. This indicates that action information helps more effectively transfer instruction-generation supervision back to navigation; we further examine this effect from the generation side next.

How Does Joint Training Affect Instruction Generation? For RSI-Nav, we use the jointly trained VL2A-VA2L generator evaluated in Tab. 5. We examine the reverse benefit of dual learning from instruction following to instruction generation. Prior methods are often evaluated in discrete Matterport3D settings with panoramic observations, whereas RSI-Nav operates in continuous VLN-CE using low-resolution Habitat observations; their NLP scores thus serve only as reference rather than strictly comparable baselines. NLP metrics also mainly measure similarity to human-written

Table 4: **Trajectory-to-instruction generation and training utility.** We report some NLP metrics for instruction generation and evaluate training utility by training navigation models on the generated supervision and measuring downstream R2R-CE navigation performance.

Method	Instruction Generation						Training Utility			
	BLEU-1↑	BLEU-4↑	METEOR↑	ROUGE↑	CIDEr↑	SPICE↑	NE↓	OSR↑	SR↑	SPL↑
BT-speaker (Fried et al., 2018)	68.2	26.3	21.7	48.0	37.9	18.9	-	-	-	-
EnvDrop (Tan et al., 2019)	72.3	27.1	23.6	49.0	41.7	21.8	-	-	-	-
CCC (Wang et al., 2022)	70.8	27.2	23.1	47.7	46.1	21.4	-	-	-	-
KEFA (Zeng et al., 2023)	73.8	28.3	24.4	50.3	42.7	23.4	-	-	-	-
LANA (Wang et al., 2023a)	73.6	28.9	23.7	49.8	45.7	22.6	-	-	-	-
C-Instructor (Kong et al., 2024)	71.3	26.6	23.9	47.3	44.7	21.2	-	-	-	-
BEVInstructor (Fan et al., 2024)	69.9	26.4	23.0	46.7	44.9	20.8	-	-	-	-
SRDF (Wang et al., 2025b)	75.3	31.1	25.0	51.4	49.2	26.2	5.67	55.2	36.8	30.4
RSI-Nav	71.0	28.2	23.2	48.5	**50.6**	20.1	**5.58**	**58.7**	**41.3**	**35.1**

Table 5: **Ablation of bidirectional language–action learning.** Joint VL2A–VA2L training substantially improves navigation while maintaining comparable instruction-generation quality.

Training Objective		Instruction Following (VL2A)				Instruction Generation (VA2L)					
VL2A	VA2L	NE↓	OSR↑	SR↑	SPL↑	BLEU-1↑	BLEU-4↑	METEOR↑	ROUGE↑	CIDEr↑	SPICE↑
✓		6.19	57.2	44.7	38.6	-	-	-	-	-	-
	✓	-	-	-	-	71.7	28.8	23.3	48.8	49.8	20.6
✓	✓	**5.69**	**59.9**	**48.8**	**41.0**	71.0	28.2	23.2	48.5	**50.6**	20.1

instructions and may not reflect downstream training utility (Gadre et al., 2024). We therefore annotate the same trajectories with different generators and train identical navigation models on the resulting data. As shown in Tab. 4, although SRDF scores higher on most NLP metrics, RSI-Nav yields substantially better downstream navigation, improving SR from 36.8% to 41.3% and SPL from 30.4% to 35.1%. Together with the bidirectional-training results in Tab. 5, these results show that joint VL2A-VA2L training benefits navigation while preserving instruction-generation quality, and that the resulting RSI-Nav generator yields stronger downstream utility than prior work.

Does Spatial Understanding Benefit Navigation? Adding spatial-understanding supervision on top of VA2L further improves all navigation metrics, reducing navigation error while increasing success rate and efficiency. The gains support the role of spatial relations and viewpoint reasoning in navigation, highlighting spatial understanding as complementary supervision.

Limitations. RSI-Nav still relies on simulated environments and an offline trajectory pool, while execution-based filtering incurs considerable computational cost. Although we provide qualitative real-world demonstrations, systematic robustness across diverse real-world environments, unseen domains, and substantially longer-horizon navigation remains to be established.

