WHAT SHOULD A NAVIGATOR REMEMBER? SPATIOTEMPORAL MEMORY FOR ZERO-SHOT VISION-LANGUAGE NAVIGATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Zero-shot vision-and-language navigation (VLN) agents built on large proprietary models generalize well but collapse when the backbone is replaced with an open 8B-scale model that can run on a robot's compute budget. Post-trained agents achieve high success in simulation but tend to follow memorized routes rather than the given instruction, and degrade in the real world. We argue that the gap is not perception but memory: existing agents present the model with an unorganized stream of past observations, from which a small model cannot extract the few facts each decision requires. We ask what a navigator should remember, and answer it through a semantic communication view in which a navigation decision is a query and memory is a set of facts that must entail it. Memory therefore need not reproduce the observation stream, only preserve what may establish future queries, and the agent should act to gather evidence when memory is insufficient. We instantiate this view as a spatiotemporal memory agent with task memory over ordered subgoals, hierarchically compressed temporal memory anchored to poses, and a geometric spatial map with a committed navigation target. The agent selects waypoint intents executed by a geometric controller and issues a preview action when the current view cannot resolve the decision. To evaluate whether an agent follows instructions rather than routes, we introduce three instruction-perturbation protocols. On R2R-CE val-unseen, our method achieves the best SR and SPL among zero-shot methods on 8B-scale models, consumes roughly $4\times$ fewer tokens per decision than the strongest zero-shot baseline, and transfers to a Unitree Go2 quadruped in real-world environments where a post-trained baseline fails.

1 INTRODUCTION

Vision-and-Language Navigation (VLN) requires an embodied agent to follow a natural language instruction through a previously unseen environment, and it is a prerequisite for higher-level tasks such as object manipulation and human–robot interaction. Post-training methods (Guo et al., 2026; Wei et al., 2026; Wang et al., 2026) fine-tune vision-language models on navigation trajectories and achieve strong success rates on R2R-CE (Krantz et al., 2020). Zero-shot methods (Chen et al., 2025; Shi et al., 2025; Yuan et al., 2025; Liu & Yu, 2025) instead rely on the reasoning ability of large pretrained models. Both directions, however, leave a gap between benchmark numbers and deployable behavior.

Post-training methods tend to memorize routes rather than follow instructions. When we swap *left* and *right* in the first turn command, a state-of-the-art post-trained agent drops from 91% to 43% in following the commanded turn (Table 5), and its success rate degrades markedly on a real quadruped robot (Table 4). Zero-shot methods avoid this overfitting but typically rely on large proprietary models. With an open 8B-scale backbone that fits a robot's compute budget, their performance collapses: HSGM (Liu & Yu, 2025) falls from 0.463 to 0.049 SR (Table 1). The failure lies not in perception but in structure: these agents feed the model a long, unorganized stream of observations, from which a small model cannot extract the few facts each decision requires.

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.

We therefore ask how past observations should be organized so that each decision can be justified from a compact memory. Following the semantic communication framework of Lastras et al. (2026), we treat a navigation decision as a query and memory as a set of facts that must entail it. Memory then need not reproduce the observation stream, only preserve what may establish future queries; when it is insufficient, the agent must act to gather more evidence.

Guided by this view, we propose a spatiotemporal memory agent that runs on Qwen3-VL-8B from a single egocentric RGB-D view. It maintains task memory over ordered subgoals, hierarchically compressed temporal memory anchored to poses, and a geometric spatial map with a committed navigation target. The policy selects waypoint intents executed by a geometric controller and issues a `PREVIEW` action when the current view cannot resolve a decision. To test whether agents follow instructions rather than routes, we further introduce three instruction-perturbation protocols.

Our contributions are as follows.

- We formulate zero-shot VLN as memory-conditioned decision making under a semantic communication view, motivating a memory that preserves what future decisions require and treating active perception as information gathering.
- We instantiate this view as a spatiotemporal memory agent with task, temporal, and spatial memory, an intent-level action space, and a preview mechanism, running on an 8B open model from a monocular view.
- We propose the FLIP, GOAL-ONLY, and PARAPHRASE protocols and the RSR metric, revealing that a strong post-trained baseline follows memorized routes rather than the given instruction.

2 Related Work

2.1 Vision Language Navigation

Vision-and-Language Navigation (VLN) is a fundamental capability for embodied AI, as it endows robots with autonomous mobility by integrating natural language understanding and visual perception. The ability to accurately follow language instructions and reach designated target locations serves as a prerequisite for performing higher-level tasks, such as object manipulation and human–robot interaction.

The VLN task was first formally introduced by R2R Anderson et al. (2018), which establishes an indoor benchmark featuring a discrete navigation graph. To bridge the gap between discrete navigation and real-world robot movement, R2R-CE Krantz et al. (2020) extends VLN to continuous environments, enabling agents to execute low-level control actions. REVERIE Qi et al. (2020) further expands the VLN paradigm to an object-oriented setting, requiring agents to simultaneously navigate and identify target objects grounded in language instructions.

Despite progress achieved by these benchmarks, most pretrained VLN models are optimized for task-specific datasets, resulting in overfitting and limited generalization to unseen environments, alongside poor instruction understanding. ETPNav An et al. (2024) addresses this challenge by constructing online topological maps and planning obstacle-avoiding trajectories based on learned map representations. BEVBERT An et al. (2022) builds upon this work by introducing a hybrid perception architecture that integrates global topological maps with local metric maps, enabling both effective long-term planning and fine-grained short-term navigation.

