REDUAL-VLN: RECOVERABLE DUAL-SYSTEM VISION-AND-LANGUAGE NAVIGATION VIA LANGUAGE-CONDITIONED BACKTRACKING

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-and-language navigation (VLN) aims to enable embodied agents to navigate unseen environments by following natural-language instructions. Long-horizon VLN is vulnerable to route deviations during local execution, which can disrupt the remaining task. Recovery requires selecting an appropriate recovery anchor and maintaining task context for subsequent navigation, placing additional reasoning and memory demands on the VLN model. We propose ReDual-VLN, a dual-system framework that separates high-level reasoning from local navigation execution. A VLM-based Route Supervisor performs route planning, progress tracking, and recovery reasoning, while Hierarchical Context Management (HCM) maintains semantic plans, execution history, and recovery anchors. A Backtracking-Enabled VLN (BE-VLN) then focuses on executing local navigation and recovery instructions. To equip BE-VLN with the required execution capabilities, we construct ReTrace-RxR with paired short-horizon navigation and backtracking data, enabling the model to follow local instructions, predict instruction termination, and perform language-conditioned backtracking. When a deviation occurs, the Route Supervisor selects a previously visited anchor on the intended route as the recovery target and instructs BE-VLN to backtrack to it. After verifying the return, the supervisor directs the executor to resume the remaining instructions, forming a closed loop of deviation detection, backtracking-based recovery, and task resumption. Systematic evaluations in simulation and the real world demonstrate that ReDual-VLN improves long-horizon navigation performance and substantially enhances recovery and task resumption following route deviations.

1 INTRODUCTION

Vision-and-Language Navigation (VLN) aims to enable embodied agents to navigate unseen environments based on language instructions and visual observations Zhang et al. (2026c); Ma et al. (2026). Long-horizon VLN instructions usually contain sequential landmarks, motion directions, and spatial relations Liu et al. (2026b); Yin et al. (2025). Agents need not only to generate accurate navigation actions from current observations, but also to infer task progress and remaining objectives from execution history Liu et al. (2026b); Feng et al. (2026). Although recent VLN models have demonstrated strong long-horizon navigation capability, local execution errors tend to accumulate as navigation trajectories become longer Liu et al. (2026b); Zheng et al. (2026b). However, local execution errors can accumulate over long trajectories, making recovery from route deviations increasingly important Shi et al. (2025); Zheng et al. (2026b).

Effective recovery requires more than correcting a local action: the agent must diagnose the deviation, select an appropriate recovery anchor using the instruction, execution history, and current observations, return to it, and resume the unfinished instruction from the correct task state. AwareVLN Guo et al. (2026) monitors sub-task completion, path deviation, and stopping errors; Route2Step Huang et al. (2026b) separates semantic progress analysis from local execution; ReflectVLN Wang et al. (2026a) supports closed-loop correction through reflection and sub-goal updates; and Uni-LaViRA Ding et al. (2026) incorporates backtracking and replanning. However, maintaining task state and resuming unfinished instructions after recovery remain challenging Rui

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.

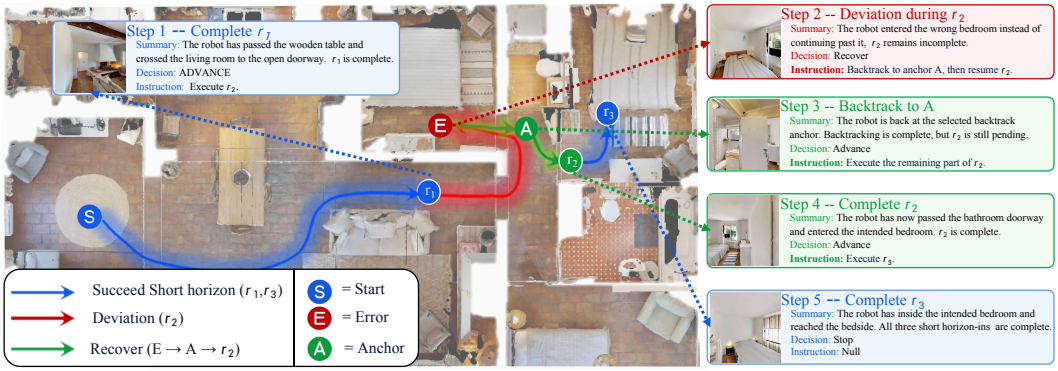

Figure 1: ReDual-VLN enables recovery from route deviations in long-horizon vision-and-language navigation. Instead of allowing local execution errors to disrupt the remaining task, our method enables the agent to return to the intended route and resume unfinished navigation.

et al. (2026); Ma et al. (2026). Returning to a visited location requires identifying a valid anchor from history and retracing the executed path, making recovery dependent on both executable backtracking and effective historical-context management Shi et al. (2025); Yin et al. (2025); Zheng et al. (2026b).

To accommodate this growing dependence on historical context, VLN methods typically require sophisticated history mechanisms to maintain the execution context accumulated throughout the task. Early approaches compress historical observations and actions using recurrent states Anderson et al. (2018); Ku et al. (2020); Ma et al. (2019), while later methods preserve trajectory histories, topological nodes, or map memories Zhang et al. (2025b); Li et al. (2026); An et al. (2023). Recent studies further introduce context compression, selective memory, and hierarchical semantic-spatial memory Wei et al. (2026b); Zeng et al. (2026); Zhang et al. (2025c). However, these approaches still require VLN models to continuously maintain and retrieve accumulated context. This burden becomes greater during recovery, where the model must also identify a recovery anchor and preserve task continuity. As tasks become longer, training and inference burdens increase, while accessible history remains constrained by the model's context window and memory capacity.

To improve recovery in long-horizon navigation tasks while reducing the burden of maintaining historical context on VLN models, we propose ReDual-VLN, a dual-system framework in which an upper VLM performs route-level reasoning and uses Hierarchical Context Management (HCM) to maintain task context, while BE-VLN executes local navigation and recovery instructions. Upon detecting a deviation, the VLM selects a visited anchor, coordinates backtracking and return verification, and resumes the unfinished instruction. We further construct ReTrace-RxR to train BE-VLN for short-horizon execution, precise termination, and language-conditioned backtracking.

Our contributions are summarized as follows:

1. **A dual-system VLN framework for long-horizon deviation recovery.** We propose a cooperative dual-system framework that assigns long-horizon understanding, deviation reasoning, and recovery decisions to the upper VLM, while assigning local navigation execution to the lower VLN. Through language-conditioned backtracking, the framework converts high-level recovery decisions into executable navigation behaviors.

2. **We introduce a hierarchical context management mechanism** where the upper VLM maintains semantic plans, execution history, and recovery anchors, while dynamically generating local instructions. This decouples task progress tracking, recovery-related context, and long-horizon history management from VLN, allowing the VLN to perform long-horizon navigation and recovery without maintaining the complete trajectory history.

3. **A VLN executor and training data for short-horizon navigation and backtracking.** We construct paired forward short-horizon navigation and reverse backtracking data with independent termination boundaries, enabling VLN models to follow local instructions, terminate accurately, and perform language-conditioned backtracking.

2 Related Work

Dual-System Vision-and-Language Navigation. Early VLN methods commonly relied on a single navigation policy to jointly handle instruction understanding, history modeling, and action prediction Anderson et al. (2018); Ku et al. (2020); Krantz et al. (2020). With the development of large-scale Vision-Language Models, recent work has begun to separate task-level reasoning from local navigation execution. Route2Step Huang et al. (2026b) separates semantic progress analysis from local action generation through sub-instruction management. ReflectVLN Wang et al. (2026a) introduces separate intention and execution agents and uses execution feedback to support task completion and deviation correction. AgentVLN Xin et al. (2026a) and Qwen-RobotNav Zhang et al. (2026a) further explore agentic navigation systems in which general VLMs coordinate specialized navigation models or skills. Related dual-system approaches include DualVLN Wei et al. (2026a), which combines a reasoning-oriented VLM with a fast trajectory generator, ABot-N0/N1 Chu et al. (2026); Gong et al. (2026), which coordinate slow reasoning with fast action generation, and SEDualVLN Huang et al. (2026a), which connects high-level planning with low-level execution through spatial representations.