5 Conclusion

We introduce **RSI-Nav**, a data-centric recursive self-improvement framework for navigation via policy-synchronized language-action dual bootstrapping. RSI-Nav unifies instruction following, instruction generation, and spatial understanding in one shared policy, and turns unlabeled trajectories into supervision through instruction generation, execution-based filtering, and shard-wise updates. By refreshing supervision with the latest model, RSI-Nav maintains a policy-synchronized learning process while jointly training instruction following and generation. Experiments on R2R-CE and RxR-CE demonstrate substantial navigation improvements, while trajectory-to-instruction and spatial-understanding evaluations show strong complementary capabilities. Ablations examine the effects of update granularity, dual-task training, action conditioning, and spatial supervision. Overall, RSI-Nav shows that large-scale unlabeled navigation trajectories can support model improvement through generated and filtered supervision, providing a scalable direction for self-improving embodied navigation. We discuss limitations and future work in Appendix H.

AI USE DISCLOSURE

We used generative AI tools only to assist with language polishing and editing of the manuscript, and with software programming. All AI-assisted text was reviewed and revised by the authors, and all AI-assisted code was inspected and tested for correctness. The authors take full responsibility for the final content of this work.

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.

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. In *CVPRW*, 2022.

Dong An, Yuankai Qi, Yangguang Li, Yan Huang, Liang Wang, Tieniu Tan, and Jing Shao. Bevbert: Multimodal map pre-training for language-guided navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2737–2748, 2023.

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. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

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 Conference on Computer Vision and Pattern Recognition*, pp. 3674–3683, 2018.

Daichi Azuma, Taiki Miyanishi, Shuhei Kurita, and Motoaki Kawanabe. Scanqa: 3d question answering for spatial scene understanding. In *CVPR*, 2022.

Konstantinos Bousmalis, Giulia Vezzani, Dushyant Rao, Coline Devin, Alex X Lee, Maria Bauza, Todor Davchev, Yuxiang Zhou, Agrim Gupta, Akhil Raju, et al. Robocat: A self-improving foundation agent for robotic manipulation. *arXiv preprint arXiv:2306.11706*, 2023.

Jiaqi Chen, Bingqian Lin, Xinmin Liu, Xiaodan Liang, and Kwan-Yee K Wong. Affordances-oriented planning using foundation models for continuous vision-language navigation. *arXiv preprint*, 2024.

Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. History aware multimodal transformer for vision-and-language navigation. *Advances in Neural Information Processing Systems*, 34:5834–5847, 2021.

Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. Think global, act local: Dual-scale graph transformer for vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16537–16547, 2022a.

Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. Learning from unlabeled 3d environments for vision-and-language navigation. In *European Conference on Computer Vision*, pp. 638–655. Springer, 2022b.

An-Chieh Cheng, Hongxu Yin, Yang Fu, Qiushan Guo, Ruihan Yang, Jan Kautz, Xiaolong Wang, and Sifei Liu. Spatialrgpt: Grounded spatial reasoning in vision-language models. In *NeurIPS*, 2024.

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. *Robotics: Science and Systems*, 2025.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. Raft: Reward ranked finetuning for generative foundation model alignment. *arXiv preprint arXiv:2304.06767*, 2023.

Sheng Fan, Rui Liu, Wenguan Wang, and Yi Yang. Navigation instruction generation with bev perception and large language models. In *European Conference on Computer Vision*, pp. 368–387. Springer, 2024.

Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. Speaker-follower models for vision-and-language navigation. In *Advances in Neural Information Processing Systems*, pp. 3314–3325, 2018.

Samir Yitzhak Gadre, Gabriel Ilharco, Alex Fang, Jonathan Hayase, Georgios Smyrnis, Thao Nguyen, Ryan Marten, Mitchell Wortsman, Dhruba Ghosh, Jieyu Zhang, et al. Datacomp: In search of the next generation of multimodal datasets. *Advances in Neural Information Processing Systems*, 36, 2024.

Ziyang Gong, Zehang Luo, Anke Tang, Zhe Liu, Shi Fu, Zhi Hou, Ganlin Yang, Weiyun Wang, Xiaofeng Wang, Jianbo Liu, Gen Luo, Haolan Kang, Shuang Luo, Yue Zhou, Yong Luo, Li Shen, Xiaosong Jia, Yao Mu, Xue Yang, Chunxiao Liu, Junchi Yan, Hengshuang Zhao, Dacheng Tao, and Xiaogang Wang. Ace-brain-0: Spatial intelligence as a shared scaffold for universal embodiments, 2026.