2.2 Agentic Vision Language Navigation

Agentic VLN primarily leverages agent systems that enable VLMs to perform navigation tasks. These approaches emphasize task generalization, and memory is a central component of the agent. Agentic System for VLN Xin et al. (2026); Huang et al. (2026a;b) introduces an agent-based framework that utilizes a skill library and other system modules for environment perception, localization, and navigation planning. FreeAskWorld Peng et al. (2026) advances VLN task modeling by incorporating dynamic environments and long-horizon tasks, reflecting real-world scenarios. However, existing methods An et al. (2024; 2022) on this benchmark achieve only 0% success rate, indicating that current samll expert models lack the generalization capability to understand complex

environments. To address this limitation, recent work explores leveraging large language models' reasoning abilities. AgenticNav Li et al. (2026) adopts techniques similar to those in OpenClaw, however, it lacks explicit memory modeling mechanisms. CA-Nav Chen et al. (2025) models spatial memory as a value map that scores candidate waypoints by their relevance to the instruction. Whereas this memory directly encodes task-conditioned values. BSC-Nav Ruan et al. (2026) adopts brain-inspired spatial memory for the vision-language navigation task.

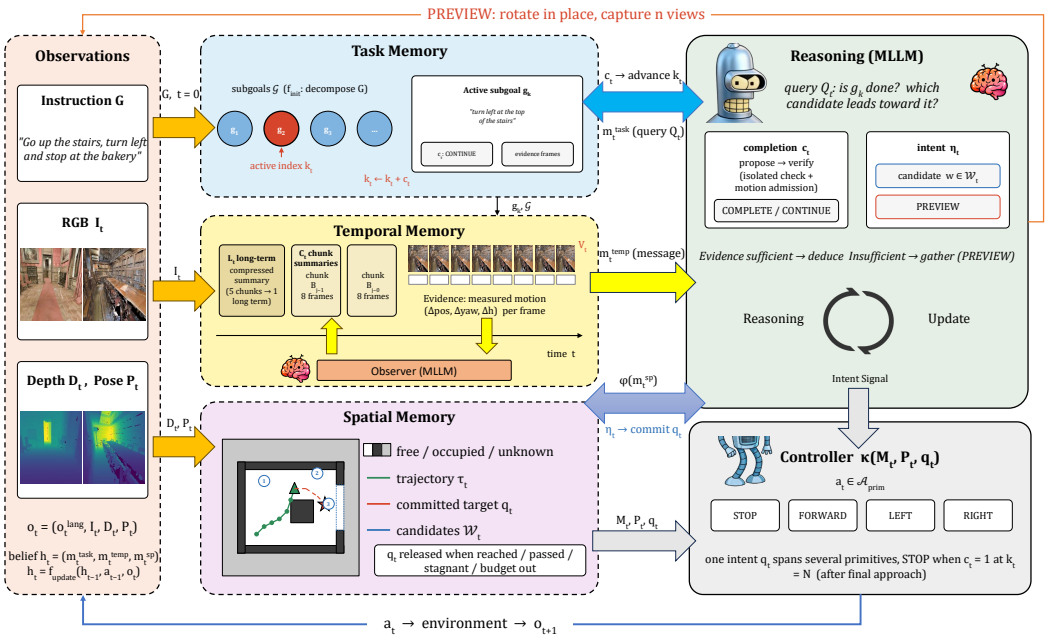


Figure 1: The overview of our method

3 Methodology

3.1 Problem Formulation

We formulate vision-language navigation as a partially observable Markov decision process (POMDP) $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{T}, R, \Omega, O, \gamma \rangle$. A state $s_t \in \mathcal{S}$ comprises the 3D environment and the agent pose P_t, which is not directly observable. The action space $\mathcal{A} = \mathcal{A}_{\text{prim}} = \{\texttt{FORWARD}, \texttt{LEFT}, \texttt{RIGHT}, \texttt{STOP}\}$ consists of low-level motion primitives, and the transition function $\mathcal{T}(s_{t+1} \mid s_t, a_t)$ updates the agent pose according to the executed action. The reward R is given when the agent issues `STOP` within a success radius of the goal, with discount factor $\gamma \in (0, 1]$; since our method does not optimize R directly, it serves only to define task success.

The observation space is $\Omega = \widetilde{\Omega}_{\text{lang}} \times \Omega_{\text{vis}}$ with $\widetilde{\Omega}_{\text{lang}} = \Omega_{\text{lang}} \cup \{\varnothing\}$, and the observation function $O(o_t \mid s_t)$ renders what the agent perceives from its current state. The agent receives the language instruction G only once at the beginning of an episode, while visual observations v_t are acquired at every step:

$$o_t = (o_t^{\text{lang}}, v_t), \qquad o_t^{\text{lang}} = \begin{cases} G, & t = 0, \\ \varnothing, & t > 0. \end{cases} \tag{1}$$

Since s_t is hidden, the agent maintains a belief representation h_t that integrates the instruction with the interaction history, $h_0 = f_{\text{init}}(G, v_0)$ and $h_t = f_{\text{update}}(h_{t-1}, a_{t-1}, v_t)$, and acts according to $a_t \sim \pi(\cdot \mid h_t)$. The instruction is thus an initial observation rather than a repeated input, although its encoded representation is retained throughout navigation.

Section 3.2 characterizes which information in h_t each decision requires.