These methods establish different interfaces between high-level reasoning and local execution, but this separation alone does not ensure recovery after route deviation. Backtracking-based recovery requires selecting a previously visited anchor on the intended route, retracing the executed path, and resuming the unfinished instruction. Our work focuses on how a general-purpose VLM and a specialized VLN model coordinate these stages while preserving task continuity throughout recovery.

Historical Context Management in Long-Horizon VLN. Long-horizon navigation requires the agent to retain information about previous observations, actions, and task progress. Early methods compressed historical information into recurrent hidden states Anderson et al. (2018); Ku et al. (2020); Ma et al. (2019). Later approaches explicitly represented trajectory history, topological structure, or spatial memory. Representative examples include BEVBert An et al. (2023), MapNav Zhang et al. (2025b), and HSGM Li et al. (2026).

More recent work has explored more structured ways to maintain long navigation histories. StreamVLN Wei et al. (2026b) uses slow and fast context modeling for continuous visual streams. COSMO Zhang et al. (2025c) selectively preserves historical information, while JanusVLN Zeng et al. (2026) separates semantic and spatial memories. HAM-VLN Liu et al. (2026a), HiMemVLN Lyu et al. (2026), SpaceVLN Deng et al. (2026), and LH-VLN Song et al. (2025) further investigate hierarchical and multi-granularity memory representations for long-horizon navigation.

These methods offer different ways to represent and retain execution history. A complementary question is how context-management responsibilities should be distributed between reasoning and execution, particularly when recovery requires relating previously visited locations to unfinished objectives. Our work adopts a global–local division in which the upper VLM maintains task progress, execution history, and recovery anchors, while the lower VLN uses the current instruction and a bounded recent visual history.

3 Method

3.1 Overview

We propose **ReDual-VLN**, a dual-system framework that separates route-level reasoning and context management from local navigation execution. The upper VLM-based **Route Supervisor** tracks task progress through **Hierarchical Context Management (HCM)** and **Recover Mechanism** to determines when and where to recover from deviations. The lower **Backtracking-Enabled VLN (BE-VLN)** executes the current short horizon or backtracking instruction and submits **Instruction Termination Feedback (ITF)** to the supervisor. Recovery forms a closed loop of deviation detection, language-conditioned backtracking to a visited route anchor, return verification, and task resumption. We construct **ReTrace-RxR** to train the short-horizon navigation, backtracking, and termination-feedback capabilities required by this interaction.

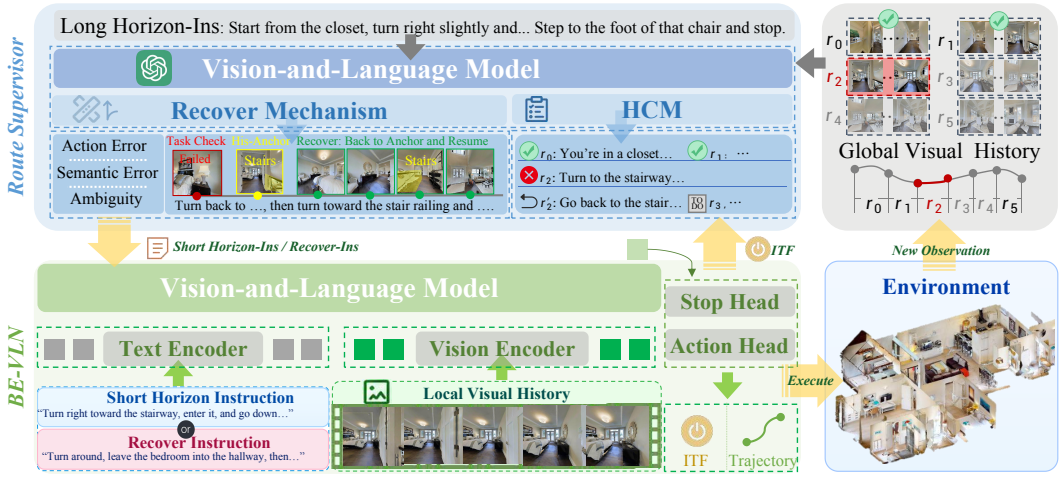

Figure 2: Architecture of ReDual-VLN. The Route Supervisor maintains route-level task progress and execution context through HCM and dispatches short-horizon navigation or recovery instructions. BE-VLN executes the active instruction with bounded local visual history and provides ITF for supervisory review. Through this interaction, ReDual-VLN coordinates route-level reasoning with local navigation execution to support deviation recovery and subsequent task resumption.

3.2 Dual-System VLN

We consider continuous-environment VLN with a full route instruction L and egocentric observations o_t (Ku et al., 2020; Krantz et al., 2020). Before execution, the high-level Router Supervisor decomposes L into an ordered sequence of sub-instructions $\mathcal{R} = (r_1, \ldots, r_K)$. Let j index supervisory reviews and ℓ_j denote the active local instruction before review j. At each execution step, the lower-level BE-VLN model π_θ takes ℓ_j and the local visual context $\mathcal{O}_t^{\text{loc}}$ as input and predicts a waypoint trajectory together with an instruction-completion probability:

$$(\widehat{\mathbf{W}}_t, p_t^{\text{ITF}}) = \pi_\theta(\mathcal{O}_t^{\text{loc}}, \ell_j), \qquad \widehat{\mathbf{W}}_t \in \mathbb{R}^{H\times 3}. \tag{1}$$

Here, $H = 8$, and each waypoint specifies a planar position and heading relative to the current agent frame. The local context $\mathcal{O}_t^{\text{loc}}$ contains the most recent 32 observations, including the current frame, or all available observations when fewer than 32 have been collected. The output $p_t^{\text{ITF}} \in [0, 1]$ estimates the probability that the active instruction ℓ_j has been completed. It provides a signal for supervisory review rather than directly issuing an environment-level STOP action.

The Router Supervisor is invoked when ITF satisfies the review condition or a periodic review is due, allowing intervention during instruction execution rather than only at its endpoint. At review time t_j, it receives the precomputed sub-instruction sequence $\mathcal{R}$, the complete visual observation history $\mathcal{H}_j = (o_0, \ldots, o_{t_j})$, the active instruction ℓ_j, and the current completion probability $p_{t_j}^{\text{ITF}}$. Using its internally maintained route-progress and recovery state, it produces a supervisory decision and the subsequent local instruction:

$$(d_j, \ell_{j+1}) = \Psi_{\text{review}}(\mathcal{R}, \mathcal{H}_j, \ell_j, p_{t_j}^{\text{ITF}}). \tag{2}$$

The decision d_j has four possible outcomes. **CONTINUE** retains the active instruction, with $\ell_{j+1} = \ell_j$. **ADVANCE** dispatches the next required sub-instruction from $\mathcal{R}$ after verifying completion of the active instruction. **RECOVER** identifies a recovery anchor from $\mathcal{H}_j$ and generates a corrective instruction that guides BE-VLN back to that anchor while preserving the unfinished route objective. **STOP** terminates the episode only after the final sub-instruction and the overall route-completion conditions have been verified; in this case, $\ell_{j+1} = \varnothing$, and the decision is converted into a Habitat STOP action. Thus, only ADVANCE and RECOVER dispatch a new local instruction to BE-VLN. Additional state-transition details appear in Appendix A.1.

3.2.1 RECOVERY MECHANISM

Deviation detection and anchor selection. We define recovery as returning from a deviated position to a previously visited anchor on the intended route and then resuming the unfinished instruction. During either type of review, the Route Supervisor assesses consistency between the observed execution and the pending route objective. If a deviation is detected, it selects a return target from $\mathcal{A}_j$. Anchors include verified instruction boundaries and salient intermediate locations, such as doorways or corridor junctions, supported by visual evidence and associated route progress. A visited location qualifies only when the instruction and execution evidence place it on the intended route. The supervisor prioritizes an unambiguous intermediate anchor close to the deviation along the traversed path and suitable for resumption. A verified starting boundary serves as a fallback when intermediate evidence is insufficient.

Language-conditioned backtracking. Using the selected anchor and evidence of the actual traversal, the supervisor generates a backtracking instruction. BE-VLN executes it through the same policy in Eq. equation 1, retracing the visited route from its actual deviated position, including the off-route portion. The target is specified in language rather than by an additional goal image or coordinate. Backtracking therefore does not require completing the interrupted instruction or returning to its starting boundary.