Pierre-Louis Guhur, Makarand Tapaswi, Shizhe Chen, Ivan Laptev, and Cordelia Schmid. Airbert: In-domain pretraining for vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1634–1643, 2021.

Wenxuan Guo, Xiuwei Xu, Yichen Liu, Xiangyu Li, Hang Yin, Huangxing Chen, Wenzhao Zheng, Jianjiang Feng, Jie Zhou, and Jiwen Lu. Awarevln: Reasoning with self-awareness for vision-language navigation. *arXiv preprint arXiv:2605.22816*, 2026.

Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. Towards learning a generic agent for vision-and-language navigation via pre-training. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13137–13146, 2020.

Xiaoshuai Hao, Lei Zhou, Zhijian Huang, Zhiwen Hou, Yingbo Tang, Lingfeng Zhang, Guang Li, Zheng Lu, Shuhuai Ren, Xianhui Meng, et al. Mimo-embodied: X-embodied foundation model technical report. *arXiv preprint arXiv:2511.16518*, 2025.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. Dual learning for machine translation. In *Advances in Neural Information Processing Systems*, 2016.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15439–15449, 2022.

Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.

Xianghao Kong, Jinyu Chen, Wenguan Wang, Hang Su, Xiaolin Hu, Yi Yang, and Si Liu. Controllable navigation instruction generation with chain of thought prompting. In *European Conference on Computer Vision*, pp. 37–54. Springer, 2024.

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 *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII 16*, pp. 104–120. Springer, 2020.

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)*, pp. 4392–4412, 2020.

Songze Li, Zun Wang, Gengze Zhou, Jialu Li, Xiangyu Zeng, Ziyang Gong, Limin Wang, Yu Qiao, Qi Wu, Mohit Bansal, et al. Learning goal-oriented vision-and-language navigation with self-improving demonstrations at scale. *arXiv preprint arXiv:2509.24910*, 2025.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Bingqian Lin, Yunshuang Nie, Khun Loun Zai, Ziming Wei, Mingfei Han, Rongtao Xu, Minzhe Niu, Jianhua Han, Liang Lin, Cewu Lu, et al. Evolvenav: Self-improving embodied reasoning for llm-based vision-language navigation. *arXiv preprint arXiv:2506.01551*, 2025.

Yuxing Long, Wenzhe Cai, Hongcheng Wang, Guanqi Zhan, and Hao Dong. Instructnav: Zero-shot system for generic instruction navigation in unexplored environment, 2024. URL https://arxiv.org/abs/2406.04882.

Xiaojian Ma, Silong Yong, Zilong Zheng, Qing Li, Yitao Liang, Song-Chun Zhu, and Siyuan Huang. Sqa3d: Situated question answering in 3d scenes. *arXiv preprint arXiv:2210.07474*, 2022.

Fanqing Meng, Lingxiao Du, Qiguang Chen, Ziqi Zhao, Haocheng Lu, Mengkang Hu, and Michael Qizhe Shieh. Rsibench-data: Benchmarking data-centric research for recursive self-improvement, 2026.

Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. Reverie: Remote embodied visual referring expression in real indoor environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9982–9991, 2020.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

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 *EMNLP*, 2021.

Hao Tan, Licheng Yu, and Mohit Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2610–2621, 2019.

Huajie Tan, Enshen Zhou, Zhiyu Li, Yijie Xu, Yuheng Ji, Xiansheng Chen, Cheng Chi, Pengwei Wang, Huizhu Jia, Yulong Ao, et al. Robobrain 2.5: Depth in sight, time in mind. *arXiv preprint arXiv:2601.14352*, 2026.

Brain Team, Ziyang Gong, Haoming Gu, Zehang Luo, Tianyi Zhang, Tao Tao, Yixiao Chi, Zhe Liu, Lingsi Zhu, Jingyuan Liu, Anke Tang, Songze Li, Yilun Kong, Ningjing Liu, Tianyu Zhu, Yunpeng Qing, Shuang Luo, Xiang Liu, Shi Fu, Dawei Nie, Sixiang Liu, Zhexi Wen, Feng Pan, Xiaofeng Wang, Zhi Hou, Chunxiao Liu, Xue Yang, Junchi Yan, Hengshuang Zhao, Dacheng Tao, and Xiaogang Wang. Ace-brain-0.5: A unified embodied foundational model for physical agentic ai, 2026.

Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. Vision-and-dialog navigation. In *Conference on Robot Learning*, pp. 394–406. PMLR, 2020.

Hanqing Wang, Wei Liang, Jianbing Shen, Luc Van Gool, and Wenguan Wang. Counterfactual cycle-consistent learning for instruction following and generation in vision-language navigation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15471–15481, 2022.

Xiaohan Wang, Wenguan Wang, Jiayi Shao, and Yi Yang. Lana: A language-capable navigator for instruction following and generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 19048–19058, 2023a.

Zihan Wang, Xiangyang Li, Jiahao Yang, Yeqi Liu, and Shuqiang Jiang. Gridmm: Grid memory map for vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 15625–15636, 2023b.

Zihan Wang, Xiangyang Li, Jiahao Yang, Yeqi Liu, Junjie Hu, Ming Jiang, and Shuqiang Jiang. Lookahead exploration with neural radiance representation for continuous vision-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13753–13762, 2024.

Zihan Wang, Yaohui Zhu, Gim Hee Lee, and Yachun Fan. Navrag: Generating user demand instructions for embodied navigation through retrieval-augmented llm. *arXiv preprint arXiv:2502.11142*, 2025a.

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, 2026.

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 *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11975–11986. IEEE, 2023c.

Zun Wang, Jialu Li, Yicong Hong, Songze Li, Kunchang Li, Shoubin Yu, Yi Wang, Yu Qiao, Yali Wang, Mohit Bansal, et al. Bootstrapping language-guided navigation learning with self-refining data flywheel. In *International Conference on Learning Representations*, volume 2025, pp. 23542–23568, 2025b.

Meng Wei, Chenyang Wan, Xiqian Yu, Tai Wang, Yuqiang Yang, Xiaohan Mao, Chenming Zhu, Wenzhe Cai, Hanqing Wang, Yilun Chen, Xihui Liu, and Jiangmiao Pang. Streamvln: Streaming vision-and-language navigation via slowfast context modeling, 2025. URL `https://arxiv.org/abs/2507.05240`.

Meng Wei, Chenyang Wan, Peng Peng, Xiqian Yu, Yuqiang Yang, Delin Feng, Wenzhe Cai, Chenming Zhu, Tai Wang, Jiangmiao Pang, et al. Ground slow, move fast: A dual-system foundation model for generalizable vision-language navigation. In *International Conference on Learning Representations*, volume 2026, pp. 12380–12396, 2026.

Yixuan Weng, Minjun Zhu, Fei Xia, Bin Li, Shizhu He, Shengping Liu, Bin Sun, Kang Liu, and Jun Zhao. Large language models are better reasoners with self-verification. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 2550–2575, 2023.

Wei Xiong, Jiarui Yao, Yuhui Xu, Bo Pang, Lei Wang, Doyen Sahoo, Junnan Li, Nan Jiang, Tong Zhang, Caiming Xiong, et al. A minimalist approach to llm reasoning: from rejection sampling to reinforce. *arXiv preprint arXiv:2504.11343*, 2025.

Jihan Yang, Shusheng Yang, Anjali W Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. Thinking in space: How multimodal large language models see, remember, and recall spaces. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 10632–10643, 2025.

Shusheng Yang, Jihan Yang, Pinzhi Huang, Ellis Brown, Zihao Yang, Yue Yu, Shengbang Tong, Zihan Zheng, Yifan Xu, Muhan Wang, et al. Cambrian-s: Towards spatial supersensing in video. In *International Conference on Learning Representations*, volume 2026, pp. 78185–78225, 2026.

Baiqiao Yin, Qineng Wang, Pingyue Zhang, Jianshu Zhang, Kangrui Wang, Zihan Wang, Jieyu Zhang, Keshigeyan Chandrasegaran, Han Liu, Ranjay Krishna, et al. Spatial mental modeling from limited views. In *Structural Priors for Vision Workshop at ICCV'25*, 2025.

Zhuoyuan Yu, Yuxing Long, Zihan Yang, Chengyan Zeng, Hongwei Fan, Jiyao Zhang, and Hao Dong. Correctnav: Self-correction flywheel empowers vision-language-action navigation model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18737–18745, 2026. doi: 10.1609/AAAI.V40I22.38942.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. Self-rewarding language models. *arXiv preprint arXiv:2401.10020*, 2024.