3.2 A Logical View of Navigation

We adopt the semantic communication framework of Lastras et al. (2026) to characterize what information each navigation decision requires. Let $\mathcal{Z}$ be a finite set of hypotheses about the scene, such as where landmarks lie and which directions lead toward each subgoal. For a set of facts F, its kernel $\mathrm{Ker}(F) \subseteq \mathcal{Z}$ contains the hypotheses consistent with F. Conjunction intersects kernels, $\mathrm{Ker}(F \wedge F') = \mathrm{Ker}(F) \cap \mathrm{Ker}(F')$, and F entails a query Q if and only if $\mathrm{Ker}(F) \subseteq \mathrm{Ker}(Q)$.

At step t, temporal memory m_t^{temp} acts as a message summarizing the observation stream for the reasoning module, which additionally holds knowledge R_t from task and spatial memory. Each decision corresponds to a query Q_t: whether the active subgoal is completed, answered by c_t, or whether a candidate waypoint leads toward it, answered by η_t. A decision is justified when

$$\mathrm{Ker}\big(m_t^{\mathrm{temp}} \wedge R_t\big) \subseteq \mathrm{Ker}(Q_t). \tag{2}$$

The agent either derives a decision already entailed by its memory or acquires additional evidence when the current memory is insufficient. Reasoning derives conclusions implicit in memory, so that a compact memory can support many decisions Lastras et al. (2026). memory therefore need not reproduce the observations, only preserve what may establish future queries. Reasoning, however, cannot shrink the kernel; it only makes its consequences explicit. When the current facts are insufficient to resolve a query, the agent can act to acquire additional evidence. Navigation thus alternates between deduction, which extracts what current information implies, and information gathering, which extends what is known. Since our modules operate on natural language rather than a formal logic, this view serves as a design rationale rather than a quantitative bound.

3.3 Spatiotemporal Memory

We instantiate the latent belief representation with three complementary memory components:

$$h_t = \big(m_t^{\mathrm{task}}, m_t^{\mathrm{temp}}, m_t^{\mathrm{sp}}\big)\,, \tag{3}$$

which retain the navigation objective, observed temporal context, and accumulated spatial evidence, respectively. This structured representation approximates the information available in the interaction history rather than explicitly maintaining a posterior distribution over environment states.

Our implementation uses RGB observations I_t, aligned depth D_t, and camera poses P_t, with known camera intrinsics K. We therefore extend the visual observation in the preceding formulation to $v_t = (I_t, D_t, P_t)$, giving

$$h_0 = f_{\mathrm{init}}(G, v_0), \qquad h_t = f_{\mathrm{update}}(h_{t-1}, a_{t-1}, v_t). \tag{4}$$

The instruction remains an initial observation and is retained in memory throughout the episode.

Task memory stores the instruction, its ordered decomposition into subgoals, and the active subgoal index:

$$m_t^{\mathrm{task}} = (G, \mathcal{G}, k_t)\,, \qquad \mathcal{G} = (g_1, \ldots, g_N). \tag{5}$$

The active index k_t advances when the reasoning module determines that the current subgoal has been completed. This representation preserves route order and conditions navigation on the current stage of the instruction.

Temporal memory represents the observed trajectory at multiple resolutions:

$$m_t^{\mathrm{temp}} = (W_t, C_t, L_t, E_t)\,, \tag{6}$$

where W_t contains recent observations, C_t stores pending chunk-level summaries, L_t contains compressed long-term summaries, and E_t records measured execution history. Recent observations are associated with frame indices, the camera pose P_t, and motion evidence, including translation, rotation, and elevation change.

The short-term window retains the latest eight real observations. Every eight new observations form a chunk, whose visual content is summarized. Two frames are retained across chunk boundaries to preserve contextual continuity. Every five pending chunk summaries are compressed into a new long-term block and appended to the existing memory. Consequently, recent experience remains

visually detailed, while earlier experience is represented through compact descriptions of observed layouts, landmarks, and their relationships.

Temporal summaries are generated without access to the navigation instruction. This does not raise the cost of memory, when the sender conveys all it can infer, the communication limit is the same whether or not it knows the receiver's knowledge Lastras et al. (2026). A subsequent task-conditioned reasoning stage combines this observational memory with the instruction and execution feedback. Episode history is preserved across subgoal transitions, while completion evidence is scoped to the active subgoal.

Spatial memory maintains a geometric map, the observed agent trajectory, and a committed navigation target:

$$m_t^{\mathrm{sp}} = (M_t, \tau_t, q_t)\,. \tag{7}$$

Here, M_t is a ground-plane occupancy map, τ_t is the trajectory accumulated from observed poses, and q_t is the active world-coordinate target, or $\varnothing$ when no target is committed.

Depth observations are back-projected into world coordinates using the camera calibration:

$$\mathbf{x}_{t,u}^{w} = P_t\,\mathrm{BackProject}(u, D_t(u), K)\,, \tag{8}$$

where $\mathrm{BackProject}$ returns a homogeneous camera-coordinate point for pixel u. The ground-plane projections of these points update accumulated free-space and obstacle evidence:

$$M_t = \mathrm{Fuse}(M_{t-1}, D_t, P_t, K)\,. \tag{9}$$

Each cell is classified as free, occupied, or unknown. The resulting map supports obstacle-aware path planning and frontier exploration. The two memories are coupled through pose anchoring. Each temporal entry records the pose at which it was captured, and each summary records the trajectory segment it covers. Temporal memory retains observed landmarks and their semantic relationships, complementing the geometric map.