Return verification and task resumption. Backtracking remains subject to ITF-triggered and periodic review. An unfinished, consistent return is continued without advancing route progress. Once the supervisor verifies arrival at the selected anchor, ADVANCE replaces the backtracking command with an instruction expressing the pending objective's remaining requirements from the recovered state. This transition does not itself increment the route-objective index. In particular, an intermediate-anchor return need not replay the entire original short instruction, and anchor arrival does not establish completion of that instruction. Recovery thus reconnects physical return with the correct task state rather than treating backtracking alone as task recovery.

3.2.2 HIERARCHICAL CONTEXT MANAGEMENT

HCM maintains episode-level task state at the Router Supervisor while bounding BE-VLN's execution context. The supervisor records instruction transitions, verified outcomes, and recovery anchors, updating its state after every review, including CONTINUE. A route index k_j identifies the first unfinished objective r_{k_j}, whereas ℓ_j specifies the current execution instruction. Recovery replaces ℓ_j without advancing k_j; after verified backtracking, execution resumes the pending objective. Route progress advances only upon verified completion of the corresponding original objective.

BE-VLN receives only ℓ_j and $\mathcal{O}_t^{\text{loc}} = (o_{\max(0,t-31)}, \ldots, o_t)$, containing the latest 32 frames, including the current frame, or all available frames when fewer exist. This window may span instruction switches. The complete history and recovery records remain at the supervisory level, enabling long-horizon reasoning without expanding BE-VLN's local context.

3.3 BE-VLN: DATA CONSTRUCTION AND TRAINING

3.3.1 RETRACE-RXR: SHORT-HORIZON NAVIGATION AND BACKTRACKING DATA

Shared-Boundary Navigation. We transfer the sub-instruction–subpath alignments in Landmark-RxR (He et al., 2021), building on fine-grained instruction supervision (Hong et al., 2020), to RxR-CE rollouts and partition them into short instruction-conditioned segments. Adjacent segments share the same physical boundary state, but each receives its own instruction and terminal annotation. Consequently, a boundary observation can supervise termination under the preceding instruction and continued motion under the next one. These instruction-specific endings provide local termination targets rather than inheriting only the complete route's destination.

Anchor-Grounded Backtracking. For each eligible segment, we generate a return demonstration from its realized endpoint to its visited starting anchor, restoring an orientation suitable for subsequent navigation. The agent retraces the source route, and return observations are rendered anew in simulation rather than obtained by reversing forward images. A multimodal model generates the backtracking instruction from the source instruction and temporally ordered views of both traversals. Each forward–backtracking pair has direction-specific instructions, trajectories, and terminal

boundaries. Pairing is partial, so valid forward segments remain available even when no valid return is obtained. Boundary alignment, return synthesis, and validation are detailed in Appendix A.2.

3.3.2 Executor and Instruction Termination Feedback

BE-VLN is initialized from the general-purpose pretrained **Qwen3-VL-8B** model, with a trajectory head and a stop head following the navigation architecture in SPAN-Nav (Liu et al., 2026c). The stop head predicts q_t, giving $p_t^{\mathrm{ITF}} = \sigma(q_t)$. Full-route navigation and both ReTrace-RxR subsets share the backbone and both heads, with behavior conditioned on the assigned instruction rather than a separate backtracking policy.

Every navigation sample supervises trajectory prediction and an instruction-relative ITF label. Let n_t^{rem} denote the remaining trajectory records, including the terminal annotation, before the current action. We set $z_t = \mathbf{1}[n_t^{\mathrm{rem}} \le H]$ relative to the full-route destination, short-instruction endpoint, or backtracking target. This label indicates a demonstrated terminal window, not verified satisfaction of every spatial relation, motivating supervisory verification. Let $\mathcal{D}_{\mathrm{nav}}$ contain all navigation examples and $\mathcal{D}_{\mathrm{qa}}$ contain general question-answering data involving images and videos. Following SPAN-Nav (Liu et al., 2026c), we optimize

$$\mathcal{L} = \mathbb{E}_{\xi\sim\mathcal{D}_{\mathrm{nav}}}\big[\lambda_{\mathrm{traj}}\mathcal{L}_{\mathrm{traj}}(\xi) + \lambda_{\mathrm{ITF}}\mathcal{L}_{\mathrm{ITF}}(\xi)\big] + \lambda_{\mathrm{qa}}\mathbb{E}_{\eta\sim\mathcal{D}_{\mathrm{qa}}}\big[\mathcal{L}_{\mathrm{qa}}(\eta)\big], \tag{3}$$

where the three losses are squared waypoint error, binary cross-entropy for ITF, and token-level cross-entropy for question answering. QA examples do not receive trajectory or ITF supervision. Architecture, causal history encoding, trajectory normalization, and supervision details are provided in Appendices A.3–A.4.

4 Experiments

We conduct experiments to evaluate ReDual-VLN from both system- and component-level perspectives. At the system level, we examine whether the complete framework improves long-horizon navigation performance. We then analyze the two key mechanisms that support this improvement, namely recovery and hierarchical context management, before evaluating whether BE-VLN provides the local execution capabilities required by the dual-system framework. Specifically, our experiments address four questions. **Q1:** Does ReDual-VLN improve long-horizon navigation performance? **Q2:** Does recovery improve task completion beyond hierarchical execution alone? **Q3:** Can hierarchical context management improve navigation while relieving the executor of route-level history modeling and progress tracking? **Q4:** Does BE-VLN provide the short-horizon navigation, language-conditioned backtracking, and instruction-termination feedback capabilities required by the framework? We first report overall performance, then jointly examine HCM and recovery, and finally evaluate the executor's capabilities and its integration with the Route Supervisor.

4.1 Experimental Setup

Benchmarks. We evaluate full-route navigation on the validation-unseen splits of the continuous-environment R2R-CE and RxR-CE benchmarks Anderson et al. (2018); Ku et al. (2020). RxR-CE evaluation uses the English subset containing 3,669 episodes. We also evaluate the executor on two test sets constructed by our ReTrace-RxR: a short-horizon navigation test set formed from shared-boundary navigation, and a language-conditioned backtracking test set in which the target is a previously visited route anchor. Each local instruction is evaluated independently of the original full-route task. Comparisons follow the observation configurations and evaluation scopes indicated in the corresponding tables.

Metrics. To evaluate navigation performance, we use standard evaluation metrics (Anderson et al., 2018), including Success Rate (SR), Oracle Success Rate (OS), Success Rate Weighted by Path Length (SPL), normalized Dynamic Time Warping (nDTW), and Navigation Error relative to the goal (NE). Note that the success criteria for different navigation tasks may vary; therefore, we use the default success criteria defined for each benchmark. Implementation details and comparison controls are provided in Appendix B.

Table 1: Full-route navigation on R2R-CE and RxR-CE. Within each camera group, baselines are sorted by RxR Val-Unseen SR; our three configurations appear below.

Method	Dual System	R2R Val-Unseen				RxR Val-Unseen			
		NE↓	OSR↑	SR↑	SPL↑	NE↓	SR↑	SPL↑	nDTW↑
Monocular methods									
Uni-NaVid Zhang et al. (2025a)	✗	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. (2026b)	✗	4.98	64.2	56.9	51.9	6.22	52.9	46.0	61.9
DecoVLN Xin et al. (2026b)	✗	5.01	63.5	56.3	50.5	5.73	54.2	46.3	63.5
BudVLN He et al. (2026)	✗	4.74	65.6	57.6	51.1	5.79	56.1	46.6	63.2
JanusVLN Zeng et al. (2026)	✗	4.78	65.2	60.5	56.8	6.06	56.2	47.5	62.1
NavFoM Zhang et al. (2026b)	✗	5.01	64.9	56.2	51.2	5.51	57.4	49.4	60.2
DualVLN Wei et al. (2026a)	✓	4.05	70.7	64.3	58.5	4.58	61.4	51.8	70.0
SEDualVLN Huang et al. (2026a)	✓	3.75	73.7	67.3	62.5	4.12	63.9	52.4	72.8
AwareVLN Guo et al. (2026)	✗	4.02	73.5	65.4	55.1	3.95	67.6	56.1	65.7
CorrectNav Yu et al. (2026)	✗	4.24	67.5	65.1	62.3	4.09	69.3	63.3	75.2
LightNav-0 Wang et al. (2026b)	✗	3.91	73.7	68.5	62.8	3.66	73.6	64.5	67.4
Efficient-VLN Zheng et al. (2026a)	✗	3.29	78.2	73.2	65.0	3.18	75.6	60.0	–
Multi-view methods									
NavFoM Zhang et al. (2026b)	✗	4.61	72.1	61.7	55.3	4.74	64.4	56.2	65.8
SPAN-Nav Liu et al. (2026c)	✗	4.07	75.3	66.3	59.3	4.20	69.7	60.1	67.9
OmniNav Xue et al. (2026)	✗	3.74	74.6	69.5	66.1	3.77	73.6	62.0	–
Qwen-RobotNav-8B Zhang et al. (2026a)	✗	3.53	78.5	72.1	**66.6**	3.58	76.5	**65.7**	**72.5**
BE-VLN(ours)	✗	3.58	73.6	68.2	62.4	4.11	70.4	61.6	68.6
ReDual-VLN w/o RECOVER	✓	3.26	74.3	69.6	63.2	3.92	72.3	60.0	65.8
ReDual-VLN(ours)	✓	**2.91**	**82.98**	**76.7**	65.7	**2.80**	**78.5**	64.9	70.5