Yifu Yuan, Yaoting Huang, Xianze Yao, Yutong Li, Shuoheng Zhang, Linqi Han, Pengyi Li, Jiangeng Sun, Wenting Jia, Zhao Zhang, et al. Embodied-r1. 5: Evolving physical intelligence via embodied foundation models. *arXiv preprint arXiv:2606.11324*, 2026.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah Goodman. Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488, 2022.

Haitian Zeng, Xiaohan Wang, Wenguan Wang, and Yi Yang. Kefa: A knowledge enhanced and fine-grained aligned speaker for navigation instruction generation. *arXiv preprint arXiv:2307.13368*, 2023.

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.

Fuxiang Zhang, Jiacheng Xu, Chaojie Wang, Ce Cui, Yang Liu, and Bo An. Incentivizing llms to self-verify their answers. In *Advances in Neural Information Processing Systems*, 2025a.

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

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, 2025b. URL https://arxiv.org/abs/2412.06224.

Jiazhao Zhang, Anqi Li, Yunpeng Qi, Minghan Li, Jiahang Liu, Shaoan Wang, Haoran Liu, Gengze Zhou, Yuze Wu, Xingxing Li, et al. Embodied navigation foundation model. In *International Conference on Learning Representations*, volume 2026, pp. 127293–127322, 2026.

Duo Zheng, Shijia Huang, Lin Zhao, Yiwu Zhong, and Liwei Wang. Towards learning a generalist model for embodied navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13624–13634, 2024.

Gengze Zhou, Yicong Hong, Zun Wang, Xin Eric Wang, and Qi Wu. Navgpt-2: Unleashing navigational reasoning capability for large vision-language models. In *ECCV*, pp. 260–278. Springer, 2024.

Chenming Zhu, Tai Wang, Wenwei Zhang, Jiangmiao Pang, and Xihui Liu. Llava-3d: A simple yet effective pathway to empowering lmms with 3d-awareness. *arXiv preprint*, 2024.

Fengda Zhu, Xiwen Liang, Yi Zhu, Qizhi Yu, Xiaojun Chang, and Xiaodan Liang. Soon: Scenario oriented object navigation with graph-based exploration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12689–12699, 2021.

6 APPENDIX

A EVALUATION METRICS

We evaluate navigation using NE, SR, OSR, SPL, and nDTW. For episode i, let $P_i = (p_{i,0}, \ldots, p_{i,T_i})$ denote the executed trajectory, g_i the goal, and $R_i = (r_{i,1}, \ldots, r_{i,M_i})$ the reference trajectory. We use a success threshold of $\delta = 3$ m. Let d_{geo} denote geodesic distance on the navigation mesh and $\mathbf{1}[\cdot]$ the indicator function.

Navigation Error (NE). Measuring the geodesic distance between the final position and the goal:

$$\text{NE} = \frac{1}{N} \sum_{i=1}^{N} d_{\text{geo}}(p_{i,T_i}, g_i), \tag{6}$$

where N is the number of evaluation episodes. Lower NE indicates that the agent finishes closer to the goal.

Success Rate (SR). An episode is successful if agent executes `STOP` within the success threshold:

$$S_i = \mathbf{1}[\texttt{STOP} \text{ is executed}]\mathbf{1}[d_{\text{geo}}(p_{i,T_i}, g_i) < \delta], \qquad \text{SR} = \frac{1}{N} \sum_{i=1}^{N} S_i. \tag{7}$$

Oracle Success Rate (OSR). OSR measures whether the agent enters the success region at any point during navigation, regardless of where it stops:

$$\text{OSR} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\left[\min_{0 \le t \le T_i} d_{\text{geo}}(p_{i,t}, g_i) < \delta\right]. \tag{8}$$

Success weighted by Path Length (SPL). SPL considers both success and navigation efficiency:

$$\text{SPL} = \frac{1}{N} \sum_{i=1}^{N} S_i \frac{\ell_i}{\max(\ell_i, L_i)}, \tag{9}$$

where $\ell_i = d_{\text{geo}}(p_{i,0}, g_i)$ is the shortest start-to-goal distance and $L_i = \sum_{t=1}^{T_i} \|p_{i,t} - p_{i,t-1}\|_2$ is the traveled distance. Failed episodes receive zero, while successful episodes are penalized for taking detours.