3.4 Memory-Conditioned Navigation Policy

Under the POMDP formulation, the agent never observes the true state s_t and must act on its belief h_t. Section 3.3 instantiates h_t as structured memory; this section describes how the agent maps this memory to primitive actions and how the memory evolves with each step. Rather than predicting primitives directly, we factorize the policy $\pi(a_t \mid h_t)$ into an intent policy π_θ that decides *where to go* and a geometric controller κ that decides *how to get there*:

$$a_t = \kappa(M_t, P_t, q_t), \qquad q_t \sim \pi_\theta(\cdot \mid h_t). \tag{10}$$

The intent policy π_θ is implemented by a reasoning module

$$(c_t, \eta_t) = \mathcal{J}_\theta\big(m_t^{\mathrm{task}}, m_t^{\mathrm{temp}}, m_t^{\mathrm{sp}}\big)\,, \tag{11}$$

whose two outputs address task progress and spatial intent. The completion flag $c_t \in \{0, 1\}$ indicates whether the active subgoal g_{k_t} has been achieved, and `STOP` is issued when $c_t = 1$ at $k_t = N$. The intent $\eta_t \in \mathcal{W}_t \cup \{\texttt{PREVIEW}\}$ either selects a target waypoint or requests a panoramic preview.

Because the agent perceives only a single egocentric view, the current memory may not establish which candidate leads toward the active subgoal, for instance when the referenced landmark lies outside the field of view. In this case, $\mathcal{J}_\theta$ selects `PREVIEW`, the information-gathering action of Section 3.2: the agent rotates in place to capture n views $\mathcal{V}_t = \{v_t^{(i)}\}_{i=1}^{n}$, which are added to temporal memory and extend the candidates to the full surroundings. Since added views can only narrow the hypothesis set,

$$\mathrm{Ker}\big(m_t^{\mathrm{temp}} \wedge \mathcal{V}_t \wedge R_t\big) \subseteq \mathrm{Ker}\big(m_t^{\mathrm{temp}} \wedge R_t\big)\,, \tag{12}$$

provided that the added views are correctly and consistently interpreted as facts, previewing may provide evidence relevant to the query but does not guarantee that Eq. equation 2 holds.

When a waypoint is predicted in image coordinates, Eq. equation 8 is used to obtain its world-coordinate location for storage as the committed target in spatial memory. The target persists across steps until it is released by the memory update below, so a single intent spans multiple primitive

steps. The controller κ plans an obstacle-aware path to q_t on the occupancy map M_t and emits the next primitive action $a_t \in \mathcal{A}_{\text{prim}}$.

After executing a_{t-1}, the agent receives v_t and updates its memory as $h_t = f_{\text{update}}(h_{t-1}, a_{t-1}, v_t)$. The update decomposes over the three memory components,

$$f_{\text{update}} = \left(f^{\text{task}}, f^{\text{temp}}, f^{\text{sp}}\right), \tag{13}$$

each of which consumes a different part of the input.

For task memory, the instruction G and subgoals $\mathcal{G}$ are fixed after initialization, where f_{init} decomposes G into $\mathcal{G}$ and sets $k_0 = 1$. Only the stage index changes, $k_t = \min(k_{t-1} + c_{t-1}, N)$, where c_{t-1} is the completion flag computed by $\mathcal{J}_\theta$ from h_{t-1}. Hence f^{task} depends on h_{t-1} alone.

For temporal memory, let δ_t denote the translation, rotation, and elevation change computed from P_{t-1} and P_t. The recent window and execution history are updated as

$$W_t = \text{Last}_8(W_{t-1} \cup \{(I_t, \delta_t, P_t)\}), \qquad E_t = E_{t-1} \cup \{(a_{t-1}, \delta_t)\}. \tag{14}$$

When eight new observations have accumulated since the last chunk, they form a chunk B_j, which also includes the last two frames of B_{j-1} for continuity, and its summary is appended together with the trajectory segment it covers, $C'_t = C_{t-1} \cup \{(\text{Sum}(B_j), \tau_{[s_j, e_j]})\}$, where $[s_j, e_j]$ is the step range of B_j; otherwise $C'_t = C_{t-1}$. Pending summaries are compressed into long-term memory once five have accumulated:

$$(C_t, L_t) = \begin{cases} (\varnothing,\ L_{t-1} \cup \{\text{Comp}(C'_t)\}), & |C'_t| = 5, \\ (C'_t,\ L_{t-1}), & \text{otherwise.} \end{cases} \tag{15}$$

The summaries are designed to retain observations relevant to later decisions while reducing the amount of history presented to the reasoning module.

For spatial memory, the map and trajectory integrate the new depth and pose, and the committed target is released once it is no longer valid:

$$M_t = \text{Fuse}(M_{t-1}, D_t, P_t, K), \qquad \tau_t = \tau_{t-1} \cup \{P_t\}, \tag{16}$$

$$q_t = \begin{cases} \varnothing, & \text{Done}(q_{t-1}), \\ q_{t-1}, & \text{otherwise,} \end{cases} \tag{17}$$

where $\text{Done}(\cdot)$ holds when the target is reached, passed, or invalidated by stagnation (detected from E_t) or by exceeding its execution budget.

Each step thus proceeds in three stages: f_{update} integrates the new observation into memory; if $q_t = \varnothing$, the intent policy commits a new target, and κ emits the next primitive action toward it. Following the design rationale in Section 3.2, the agent uses memory-based reasoning to select targets and acquires additional observations when the available evidence is insufficient. This perception–reasoning–action loop repeats until `STOP` is issued.

3.5 Action Space Ablation

We compare a **primitive action space** with an **intent action space**, both conditioned on the same observation o_t and memory representation h_t. The two differ only in action abstraction: primitive actions specify low-level motion commands, whereas intent actions specify spatial targets that a geometric controller executes.