Monocular and multi-view refer to RGB camera inputs. †StreamVLN also uses depth information. ✓ and ✗ indicate whether two distinct decision modules are active in the reported VLN-CE configuration.

4.2 Overall Navigation Performance

To answer Q1, we evaluate the complete ReDual-VLN framework against existing navigation methods on both full-route benchmarks. Table 1 reports the system-level comparison and places the standalone executor and the two hierarchical configurations alongside other full-route navigation models. We use the complete configuration in Sec. 3: the Route Supervisor provides HCM and recovery, and BE-VLN provides short-horizon navigation, language-conditioned backtracking, and ITF.

Overall task completion. ReDual-VLN achieves the highest SR on both benchmarks within the comparable evaluation scope in Table 1. The consistent improvement on R2R-CE and RxR-CE establishes the effectiveness of the complete framework for long-horizon instruction following. Table 1 further shows that the complete system outperforms standalone BE-VLN, although both use the same navigation checkpoint. The gain therefore extends beyond the executor's standalone navigation capability and reflects the benefit of organizing its execution through high-level supervision. Sec. 4.3 separates the contributions of HCM and recovery within this improvement.

Trajectory characteristics. The leading success rates are accompanied by SPL and nDTW values that are not the highest among the compared methods. Because a successfully recovered trajectory retains the movement made before the deviation was corrected, the backtracking segment, and the subsequent execution of the remaining route. Backtrack from the deviated position necessarily adds traveled distance, while reverse and repeated traversal alter the temporal alignment with the reference path. These movements remain in the evaluated trajectory even when the return and subsequent navigation are correct.

Final localization and termination. ReDual-VLN also achieves the lowest NE within the comparable evaluation scope, indicating more accurate final localization. This result is consistent with combining instruction termination feedback with hierarchical context management.

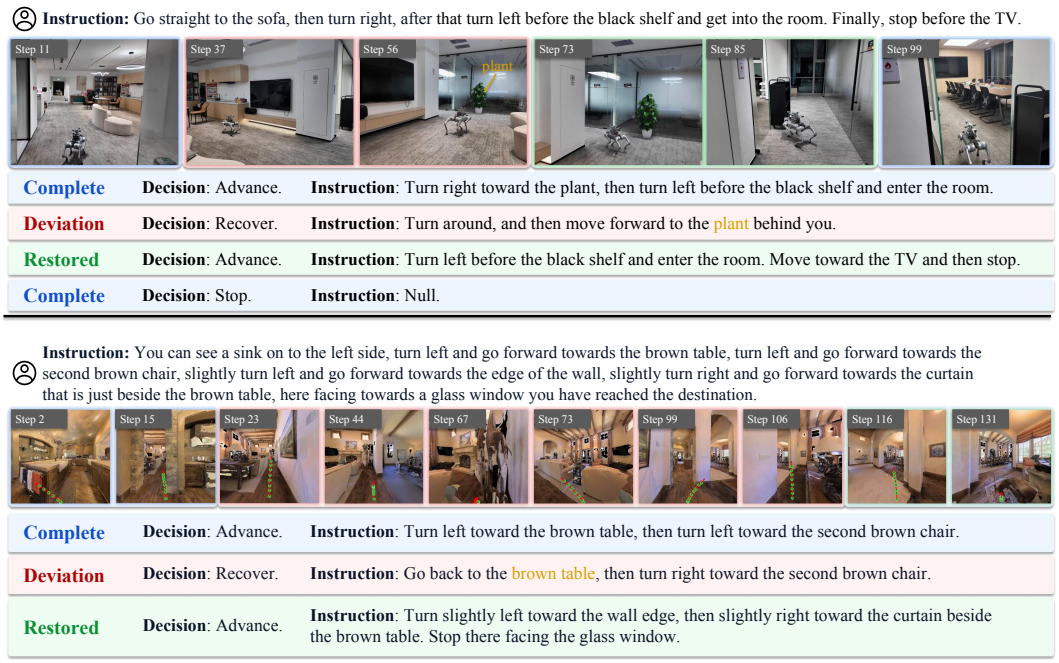

Figure 3: Backtracking Example. Successful recovery and task completion of originally failed episodes on the real world and RxR-CE unseen split with our framework.

4.3 Ablation of Hierarchical Context Management and Recovery

To jointly answer Q2 and Q3, we test whether recovery improves task completion beyond hierarchical execution and whether HCM improves navigation while transferring route-level history modeling away from BE-VLN. Both questions are evaluated using the bottom three rows of Table 1.

BE-VLN executes the original full-route instruction without the Route Supervisor. *ReDual-VLN w/o RECOVER*, equivalently *BE-VLN + HCM*, retains hierarchical context management and execution review but disables supervisor-directed recovery mechanism. *ReDual-VLN* is the complete configuration in Sec. 3. The three configurations use same BE-VLN checkpoint, and the two hierarchical variants additionally share the Route Supervisor and review settings.

Recovery and task resumption (Q2). Complete ReDual-VLN outperforms the recovery-disabled configuration. Since both variants already use HCM and the same supervisory review settings, this improvement identifies the additional benefit of enabling recovery. The no-recovery system can organize local execution and assess progress, but it cannot respond to a deviation by issuing a return-to-anchor instruction and then resuming the pending objective. The full system can carry out that intervention through BE-VLN. Its higher task success therefore supports recovery as a means of continuing an interrupted route, beyond the benefit already provided by hierarchical task organization.

Hierarchical context management (Q3). ReDual-VLN w/o RECOVER also outperforms standalone BE-VLN. This improvement demonstrates that HCM itself benefits long-horizon navigation. The Route Supervisor maintains the route-level state and expresses the active objective as a local instruction, while BE-VLN uses the same bounded recent visual window for execution. Moving global progress tracking and history association to the supervisory level therefore does not merely avoid a performance loss; it improves navigation with an unchanged executor.

Together, the two comparisons identify complementary gains. HCM improves the organization of long-horizon execution, and recovery adds the ability to return from a deviation and continue the unfinished task. The improvement of the full framework over standalone BE-VLN consequently reflects both components rather than recovery alone.

4.4 CAPABILITY ANALYSIS OF BE-VLN

To answer Q4, we compare NavFoM, SPAN-Nav, and BE-VLN on short-horizon navigation and language-conditioned backtracking tasks without Route Supervisor intervention, and make the ablation study of ITF.

Table 2: Short-Horizon Navigation Task and Backtracking Instruction Task Results.

Method	Short-Horizon Navigation Task					Backtracking Instruction Task		
	SR↑	OSR↑	SPL↑	NE↓	nDTW↑	SR↑	SPL↑	TPS@45°↑
NavFoM	51.21	76.75	43.24	4.96	55.97	32.49	22.62	9.29
SPAN-Nav	48.25	83.16	39.01	5.70	39.74	50.88	29.90	36.06
BE-VLN (ours)	**81.01**	**84.11**	**76.43**	**2.00**	**80.22**	**63.38**	**38.54**	**77.23**

TPS@45° denotes Terminal-Pose Success, defined as the proportion of successful episodes with a final heading error within 45° of the direction toward the initial position.