Normalized Dynamic Time Warping (nDTW). nDTW measures how closely the executed trajectory follows the reference trajectory while preserving traversal order. The alignment cost is

$$\text{DTW}(P_i, R_i) = \min_{W \in \mathcal{W}_i} \sum_{(t,j) \in W} \|p_{i,t} - r_{i,j}\|_2, \tag{10}$$

where $\mathcal{W}_i$ contains monotonic alignments connecting the starts and ends of both trajectories, with each step advancing along either or both sequences. The normalized score is

$$\text{nDTW} = \frac{1}{N} \sum_{i=1}^{N} \exp\left(-\frac{\text{DTW}(P_i, R_i)}{M_i \delta}\right). \tag{11}$$

Our implementation removes consecutive duplicate predicted positions and uses FastDTW by default to approximate the alignment cost. Higher nDTW indicates better agreement with the reference route.

NE is reported in meters. SR, OSR, SPL, and nDTW are reported on a percentage scale, with higher values indicating better performance.

Table A1: Action representation ablation on R2R val-unseen.

Representation	NE ↓	OSR ↑	SR ↑	SPL ↑
Arrow	6.32	51.22	43.28	37.80
Single	**5.56**	**57.48**	**48.72**	**42.67**
Double	5.92	53.78	43.83	39.22

B Action Representation

We compare three action representations: **Arrow** uses ↑ / ← / → and `stop`, **Single** uses `forward/left/right/stop`, and **Double** uses `move forward/turn left/turn right/stop`.

Table A1 shows that Single achieves the best performance across all four metrics. Compared with Arrow, it improves SR and SPL by 5.44 and 4.87 percentage points, respectively, and also outperforms Double. These results support using single-word action representations in our navigation model.

C Action Inputs and Trajectory Sampling

We compare **V2L**, which generates instructions from images alone, with **VA2L**, which also uses actions. We consider two frame sampling strategies: **Uniform-32** uniformly samples up to 32 frames across the trajectory, while **Interval-4** samples one frame every four action steps.

Table A2: Downstream navigation results with different action inputs and frame sampling strategies.

Representation	Supervision	Sampling	NE ↓	OSR ↑	SR ↑	SPL ↑
Arrow	V2L	Interval-4	6.39	48.23	41.44	36.27
Arrow	VA2L	Interval-4	6.20	47.14	41.49	37.62
Arrow	V2L	Uniform-32	6.41	48.67	42.09	36.82
Arrow	VA2L	Uniform-32	5.71	50.68	44.15	39.54
Single	V2L	Interval-4	5.40	54.98	47.25	40.59
Single	VA2L	Interval-4	5.42	55.36	48.02	41.36
Single	V2L	Uniform-32	5.15	58.34	51.22	42.88
Single	VA2L	Uniform-32	**5.00**	**60.14**	**53.13**	**45.38**

Table A2 shows that Uniform-32 consistently improves SR and SPL over Interval-4 across both action representations and supervision settings. Adding action inputs through VA2L further improves these two metrics in all settings. Combining Single, VA2L, and Uniform-32 achieves the best overall performance, with an SR of 53.13 and an SPL of 45.38.

D Discrete-to-Continuous Trajectory Conversion

We convert discrete navigation paths into executable trajectories in Habitat. Given a path $\tau^G = (v_1, \ldots, v_N)$, we query MatterSim for each viewpoint's position $p_i^G = (x_i, y_i, z_i)$ and map it to Habitat coordinates:

$$\tilde{w}_i = (x_i, z_i - 1.25, -y_i), \qquad w_i = (x_i, h_i, -y_i), \tag{12}$$

where 1.25 m is the camera-height offset and h_i is the floor height obtained by snapping $\tilde{w}_i$ to the navigation mesh. If snapping fails, we use $h_i = z_i - 1.25$.