In the primitive action space, the policy selects $a_t \sim \pi_\theta(\cdot \mid h_t)$ from $\mathcal{A}_{\text{prim}} = \{\texttt{FORWARD}, \texttt{LEFT}, \texttt{RIGHT}, \texttt{STOP}\}$, where `FORWARD` translates by a fixed distance and `LEFT`/`RIGHT` rotate by a fixed angle. Although this action space is small, reaching a spatial target requires a long sequence of such decisions, and the number of possible sequences grows as $|\mathcal{A}_{\text{prim}}|^T$ over T steps.

In the intent action space, the policy instead selects a waypoint $q_t \sim \pi_\theta(\cdot \mid h_t)$ from $\mathcal{A}_{\text{intent}} = \mathcal{W}_t \cup \{\texttt{STOP}\}$, where $\mathcal{W}_t \subseteq \mathbb{R}^2$ is either a set of K generated candidates or, for direct prediction, the local ground plane (image-space predictions are projected using depth and camera calibration). A geometric controller then produces the primitive actions $\mathbf{a}_t = \kappa(M_t, P_t, q_t)$, where M_t is the accumulated map and P_t the agent pose.

The key difference lies in the effective decision horizon. Per decision, $|\mathcal{A}_{\text{intent}}| = K+1$ can exceed $|\mathcal{A}_{\text{prim}}| = 4$. However, if a segment of T primitive steps is covered by N intents, with the i-th intent executing ℓ_i steps, then

$$\frac{H_{\text{intent}}}{H_{\text{prim}}} = \frac{N}{T} = \frac{1}{\bar{\ell}} < 1, \qquad \bar{\ell} = \frac{1}{N}\sum_{i=1}^{N} \ell_i > 1. \tag{18}$$

When an intent spans multiple primitive steps on average, intent prediction reduces the number of target-selection decisions required for a trajectory.

Consistent with this analysis, our method performs substantially better with waypoint prediction and selection than with primitive-action prediction. Model capacity also matters, HSGM processes 20 images per step. When replacing GPT-5 with Qwen-VL-8B, this leads to severe context overload. As shown in Table 3, HSGM consumes $4\times$ more tokens per decision than our method, causing its Success Rate (SR) to drop drastically from 0.479 to 0.049 (Table 1).

4 BENCHMARKS

We report standard VLN metrics on R2R-CE (Krantz et al., 2020): success rate (SR), success weighted by path length (SPL), navigation error (NE), oracle success rate (OSR), and normalized dynamic time warping (nDTW) (Ilharco et al., 2019). We further introduce the relative success rate $\text{RSR} = \text{SR}/\text{OSR}$, the fraction of goal-reaching episodes in which the agent correctly stops. RSR isolates stopping quality from exploration ability; in terms of Eq. equation 2, a low RSR means the agent reaches the goal but fails to establish the completion query for the final subgoal.

Beyond task success, we evaluate whether the agent's behavior is controllable. Inspired by Asimov's Three Laws of Robotics (Asimov, 2004), we consider three principles: avoiding unsafe behavior, following instructions, and preserving operational state; this study focuses on the second. We keep the scene and start pose fixed, perturb the instruction, and compare against the original instruction (ORIG):

- FLIP (82 episodes, balanced left/right) swaps *left* and *right* in the first turn command, which occurs at the start pose, to test instruction obedience.
- GOAL-ONLY (174 episodes) keeps only the final sub-instruction to test route execution; an agent that uses the route description should lose path fidelity (nDTW), not only SR.
- PARAPHRASE (200 episodes) rewrites each instruction without changing its meaning in four styles: same register (A1), terse (A2), colloquial (A3), and landmark re-description (A4).

5 EXPERIMENTS

5.1 SIMULATION EXPERIMENT

Our method uses JoyAI (Yao et al., 2026) as the observer, which summarizes visual observations into temporal memory, and Qwen3-VL-8B (Bai et al., 2025) as the reasoner. We compare against both post-training and zero-shot baselines. The zero-shot baselines differ in how they use the LLM; for example, CA-Nav uses it only for subgoal decomposition. For a fair comparison, we keep the original implementation of every reproduced baseline and only replace its MLLM with Qwen3-VL-8B. All experiments run on a server with 8 NVIDIA A100 GPUs.

5.2 REAL-WORLD EXPERIMENT

We deploy our method and AwareVLN on a Unitree Go2 quadruped robot in corridor and office environments. Both models run on the same A100 server, which communicates with the Go2 remotely: the robot streams its onboard observations to the server and executes the returned actions.

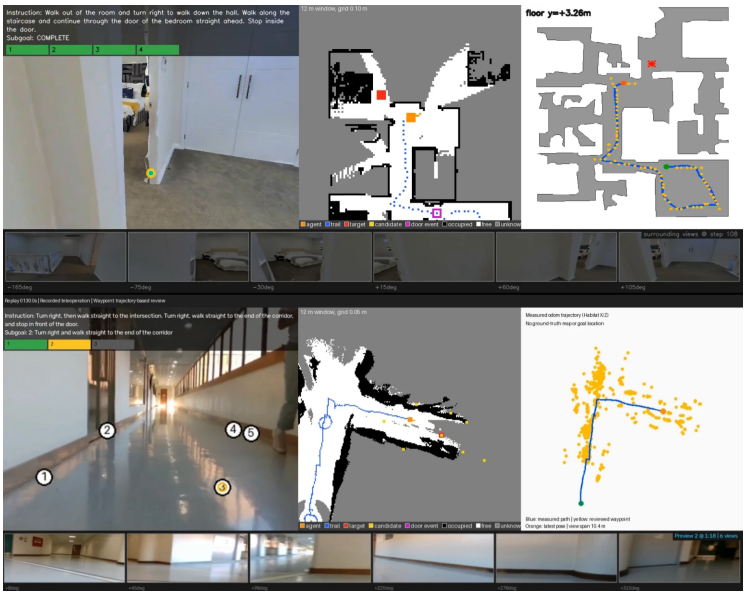


Figure 2: Qualitative results on R2R (top) and in real-world experiments (bottom).