Short-Horizon Navigation And Language-Conditioned Backtracking. We test the same three VLN models on both short-horizon navigation and language-conditioned backtracking test sets. For implementation details of the test set, see the Appendix B.4.

As shown in Table 2, BE-VLN achieves the strongest overall performance on both tasks among the three models, the two evaluations show that BE-VLN supports both forward execution of dynamically generated short-horizon instructions and language-conditioned return to previously visited locations.

Instruction-Termination Feedback Assessment To isolate the effect of explicit termination assessment, we compare *BE-VLN w/o ITF* and *BE-VLN* using the same trained BE-VLN checkpoint.

Table 3 shows that ITF consistently improves final navigation accuracy and execution efficiency on both R2R-CE and RxR-CE. Compared with BE-VLN w/o ITF, BE-VLN achieves lower NE, higher SR and SPL, and substantially shorter trajectories with fewer executed actions. On RxR-CE, ITF also leads to a clear improvement in nDTW. This result shows that explicit ITF provides a more reliable stopping criterion for independently executed instructions than inferring termination from small predicted motion alone. In the complete framework, ITF is further used as a feedback signal for supervisory verification rather than directly terminating the overall navigation episode.

Table 3: Full-route ITF ablation of BE-VLN on R2R-CE and RxR-CE.

Termination criterion	R2R Val-Unseen						RxR Val-Unseen						
	NE↓	OSR↑	SR↑	SPL↑	Path	Steps	NE↓	OSR↑	SR↑	SPL↑	nDTW↑	Path	Steps
BE-VLN w/o ITF	3.96	**77.76**	66.83	58.19	13.58	90.13	4.35	**81.11**	67.43	52.54	54.21	16.86	118.69
BE-VLN	**3.58**	73.57	**68.19**	**62.35**	**10.88**	**73.20**	**4.11**	74.52	**70.35**	**61.56**	**68.60**	**15.18**	**102.42**

Path is the mean traveled distance in meters and **Steps** is the mean number of executed actions.

5 CONCLUSION

We introduced ReDual-VLN, a dual-system framework for recoverable long-horizon VLN. A VLM-based Route Supervisor performs route planning, progress tracking, and recovery reasoning, with HCM organizing semantic plans, execution history, and recovery anchors. When a deviation occurs, the supervisor directs BE-VLN to return to a valid anchor on the intended route and resume the remaining instruction after verifying the return. To train BE-VLN for this interaction, we construct ReTrace-RxR with paired short-horizon navigation and backtracking demonstrations and instruction-specific termination boundaries. ReDual-VLN achieves success rates of 75.8% on R2R-CE and 78.5% on RxR-CE, demonstrating the effectiveness of recovery-aware dual-system navigation.

AI USE STATEMENT

Generative AI tools were used to assist with English translation and language editing, manuscript organization. They were not used to design or conduct experiments or to generate experimental data. The authors reviewed and verified all AI-assisted content and assume full responsibility for the final content of this work.

REFERENCES

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

An-Chieh Cheng, Yandong Ji, Zhaojing Yang, Zaitian Gongye, Xueyan Zou, Jan Kautz, Erdem Biyik, Hongxu Yin, Sifei Liu, and Xiaolong Wang. Navila: Legged robot vision-language-action model for navigation. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.018.

Zedong Chu, Shichao Xie, Xiaolong Wu, Yanfen Shen, Minghua Luo, Zhengbo Wang, Fei Liu, Xiaoxu Leng, Junjun Hu, Mingyang Yin, Jia Lu, Yingnan Guo, Kai Yang, Jiawei Han, Xu Chen, Yanqing Zhu, Yuxiang Zhao, Xin Liu, Yirong Yang, Ye He, Jiahang Wang, Yang Cai, Tianlin Zhang, Li Gao, Liu Liu, Mingchao Sun, Fan Jiang, Chiyu Wang, Zhicheng Liu, Hongyu Pan, Honglin Han, Zhining Gu, Kuan Yang, Jianfang Zhang, Di Jing, Zihao Guan, Wei Guo, Guoqing Liu, Di Yang, Xiangpo Yang, Menglin Yang, Hongguang Xing, Weiguo Li, and Mu Xu. Abot-n0: Technical report on the vla foundation model for versatile embodied navigation, 2026.

Yucheng Deng, Pingrui Lai, Xinhai Li, Chenjia Bai, Xiaoheng Deng, Chengnuo Sun, Xuelong Li, and Hua Yang. SpaceVLN: A zero-shot vision-and-language navigation agent with online spatial cognitive memory and reasoning, 2026.

Hongyu Ding, Sizhuo Zhang, Ziming Xu, Jinwen Guo, Hongxiu Liu, Xingzhi Cheng, Zixuan Chen, Haifei Qi, Duo Wang, Hao Xu, Jieqi Shi, Yifan Zhang, Jing Huo, Jian Cheng, Yang Gao, and Jiebo Luo. Uni-LaViRA: Language-vision-robot actions translation for unified embodied navigation, 2026.

Hongyan Feng, Sunlai Chen, Xuanyu Liu, Miao Pan, Yangfan Xie, Yuxiang Cui, Zhongxiang Zhou, Rong Xiong, Wenqi Zhang, Jianwei Yin, Yueting Zhuang, and Xuhong Zhang. Embodied-navigator: Point, think, memorize, and align for efficient navigation, 2026. URL `https://arxiv.org/abs/2608.17512`. TAMP-Nav.

Ruiyan Gong, Yingnan Guo, Junjun Hu, Jintao Kong, Xiaoxu Leng, Tianlun Li, Weize Li, Fei Liu, Zhicheng Liu, Jia Lu, Minghua Luo, Chenlin Ming, Yanfen Shen, Jiyue Tao, Zhengbo Wang, Mingyang Yin, Minqi Gu, Zihao Guan, Wei Guo, Guoqing Liu, Huachong Pang, Menglin Yang, Zeqian Ye, Xiaoxiao Geng, Zhining Gu, Honglin Han, Di Jing, Hongyu Pan, Mingchao Sun, Kuan Yang, Jianfang Zhang, Yanghong Chen, Ye He, Wei Mei, Jiahao Shi, Xiangpo Yang, Yanqing Zhu, Zedong Chu, Xiaolong Wu, and Mu Xu. Abot-n1: Toward a general visual language navigation foundation model, 2026.

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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4065–4075, 2026.

Gang He, Zhenyang Liu, Kepeng Xu, Li Xu, Tong Qiao, Wenxin Yu, Chang Wu, and Weiying Xie. Nipping the drift in the bud: Retrospective rectification for robust vision-language navigation, 2026.

Keji He, Yan Huang, Qi Wu, Jianhua Yang, Dong An, Shuanglin Sima, and Liang Wang. Landmark-RxR: Solving vision-and-language navigation with fine-grained alignment supervision. In *Advances in Neural Information Processing Systems*, volume 34, pp. 652–663, 2021.

Yicong Hong, Cristian Rodriguez, Qi Wu, and Stephen Gould. Sub-instruction aware vision-and-language navigation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pp. 3360–3376, 2020. doi: 10.18653/v1/2020.emnlp-main.271.

Jingzhi Huang, Junkai Huang, Wenxuan Song, Haoyang Yang, Hailong Huang, Haoang Li, and Yi Wang. Sedualvln: A spatially-enhanced dual-system for vision-language navigation, 2026a.

Xiangyun Huang, Xiangchen Wang, Runfeng Lin, Yihao Xu, Kangyu Huang, Hengchen Jiang, Xiwang Dong, and Jiarong Lin. From routes to steps: Separating semantic progress from local execution in vision-and-language navigation, 2026b.

Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *Proceedings of the European Conference on Computer Vision*, 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*, pp. 4392–4412, 2020. doi: 10.18653/v1/2020.emnlp-main.356.

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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15243–15252, 2026.

An Liu, Bingxi Liu, Hongyu Ding, Yixuan Jiang, Yaran Chen, Fulin Tang, Cong Leng, Hong Zhang, and Jian Cheng. Ham-vln: Harnessing hierarchical agentic memory for zero-shot vision-and-language navigation, 2026a.