Starting from the first waypoint with the converted initial orientation, a shortest-path follower follows the waypoints in order, producing

$$\tau^G \xrightarrow{\text{waypoint following}} \tau^C = (s_0, a_1, s_1, \ldots, a_L, s_L), \tag{13}$$

Algorithm 1 Discrete-to-Continuous Trajectory Conversion

Require: Discrete path $\tau^G = (v_1, \ldots, v_N)$, initial orientation, navigation mesh $\mathcal{M}$, action budget $B = 500$
Ensure: Habitat waypoints W, initial state s_0, and action sequence A, or rejection

$W \leftarrow [\,]$, $A \leftarrow [\,]$
for $i = 1, \ldots, N$ **do**
  Query MatterSim for (x_i, y_i, z_i) at v_i
  $\tilde{w}_i \leftarrow (x_i, z_i - 1.25, -y_i)$
  $q_i \leftarrow \mathrm{Snap}_{\mathcal{M}}(\tilde{w}_i)$
  $w_i \leftarrow \tilde{w}_i$
  if q_i is valid **then**
    $w_i \leftarrow (x_i, [q_i]_y, -y_i)$ {Adjust height only}
  end if
  Append w_i to W
end for
if $d_{\mathcal{M}}(w_1, w_N) \geq 100\,\mathrm{m}$ **then**
  return rejection
end if
Initialize Habitat at w_1 with the converted initial orientation; record s_0
Initialize shortest-path follower F with goal radius $0.25\,\mathrm{m}$
for $i = 2, \ldots, N$ **do**
  loop
    if $|A| \geq B$ **then**
      return rejection
    end if
    $a \leftarrow F.\mathrm{NextAction}(w_i)$
    if $a =$ `STOP` or $a =$ `None` **then**
      break
    end if
    Execute a in Habitat and append a to A
  end loop
end for
return (W, s_0, A)

where s_l denotes the agent's position and orientation after action a_l, and L is the number of executed actions. Each movement action advances the agent by 0.25 m or turns it left or right by $15°$. Trajectories that exhaust the action budget are discarded. The recorded actions and rendered observations are used to construct navigation and instruction-generation examples.

E Prompt Templates

E.1 Navigation

The navigation prompt includes up to eight history frames and the current view. The history clause is omitted when no history is available. The model predicts up to four actions per turn; the responses below illustrate the dialogue format.

Navigation Prompt

User: You are an autonomous navigation assistant. Your task is to {instruction}. Devise an action sequence to follow the instruction using the four actions: TURN LEFT (left) or TURN RIGHT (right) by 15 degrees, MOVE FORWARD (forward) by 25 centimeters, or STOP (stop).
These are your historical observations: {history_images}
This is your current view: <image>
Assistant: forward,forward,left,forward

User: in front of you is <image>
Assistant: forward,stop

E.2 Instruction Generation

VA2L uses the following system and user messages:

Instruction Generation Prompt

System: You are a navigation-instruction generator. Output only the requested instruction.
User: This is a navigation trajectory consisting of several image sequences:
(Viewpoint 1: Image:<image>, Action: {action_span_1}),
...
(Viewpoint K: Image:<image>, Action: {action_span_K}).
Could you give me its corresponding navigation instruction in details?
Assistant: {navigation_instruction}

The viewpoint list is expanded for all sampled frames. Each action span is described using phrases such as `forward 2 steps, then left`. V2L removes the action fields.

F Visualization of Spatial Question Answering

Figure 5 presents selected correct predictions on VSI-Bench, ScanQA, SQA3D, and MindCube. These examples cover relative direction reasoning, object identification, situated question answering, and viewpoint-dependent spatial relations. Each example includes one representative image, the question, and the model answer alongside a matching reference answer.

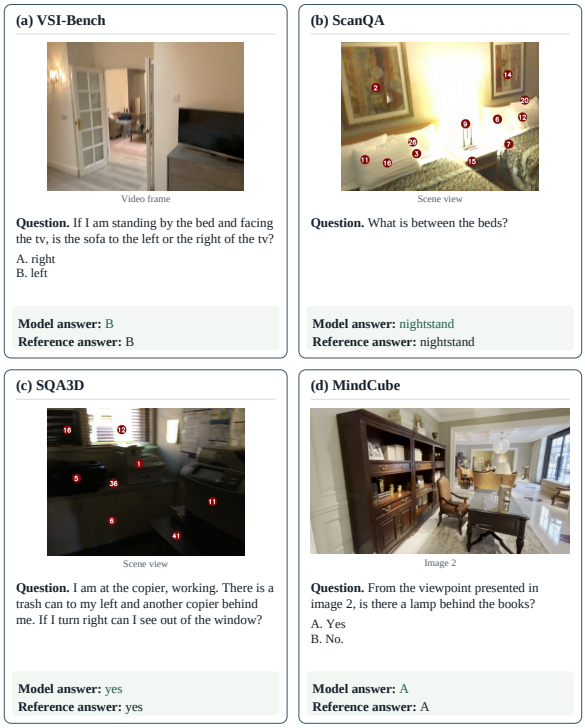


Figure 5: Selected correct QA examples on VSI-Bench, ScanQA, SQA3D, and MindCube.