5.3 Main Results

Among post-training methods, all approaches perform well in simulation. However, AwareVLN degrades markedly in real-world experiments (Section 4). Our FLIP experiment (Table 5) gives further evidence of overfitting. AwareVLN reaches 91% on the original instructions but only 43% on the flipped ones, which shows that it tends to follow the memorized original route even after the instruction has changed. In contrast, our method performs well in both settings, with 82% on the original instructions and 70% on the flipped ones. The paraphrase experiment (Table 7) leads to the same conclusion. AwareVLN degrades noticeably under the A4 paraphrased instructions, whereas our method remains stable when instructions are rephrased with semantically equivalent expressions. Among zero-shot methods built on 8B-scale models, our method achieves the best SR and SPL, and its RSR is substantially higher than that of the other methods. This indicates that our method can actually identify the target and make steady subgoal-level progress, rather than merely exploring the environment. When equipped with our memory module, CA-Nav also achieves a higher SPL, suggesting that the added memory provides richer navigational evidence for reasoning and leads to more efficient paths.

Figure 2 presents qualitative results from both the simulation and real-world experiments.

5.4 Ablation Study

Table 2 reports the ablation results. Because memory is central to the agent, the variants without spatial or temporal memory are not left memoryless. They keep a minimal base memory (e.g., the 8 most recent images), which provides local spatial and short-term temporal context so the agent can still run. The full model achieves the best overall performance. Without temporal memory, the agent reaches a higher OSR but takes considerably more steps. This suggests that, without temporal memory, the agent is less able to comply the task with subgoal-level awareness and has to wander around to find the target. Removing the preview module lowers token consumption because fewer images are processed, but SR also drops because the agent can no longer proactively gather information. This confirms that a single egocentric view is often insufficient, and that previewing lets a monocular agent recover panoramic information when a decision requires it.

Table 1: Results on R2R-CE val-unseen. Bold: best within the post-training group, and best among Qwen3(-VL)-8B-based methods in the zero-shot group.

Method	Navigation Backbone	Obs.	SR ↑	SPL ↑	nDTW ↑	OSR ↑	NE ↓	PL	Steps	RSR ↑
Post-training										
Aware VLN[c] Guo et al. (2026)	AwareVLN	Mono	0.647	0.563	0.656	0.744	4.10	12.79	107	0.869
Aware VLN[c] Guo et al. (2026)	AwareVLN (4-bit)	Mono	0.652	0.569	**0.660**	**0.760**	**3.89**	**12.69**	110	0.858
InternNav[c] Wei et al. (2026)	InternVLA	Mono	0.585	0.538	–	0.659	4.70	–	99	0.888
LightNav[c] Wang et al. (2026)	LightNav	Mono	**0.680**	**0.625**	0.633	0.722	3.99	–	84	**0.942**
Zero-shot										
CA-Nav[b,c] Chen et al. (2025)	Stack[a]	Mono	0.243	0.097	0.162	**0.479**	7.71	27.82	215	0.507
CA-Nav + Temporal Memory[b,c] Chen et al. (2025)	Stack[a]	Mono	0.247	0.126	0.225	0.423	7.57	23.50	184	0.584
GC-VLN[b,c] Yin et al. (2025)	DeepSeek-R1	Mono	0.141	0.066	–	0.289	8.31	42.10	366	0.488
SmartWay[c] Shi et al. (2025)	Q3VL-8B	Pano	0.178	0.134	0.380	0.361	**7.39**	15.81	12	0.493
SmartWay Shi et al. (2025)	GPT-4o	Pano	0.290	0.225	–	0.510	7.01	13.09	–	0.569
Open-Nav[c] Yuan et al. (2025)	Qwen3-8B-FP8	Pano	0.107	0.093	**0.430**	0.148	7.87	7.60	6	0.723
Open-Nav Yuan et al. (2025)	Llama-3.1-70B-Instruct	Pano	0.160	0.129	0.450	0.230	7.25	8.07	–	0.696
Open-Nav Yuan et al. (2025)	GPT-4	Pano	0.190	0.161	0.458	0.230	6.70	7.68	–	0.826
HSGM[c] Liu & Yu (2025)	Q3VL-8B	3-view	0.049	0.048	0.272	0.120	8.79	**6.27**	34	0.408
HSGM[c] Liu & Yu (2025)	GPT-5	3-view	0.463	0.324	0.463	0.569	5.47	15.36	43	0.814
Ours (Primitive Actions)	Q3VL-8B + JoyAI	Mono	0.125	0.085	0.320	0.225	8.05	15.76	213	0.556
Ours (Intent Actions)	Q3VL-8B + JoyAI	Mono	**0.263**	**0.196**	0.379	0.328	7.56	16.98	131	**0.802**

Q3VL-8B: Qwen3-VL-8B-Instruct. –: unavailable. Mono: single egocentric view; Pano: 12-view 360° panorama; 3-view: front, left, right. Reproductions keep the original view configuration. [a]GroundingDINO + RepViT-SAM + BLIP2-ITM + BLIP-VQA + Q3VL-8B. [b]LLM used only for subgoal decomposition, not step-by-step navigation. [c]Reproduced by us; other rows are from the original papers, with RSR computed from reported SR and OSR.