Fei Liu, Shichao Xie, Minghua Luo, Zedong Chu, Junjun Hu, Xiaolong Wu, and Mu Xu. NavForesee: A unified vision-language world model for hierarchical planning and dual-horizon navigation prediction, 2026b. URL https://arxiv.org/abs/2512.01550. CVPR 2026.

Jiahang Liu, Tianyu Xu, Jiawei Chen, Lu Yue, Jiazhao Zhang, Zhiyong Wang, Minghan Li, Qisheng Zhao, Anqi Li, Qi Su, Zhizheng Zhang, and He Wang. Span-nav: Generalized spatial awareness for versatile vision-language navigation, 2026c.

Kailin Lyu, Kangyi Wu, Pengna Li, Xiuyu Hu, Qingyi Si, Cui Miao, Ning Yang, Zihang Wang, Long Xiao, Lianyu Hu, Jingyuan Sun, and Ce Hao. HiMemVLN: Enhancing reliability of open-source zero-shot vision-and-language navigation with hierarchical memory system, 2026.

Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. Self-monitoring navigation agent via auxiliary progress estimation. In *International Conference on Learning Representations*, 2019.

Tianyi Ma, Yue Zhang, Zehao Wang, and Parisa Kordjamshidi. Breaking down and building up: Mixture of skill-based vision-and-language navigation agents. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, pp. 13035–13065, 2026. doi: 10.18653/v1/2026.acl-long.595. URL https://aclanthology.org/2026.acl-long.595/.

Ke Rui, Yushen Zuo, Jiawei Wang, Haoran Jia, Jinming Ma, Weitao Zhou, and Minglei Li. Diagnosing semantic handoff failures in agent-orchestrated vision-language-action skill composition, 2026. URL https://arxiv.org/abs/2607.06256. RSS SemRob Workshop / arXiv preprint.

Xiangyu Shi, Zerui Li, Wenqi Lyu, Jiatong Xia, Feras Dayoub, Yanyuan Qiao, and Qi Wu. SmartWay: Enhanced waypoint prediction and backtracking for zero-shot vision-and-language navigation, 2025. URL https://arxiv.org/abs/2503.10069. IROS 2025.

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

Jiahang Wang, Yirong Yang, Yanqing Zhu, Minghua Luo, Shichao Xie, Fei Liu, and Mu Xu. ReflectVLN: Training vision-language navigation agents with reflective reasoning, 2026a.

Shaoan Wang et al. Lightnav-0: Eliciting vlm spatial intelligence for generalist embodied navigation, 2026b.

Meng Wei, Chenyang Wan, Jiaqi Peng, Xiqian Yu, Yuqiang Yang, Delin Feng, Wenzhe Cai, Chenming Zhu, Tai Wang, Jiangmiao Pang, and Xihui Liu. Ground slow, move fast: A dual-system foundation model for generalizable vision-and-language navigation. In *International Conference on Learning Representations*, 2026a.

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. In *IEEE International Conference on Robotics and Automation*, 2026b.

Zihao Xin, Wentong Li, Yixuan Jiang, Ziyuan Huang, Bin Wang, Piji Li, Jianke Zhu, Jie Qin, and Shengjun Huang. AgentVLN: Towards agentic vision-and-language navigation, 2026a.

Zihao Xin, Wentong Li, Yixuan Jiang, Bin Wang, Runmin Cong, Jie Qin, and Shengjun Huang. DecoVLN: Decoupling observation, reasoning, and correction for vision-and-language navigation, 2026b. CVPR 2026.

Xinda Xue et al. Omninav: A unified framework for prospective exploration and visual-language navigation. In *International Conference on Learning Representations*, 2026.

Hang Yin, Haoyu Wei, Xiuwei Xu, Wenxuan Guo, Jie Zhou, and Jiwen Lu. GC-VLN: Instruction as graph constraints for training-free vision-and-language navigation. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 1809–1824. PMLR, 2025. URL `https://proceedings.mlr.press/v305/yin25a.html`.

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. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(22):18737–18745, 2026. doi: 10.1609/aaai.v40i22.38942.

Shuang Zeng et al. Janusvln: Decoupling semantics and spatiality with dual implicit memory for vision-language navigation. In *International Conference on Learning Representations*, 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. In *Robotics: Science and Systems*, 2025a.

Jiazhao Zhang, Gengze Zhou, Hale Yin, Yiyang Huang, Zixing Lei, Qihang Peng, Haoqi Yuan, Jie Zhang, Xudong Guo, Xiaoyue Chen, An Yang, Fei Huang, Zhibo Yang, Junyang Lin, Dayiheng Liu, Jingren Zhou, Zhuoyuan Yu, Jingyang Fan, Zhixuan Liang, Pei Lin, Ye Wang, Haoyang Li, Anzhe Chen, Kun Yan, Xiao Xu, Jiahao Li, Lulu Hu, Minying Zhang, Shurui Li, Wenhu Xiao, Shuai Bai, Xuancheng Ren, Chenxu Lv, Chenfei Wu, and Xiong-Hui Chen. Qwen-RobotNav technical report: A scalable navigation model designed for an agentic navigation system, 2026a. Technical report.

Jiazhao Zhang et al. Embodied navigation foundation model. In *International Conference on Learning Representations*, 2026b.

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)*, 2025b. doi: 10.18653/v1/2025.acl-long.638.

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, 2026c. URL https://arxiv.org/abs/2606.30367.

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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5511–5522, 2025c.

Duo Zheng, Shijia Huang, Yanyang Li, and Liwei Wang. Efficient-VLN: A simple yet strong baseline for efficient vision-language navigation, 2026a.

Wanrong Zheng, Yunhao Ge, and Laurent Itti. Three-step nav: A hierarchical global–local planner for zero-shot vision-and-language navigation. In *Proceedings of The 29th International Conference on Artificial Intelligence and Statistics*, volume 300 of *Proceedings of Machine Learning Research*, pp. 4645–4653. PMLR, 2026b. URL https://proceedings.mlr.press/v300/zheng26b.html.

A METHOD AND TRAINING DETAILS

This appendix provides the supervisory state conventions, ReTrace-RxR construction procedures, and BE-VLN architecture and supervision details accompanying Section 3.

A.1 SUPERVISORY STATE AND REVIEW PROTOCOL

Review timing and instruction state. The two decision timescales refer to how often decisions are made, not a fixed ratio of model inference times. BE-VLN continually updates its trajectory and ITF predictions as observations arrive, while its instruction remains unchanged between supervisory decisions. Both ITF-triggered and periodic reviews use Eq. equation 2. Periodic review can assess ongoing execution without a termination proposal. A review can retain the instruction, and multiple reviews may occur within one instruction. When a new instruction is dispatched, ITF is evaluated under that command rather than inherited from its predecessor. Observations are recorded throughout execution for subsequent review.

Progress transitions. CONTINUE preserves both ℓ_j and k_j while permitting execution-history updates. ADVANCE has two distinct uses. After verified completion of a route objective, it dispatches the next objective's instruction. After a verified return, it resumes the pending objective from the recovered state without automatically incrementing k_j. RECOVER preserves the unfinished objective while replacing its current execution command with a backtracking instruction. STOP produces $\ell_{j+1} = \emptyset$ and is authorized only after the final instruction and the original route's completion conditions have been verified. An incomplete return that remains consistent with its instruction is continued rather than interpreted as route progress.

Anchor records and validity. During reviews, the supervisor examines recorded observations to identify salient visited locations. Each anchor associates a language description with supporting visual observations, visit order, and corresponding route progress. Seeing a landmark does not establish that the agent has visited its associated location, and a location visited on an incorrect branch is not thereby a valid route anchor. Selection prioritizes a nearby, unambiguous intermediate anchor supported by the intended route and suitable for resuming the pending objective. Verified instruction-start boundaries provide fallback targets, with the initial observation supplying the first route anchor. This preference avoids unnecessary retracing but does not assume that the most recent location is valid or that its original image remains within BE-VLN's local window.

The recovery procedure uses these records to select a target and formulate the return and resumption instructions. HCM maintains the shared route state and incorporates verified outcomes. The two components therefore share execution evidence while serving distinct roles, without introducing another navigation policy.