G VISUALIZATION OF R2R-CE AND RXR-CE

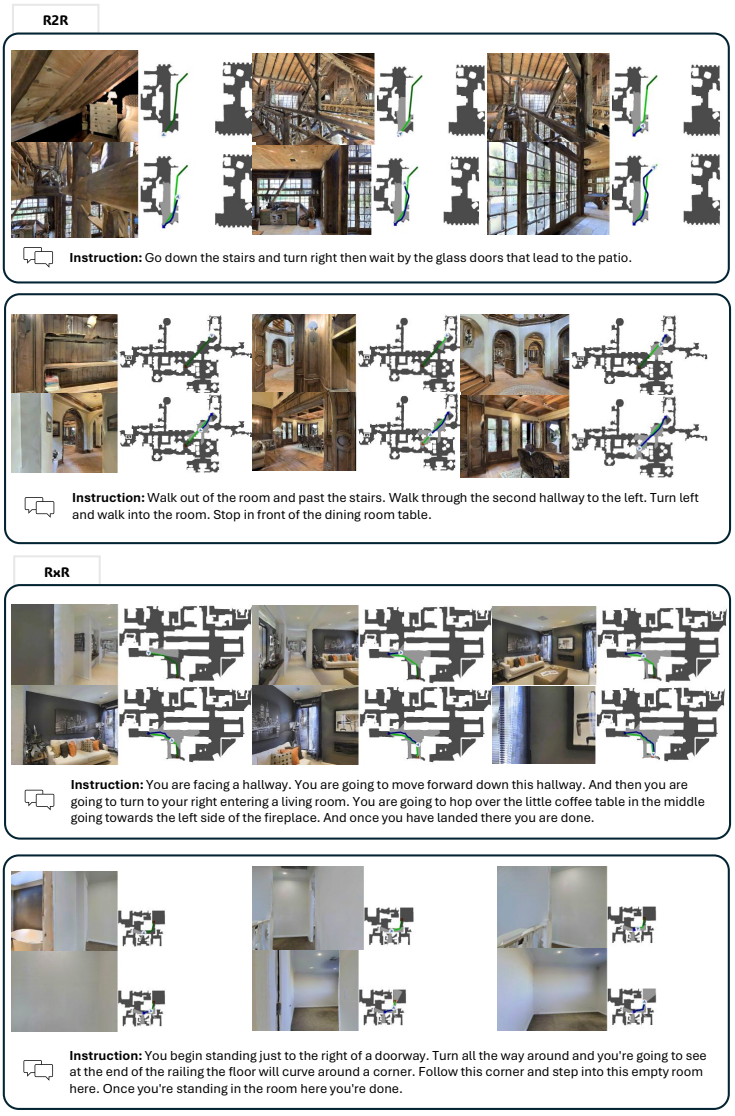


Figure 6: Visualization of navigation trajectories on R2R-CE and RxR-CE.

H LIMITATIONS AND FUTURE WORK

RSI-Nav's self-improvement is constrained by the capabilities of its current policy. Since the same model generates instructions and executes them for verification, the resulting supervision may reinforce existing biases. Instructions that are valid but difficult to execute may also be rejected, potentially favoring easier trajectories and limiting further improvement. Future work could explore difficulty-aware data selection, adaptive filtering, and complementary verification signals to retain challenging examples while controlling supervision quality.

Execution-based filtering also provides an incomplete measure of instruction quality. Reaching the intended destination efficiently does not necessarily imply that every landmark description or spatial relation is correct. Such semantic errors may persist in accepted examples and propagate through subsequent updates. Future work could combine trajectory-level verification with grounded semantic checks and targeted human evaluation to better assess instruction fidelity and understand how errors accumulate across self-improvement rounds.