6 Conclusion

We asked what a navigator should remember and argued that memory should preserve the facts future decisions require rather than the raw observation stream, and that the agent should gather evidence when memory cannot justify a decision. Our spatiotemporal memory agent follows this principle. On R2R-CE val-unseen, it achieves the best SR and SPL among zero-shot methods built on 8B-scale open models, and its high RSR indicates that it recognizes the goal rather than merely reaching it. The agent also transfers to a Unitree Go2 quadruped in environments where a strong post-trained baseline degrades. Our instruction-perturbation protocols further show that a post-trained agent with high simulation success largely follows memorized routes, a failure mode that standard metrics do not expose.

Our absolute success rate remains below that of post-trained methods in simulation, and episodes require many primitive steps. The logical view guides the design but provides no quantitative guarantee, since natural-language summaries may omit or misstate facts. The occupancy map also assumes reliable depth and pose. Future work includes learning which facts to retain, extending spatial memory to semantic maps, and evaluating the remaining principles of controllable behavior.

AI use statement

In this work, we used generative AI tools for no task that requires disclosure. We have not used generative AI tools for generating synthetic data, developing theoretical models, formulating mathematical claims, writing proofs, proposing hypotheses, designing experiments, implementing methods, translation, cleaning datasets, qualitative analysis, or interpreting results, and no other required-disclosure task is applicable to this work. Additionally, we used generative AI tools for editing the manuscript to improve readability. We have reviewed all AI-assisted work. The authors checked the edited text against the manuscript and the reported results. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Reproducibility statement

We aim to make the reported results reproducible. The method, and experimental protocol are described in the paper, and the same fixed settings are used across the reported runs. Implementation details needed to repeat the experiments will be included with the public code release. After acceptance, we will release the code publicly.

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. *arXiv preprint arXiv:2212.04385*, 2022.

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

Isaac Asimov. *I, robot*, volume 1. Spectra, 2004.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Kehan Chen, Dong An, Yan Huang, Rongtao Xu, Yifei Su, Yonggen Ling, Ian Reid, and Liang Wang. Constraint-aware zero-shot vision-language navigation in continuous environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

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.

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. *arXiv preprint arXiv:2605.17249*, 2026a.

Jingzhi Huang, Junkai Huang, Haoyang Yang, Haoang Li, and Yi Wang. Aerr-nav: Adaptive exploration-recovery-reminiscing strategy for zero-shot object navigation. *arXiv preprint arXiv:2603.17712*, 2026b.

Gabriel Ilharco, Vihan Jain, Alexander Ku, Eugene Ie, and Jason Baldridge. General evaluation for instruction conditioned navigation using dynamic time warping. *arXiv preprint arXiv:1907.05446*, 2019.

Jacob Krantz, Erik Wijmans, Arjun Majundar, Dhruv Batra, and Stefan Lee. Beyond the navgraph: Vision and language navigation in continuous environments. In *European Conference on Computer Vision (ECCV)*, 2020.

Luis A Lastras, Jonathan Lenchner, Barry M Trager, Wojciech Szpankowski, Mark S Squillante, Chai Wah Wu, Ronald Fagin, and Alexander Gray. Fundamental limits incorporating logical reasoning into shannon's information theory. *Proceedings of the National Academy of Sciences*, 123(35):e2525600123, 2026.

Yijian Li, Changze Li, Hantian Shi, Jiaying Luo, Jiyuan Cai, Ming Yang, and Tong Qin. Agenticnav: Zero-shot vision-and-language navigation as a tool-calling harness. *arXiv preprint arXiv:2606.10577*, 2026.

Dong Liu and Yanxuan Yu. Hsgm: Hierarchical segment-graph memory for scalable long-text semantics. In *Proceedings of the 14th Joint Conference on Lexical and Computational Semantics (* SEM 2025)*, pp. 328–337, 2025.

Yuhang Peng, Yizhou Pan, Xinning He, Jihaoyu Yang, Xinyu Yin, Han Wang, Xiaoji Zheng, Chao Gao, and Jiangtao Gong. Freeaskworld: An interactive and closed-loop simulator for human-centric embodied ai. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 926–934, 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.

Shouwei Ruan, Liyuan Wang, Caixin Kang, Qihui Zhu, Songming Liu, Xingxing Wei, and Hang Su. Brain-inspired spatial intelligence for embodied agents. *Nature Communications*, 2026.

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. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 16923–16930. IEEE, 2025.

Shaoan Wang, Aocheng Luo, Fei Huang, Jingyi Xu, Xiaoyang Wang, Yueyu Wang, Qianli Ma, Fan Yang, Ran Mei, Jia Wei, et al. Lightnav-0: Eliciting vlm spatial intelligence for generalist embodied navigation. *arXiv preprint arXiv:2608.30935*, 2026.

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.

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. *arXiv preprint arXiv:2603.17670*, 2026.

Dingyu Yao, Junhao Zhou, Chenxu Yang, Chuanyu Qin, Haowen Hou, Zheming Liang, Congcong Wang, Yuhang Cao, Shenglong Ye, Shuai Xie, et al. Joyai-vl-interaction: Real-time vision-language interaction intelligence. *arXiv preprint arXiv:2606.14777*, 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. *arXiv preprint arXiv:2509.10454*, 2025.

Mingfeng Yuan, Letian Wang, and Steven L Waslander. Opennav: Open-world navigation with multimodal large language models. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 18948–18955. IEEE, 2025.