A.2 RETRACE-RXR CONSTRUCTION

A.2.1 SHARED-BOUNDARY NAVIGATION

We map human annotations from Landmark-RxR (He et al., 2021) onto collected RxR-CE rollouts. These annotations align short instructions with corresponding subpaths.

Consider a rollout with states $s_0, \ldots, s_N$, observations o_t, and actions $a_t : s_t \to s_{t+1}$. Temporally ordered spatial anchors define segment boundaries. When a proposed boundary falls within a maximal sequence of same-direction turns, language and heading evidence assign the entire turning sequence to one neighboring segment. We then repair boundary order and merge degenerate adjacent units. This prevents a single turning maneuver from being divided across independently issued instructions.

For ordered boundaries $0 = b_0 < b_1 < \cdots < b_M = N$, the forward example $F_i = (\ell_i^F, \tau_i^F)$ contains a short instruction and its corresponding execution,

$$\tau_i^F = \left(o_{b_{i-1}:b_i},\, [a_{b_{i-1}}, \ldots, a_{b_i-1}, \mathrm{STOP}_{\mathrm{ann}}]\right). \tag{4}$$

Both boundary observations are retained. The symbol $\mathrm{STOP}_{\mathrm{ann}}$ denotes a stationary terminal annotation in the demonstration, not a STOP token predicted by BE-VLN. The following segment begins at the same physical state and retains the outgoing source action. Thus, the same boundary

observation can have a stationary target under the preceding instruction and a movement target under the next. Each example's ITF labels are computed relative to its own endpoint using the rule in Appendix A.4.2.

A.2.2 Anchor-Grounded Backtracking

Return trajectory synthesis. For each eligible forward segment, the return begins at the realized endpoint rather than an idealized goal state. The agent first turns around, retraces the source route, and restores an orientation suitable for subsequent forward execution. Retracing reverses route traversal, exchanges left and right turns, and retains forward translations in the agent frame. The source segment's initial in-place turning prefix is excluded from retracing. Accordingly, the return target uses the segment's starting position and the heading of its first forward translation, rather than necessarily reproducing its original pre-turn pose.

Rendering and validation. We execute the return sequence in the simulator and render an egocentric observation before every action. The resulting views depict the route as encountered during return and are not a reversal of the forward image sequence. Demonstrations undergo geometric and execution-consistency checks covering poses, action phases, observation–action correspondence, and trajectory completeness. Only valid returns are retained.

Instruction generation and pairing. A multimodal model receives the source instruction and temporally ordered views of the forward and return executions. Return views ground the retracing directions and landmark descriptions, while forward views establish correspondence with the source route. The generated instruction ℓ_i^B and return demonstration τ_i^B form $B_i = (\ell_i^B, \tau_i^B)$, paired with F_i. Each direction has its own instruction and terminal boundary. Forward examples are retained when no valid return can be produced, so pairing does not require all source segments to be reversible.

These training examples return from segment endpoints to starting anchors. At deployment, the recovery mechanism in Section 3.2.1 may instead interrupt an instruction mid-execution and select a visited intermediate anchor. The dataset supplies instruction-conditioned retracing behavior, while the supervisor determines the online recovery target and the remaining task.

A.3 BE-VLN Architecture and Visual History

A.3.1 Backbone and Prediction Heads

BE-VLN is initialized from general-purpose pretrained Qwen3-VL-8B rather than a navigation checkpoint. We adopt the visual encoder, cross-modal projector, and language backbone used by SPAN-Nav (Liu et al., 2026c), with a trajectory prediction head and a stop head. The model predicts motion directly from language and RGB observations, without an occupancy-prediction branch or spatial chain-of-thought conditioning.

Let E_j^L and E_t^V denote the instruction and visual representations. The backbone produces an instruction-conditioned action representation h_t, which two lightweight MLP heads map to normalized waypoints and the ITF logit,

$$\begin{aligned} h_t &= f_\theta(E_j^L, E_t^V), \qquad \widehat{\overline{\mathbf{W}}}_t = g_{\mathrm{traj}}(h_t), \\ q_t &= g_{\mathrm{stop}}(h_t), \qquad p_t^{\mathrm{ITF}} = \sigma(q_t). \end{aligned} \tag{5}$$

The trajectory head produces $H = 8$ waypoints, each with planar position and heading. Predictions are rescaled to physical units before execution. Full-route navigation, short-horizon navigation, and backtracking share both heads and the backbone. The stop head supplies instruction-relative feedback to the supervisor rather than replacing task-completion verification.

A.3.2 Observation Encoding and Causal History

At every step, the visual input consists of the current frame and up to 32 immediately preceding frames. Selection is deterministic by temporal recency, using all available preceding frames when fewer than 32 exist. Following the feature-compression procedure in SPAN-Nav (Liu et al., 2026c),

historical visual features are spatially average-pooled while current-frame features are retained intact. The historical features remain in temporal order, are concatenated with current-frame features, and are projected into the language backbone's representation space to form E_t^V.

The same causal observation rule applies during navigation training and execution across all navigation sources. The recent window may extend across an instruction switch, but older frames are not retained selectively. During backtracking, observations from the preceding forward traversal can support correspondence with a visited anchor when they remain within the window. Future observations from the return demonstration are never used as historical input to an earlier prediction step. The supervisor retains broader route evidence, but its complete visual history is not passed to BE-VLN.

Forward–backtracking pairing identifies the source segment that a return retraces. It does not override the history budget, imply access to the complete source trajectory, or introduce a pairwise consistency loss. All behaviors use the same observation encoder and prediction heads, without a separate backtracking policy or memory network. The model also omits occupancy reconstruction, occupancy-latent consistency, and occupancy-conditioned training stages.

A.4 Training Targets and Objectives

A.4.1 Trajectory Representation

For current planar position and heading $(\mathbf{v}_t, \theta_t)$ and future trajectory record $(\mathbf{v}_{t,h}, \theta_{t,h})$, the normalized target waypoint is

$$\overline{\mathbf{y}}_{t,h} = \begin{bmatrix} R(\theta_t)^\top(\mathbf{v}_{t,h} - \mathbf{v}_t) \\ \mathrm{wrap}(\theta_{t,h} - \theta_t) \end{bmatrix} \oslash \mathbf{s}. \tag{6}$$

Here, R is the planar rotation matrix, wrap maps heading differences to their principal interval, $\mathbf{s}$ contains componentwise scale factors, and $\oslash$ denotes componentwise division. All future records use the same current reference frame and begin with the action associated with the current pre-action observation. Sequences shorter than H are padded by repeating the terminal state. At inference, the componentwise scales are restored before trajectory execution.

A.4.2 Instruction-Relative ITF Supervision

All navigation samples receive a binary termination label relative to their assigned instruction. The relevant terminal boundary is the destination for a full route, the segment endpoint for short-horizon navigation, or the return target for backtracking. Actions are serialized into groups of planar trajectory records, with two stationary records representing the terminal annotation. If n_t^{rem} counts the remaining records at the pre-action observation, including this terminal group, the label is

$$z_t = \mathbf{1}[n_t^{\mathrm{rem}} \le H], \qquad H = 8. \tag{7}$$

The count refers to remaining demonstration records rather than additional padding used to fill the prediction horizon.

In the RxR-CE representation used for ReTrace-RxR, a nominal $0.25\,\mathrm{m}$ forward action contributes two records. Reserving two of the eight records for the terminal annotation leaves at most $(8-2)\times 0.125 = 0.75\,\mathrm{m}$ of nominal remaining translation in the positive window, within the local evaluation's $3\,\mathrm{m}$ positional tolerance. This interpretation is specific to that discretization. The labeling rule itself is defined by trajectory-record count, not by a universal metric distance threshold.

ITF therefore supervises a demonstrated terminal window rather than frame-exact completion of every requested transition or spatial relation. During dual-system execution, the supervisor must still verify that the current observation satisfies the assigned instruction. The same labeling rule applies to full-route demonstrations and both ReTrace-RxR subsets, rather than only the newly constructed local data.

A.4.3 Navigation and Vision–Language Co-Training

The navigation corpus $\mathcal{D}_{\mathrm{nav}}$ combines full-route demonstrations, Shared-Boundary Navigation, and Anchor-Grounded Backtracking. Full routes supervise complete instruction following, while

ReTrace-RxR adds short-instruction execution and language-conditioned route retracing. Following SPAN-Nav (Liu et al., 2026c), we also use general question-answering examples involving images and videos, denoted by $\mathcal{D}_{\mathrm{qa}}$. These examples train the backbone through its original language-modeling output.