A Additional Experiment Results

Table 2: Ablation study evaluation of model variants across navigation performance, path efficiency, step budget, and average token consumption per episode. ✓ indicates component enabled, while −− indicates component removed. ↑ indicates higher is better; ↓ indicates lower is better. Best performance per metric is highlighted in bold.

Model Variant	**Full**	**w/o Spatial**	**w/o Temporal**	**w/o S&T**	**w/o Preview**
Spatial	✓	–	✓	–	✓
Temporal	✓	✓	–	–	✓
Preview	✓	✓	✓	✓	–
Navigation Performance					
SR ↑	**0.263**	0.205	0.190	0.185	0.190
SPL ↑	**0.196**	0.142	0.112	0.139	0.151
OSR ↑	0.328	0.345	0.475	**0.492**	0.295
NE ↓	**7.56**	8.48	9.31	8.46	8.05
Path & Trajectory Efficiency					
Path ↓	16.9	18.6	39.1	31.7	**16.7**
Steps ↓	131	173	292	300	**130**
nDTW ↑	**0.379**	0.316	0.201	0.202	0.378
SDTW ↑	**0.168**	0.131	0.095	0.117	0.141
Reach 500 Steps ↓	162	171	657	693	**108**
Budget & Speed					
SR@100 Step ↑	**0.185**	0.115	0.100	0.090	0.145
SR@150 Step ↑	**0.210**	0.140	0.110	0.150	0.180
Med. Steps ↓	68	85	96	120	**67**
Token Consumption per Episode					
8B In ↓	640k	819k	1084k	1134k	**523k**
8B Out ↓	10.9k	13.9k	27.1k	27.9k	**9.4k**
JoyAI In ↓	224k	282k	**114k**	119k	200k
JoyAI Out ↓	5.4k	8.2k	10.1k	10.7k	**1.6k**

Table 3: Comparison of VLM call frequency and token consumption between HSGM and Ours, reported per call/step and per episode.

	VLM Calls	**Per Call / Step**		**Per Episode**		
Method	**/ Episode**	**Input**	**Output**	**Input**	**Output**	**Total**
HSGM	11.7	≈23.0k	285	≈270k	≈3.3k	≈273.3k
Ours	143 steps†	≈6.0k	≈114	864k	16.3k	880.3k
– *Qwen3-VL-8B*	167	≈4.5k	≈76	640k	10.9k	651.9k
– *JoyAI*	143	≈1.6k	≈38	224k	5.4k	229.4k

† Ours averages 143 decision steps, 24 gate triggers, and 143 observer calls per episode.

Table 4: Real-world evaluation in the corridor and the office. *Simple Instruction*: follow a short instruction and stop. *Specific Object*: find a named object and stop in front of it. NE (m) is the navigation error; SR is the success rate within 3.0 m.

	Corridor				Office			
	Simple Instruction		Specific Object		Simple Instruction		Specific Object	
Method	NE↓	SR↑	NE↓	SR↑	NE↓	SR↑	NE↓	SR↑
AwareVLN	2.85	0.3	3.20	0.1	3.10	0.2	3.46	0.0
Ours	**0.68**	**0.8**	**0.84**	**0.8**	**0.91**	**0.6**	**1.07**	**0.4**

Table 5: FLIP experiment on 82 selected episodes of R2R Val Unseen. ORIG uses the original instructions; FLIP reverses the turning directions. Follow, Opposite, and Straight are the fractions of turning decisions that follow the given instruction, go in the opposite direction, or go straight.

Method	Run	Follow↑	Opposite↓	Straight	SR↑
Ours	ORIG	0.829	0.134	0.037	0.286
	FLIP	0.707	0.220	0.073	0.143
AwareVLN	ORIG	0.915	0.073	0.012	0.646
	FLIP	0.439	0.549	0.012	0.463
CA-Nav	ORIG	0.646	0.268	0.085	0.256
	FLIP	0.268	0.671	0.061	0.171

Table 6: GOAL-ONLY experiment on 174 episodes of R2R Val Unseen, where only the last sub-instruction is kept.

Method	SR↑	
	ORIG	GOAL-ONLY
Ours	0.293	0.181
AwareVLN	0.624	0.353
CA-Nav (GPT-4 parse)	0.241	0.184

Table 7: PARAPHRASE experiment on 200 episodes of R2R Val Unseen. A1: same register; A2: terse; A3: colloquial; A4: landmark shift. NE is in meters. CA-Nav uses Qwen for instruction parsing.

Method	Instruction	SR↑	OSR↑	RSR↑	SPL↑	NE↓	nDTW↑
Ours	ORIG	0.270	0.350	0.771	0.155	8.15	0.387
	A1	0.260	0.340	0.765	0.149	7.98	0.357
	A2	0.240	0.330	0.727	0.123	8.19	0.365
	A3	0.260	0.340	0.765	0.136	8.06	0.350
	A4	0.280	0.360	0.778	0.145	8.67	0.351
AwareVLN	ORIG	0.590	0.760	0.776	0.499	4.11	0.621
	A1	0.630	0.720	0.875	0.509	4.60	0.611
	A2	0.610	0.720	0.847	0.517	4.79	0.641
	A3	0.620	–	–	–	–	0.626
	A4	0.500	–	–	–	–	0.566
CA-Nav	ORIG	0.260	0.500	0.520	0.127	6.60	0.156
	A1	0.250	–	–	–	–	0.141
	A2	0.280	–	–	–	–	0.131
	A3	0.210	0.520	0.404	0.088	7.36	0.130
	A4	0.210	–	–	–	–	0.086