The losses in Eq. equation 3 are

$$\begin{aligned}\mathcal{L}_{\mathrm{traj}} &= \frac{1}{3H}\sum_{h=1}^{H}\left\|\widehat{\overline{\mathbf{y}}}_{t,h}-\overline{\mathbf{y}}_{t,h}\right\|_2^2,\\ \mathcal{L}_{\mathrm{ITF}} &= \mathrm{BCEWithLogits}(q_t, z_t),\\ \mathcal{L}_{\mathrm{qa}} &= -\sum_{n=1}^{N_a}\log P_\theta(a_n \mid a_{<n}, X),\end{aligned} \tag{8}$$

where X denotes the visual input and question, and $a_1, \ldots, a_{N_a}$ are answer tokens. Every navigation sample contributes both $\mathcal{L}_{\mathrm{traj}}$ and $\mathcal{L}_{\mathrm{ITF}}$. QA examples contribute only $\mathcal{L}_{\mathrm{qa}}$ and receive no trajectory or ITF labels. The coefficients in Eq. equation 3 balance the three objectives.

Full-route, short-horizon, and backtracking examples differ in their instructions, available causal history, demonstrated motion, and terminal boundaries, but not in their prediction heads or navigation losses. ReTrace-RxR thereby extends a shared executor's motion and termination supervision to the local commands used by the dual-system framework.

B Experimental Setup and Comparison Controls

This appendix provides the shared executor configuration, supervisory execution protocol, and controlled settings for the comparisons in Sec. 4. The baseline models and ablation configurations are defined in Sec. 4.1.

B.1 Executor Configuration

BE-VLN uses four-view RGB observations covering the front, right, back, and left directions. Its input comprises the active instruction, the current frame, and up to 32 immediately preceding frames; it predicts eight future waypoints and an Instruction-Termination Feedback (ITF) score. The architecture and training follow Sec. 3 and are not modified across the experiments. All evaluations of BE-VLN, including its use in the complete system, share the same checkpoint. This local history window can span an instruction switch but does not provide arbitrary earlier frames from the Supervisor's global history.

B.2 High-Level Review and Execution

In the hierarchical configurations, the Route Supervisor reviews execution both when ITF triggers a review and at periodic review points, following Sec. 3. It uses route-level execution history to maintain progress and select subsequent instructions. In complete ReDual-VLN, the available decisions are CONTINUE, ADVANCE, RECOVER, and STOP; only a verified STOP decision terminates the episode. In standalone full-route and local capability evaluations, the Route Supervisor is not invoked, and the evaluation interface applies the selected termination criterion directly to the instruction under test. Controlled comparisons retain the same episode-level environment-action budget and execution settings unless the evaluated factor explicitly requires a change.

In the recovery-disabled configuration, the Route Supervisor can CONTINUE, ADVANCE, or STOP but cannot initiate a recovery process that returns to a historical anchor and then resumes the unfinished route. Complete ReDual-VLN additionally enables RECOVER: the Supervisor can select a previously visited anchor on the intended route, issue a language-conditioned backtracking instruction, verify the return, and resume the unfinished objective. The standalone BE-VLN reference uses its ITF criterion to end the full-route episode without high-level instruction decomposition, route-progress management, or recovery-command generation.

B.3 Controls for Ablation and Executor-Level Comparisons

HCM and recovery ablation. All three configurations in Sec. 4.1 share the BE-VLN checkpoint, four-view RGB interface, current-frame-plus-recent-32-frame input window, test episodes, and episode-level action budget. The two hierarchical configurations additionally share the Route Supervisor and review settings, differing in the availability of recovery. The trained BE-VLN is unchanged when RECOVER is disabled.

Independent local evaluations. The short-horizon and backtracking tests do not use high-level correction or completion review. In backtracking, BE-VLN retains the local history window described in Appendix B.1; it does not receive an additional target image, goal coordinates, a reversed control sequence, or future observations from the return trajectory. These input restrictions apply to the independent test, while the target and endpoint annotations follow the local evaluation protocol in Sec. 4.1.

Termination-criterion comparison. The small-motion and stop-head criteria in Sec. 4.1 are evaluated using the same trained BE-VLN checkpoint. The checkpoint, input configuration, trajectory predictor, action execution, test instructions, and budget are unchanged; only the inference-time termination criterion is switched. No Route Supervisor is used. This comparison does not retrain the model or remove the stop head.

Executor replacement. We retain the high-level model, route and context management, recovery procedure, test episodes, and environment-action budget, and replace only the lower-level navigator. Each executor uses its applicable observation, action, and completion-feedback interface, with any interface differences identified in Table 4. This comparison evaluates complete navigation under high-level supervision, unlike the independent capability tests.

B.4 Additional Analysis of BE-VLN Capabilities

B.4.1 Short-Horizon Navigation

The short-horizon evaluation directly examines execution at the instruction granularity used by the Route Supervisor rather than inferring local capability from full-route performance. Because each instruction defines an intermediate route segment with its own endpoint, successful execution requires BE-VLN to treat that segment as an independently executable task and terminate relative to its local objective. No high-level correction or completion review is available in this evaluation, so the performance gains in Table 2 can be attributed to the executor rather than supervisory replanning or verification. Moreover, the same BE-VLN checkpoint is used for both standalone full-route navigation and local execution, showing that these capabilities coexist within a shared policy.

B.4.2 Language-Conditioned Backtracking

Language-conditioned backtracking differs from ordinary local navigation because the instruction refers to a previously visited location and requires execution in the return direction of the traversed route. BE-VLN generates this behavior using the same policy as forward navigation, without a separate backtracking controller. The results in Table 2 therefore establish an executor capability distinct from navigating toward a new local destination. This independent evaluation isolates the execution of the return instruction, whereas the system-level ablation in Sec. 4.3 measures whether backtracking, return verification, and subsequent task resumption improve completion of the original route. The two evaluations thus connect local route-retracing capability with end-to-end recovery.

B.5 Instruction-Termination Assessment

We evaluate whether explicit Instruction-Termination Feedback (ITF) provides a more effective stopping criterion than inferring termination from small predicted motion. Following the controlled setting described in Appendix B.3, both criteria use the same trained BE-VLN checkpoint, with only the inference-time termination decision changed.

Table 3 shows that ITF improves SR, SPL, and NE while reducing average path length and action steps on both R2R-CE and RxR-CE. It also substantially improves nDTW on RxR-CE. Since the navigation weights and trajectory predictor remain unchanged, these gains reflect the learned

Table 4: Navigation-executor comparison under a fixed Route Supervisor on RxR-CE.

Executor	SR↑	OSR↑	SPL↑	NE↓	nDTW↑
NavFoM	65.00	72.00	34.27	5.35	28.64
SPAN-Nav	70.00	76.00	53.67	4.35	47.38
BE-VLN (ours)	**79.00**	**82.00**	**65.92**	**3.32**	**70.11**

All executors are evaluated on RxR Val-Unseen while the Route Supervisor and high-level procedures are held fixed. BE-VLN (ours) is the complete ReDual-VLN configuration. The NavFoM result is still pending; TBD denotes a pending result.

termination output rather than additional training or a stronger action predictor. In the complete ReDual-VLN framework, ITF serves as feedback for supervisory verification rather than directly terminating the episode.

B.6 Executor Comparison under a Fixed Route Supervisor

We further examine whether the local capabilities of BE-VLN translate to complete-system navigation by replacing the lower-level navigator while keeping the Route Supervisor, HCM, recovery procedure, evaluation episodes, and action budget fixed. Unlike the independent capability evaluations in Tables 2 and 2, this experiment evaluates full-route navigation with VLM supervision. Detailed controls and interface considerations are provided in Appendix B.3.

As shown in Table 4, using BE-VLN as the executor outperforms SPAN-Nav across all reported metrics under the same high-level system. This result connects the independently evaluated short-horizon navigation, backtracking, and termination capabilities of BE-VLN to their use in complete-system navigation. Because each navigator retains its applicable action and completion-feedback interface, the comparison evaluates the resulting executor configuration as a whole rather than an isolated model component.