GROUNDINGVLN: REASONING AND ACTING WITH GROUNDING FOR VISION-LANGUAGE NAVIGATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Although vision-language models (VLMs) possess strong visual understanding and reasoning capabilities, vision-and-language navigation (VLN) agents built upon them still struggle to bridge semantic reasoning and spatial execution. This disconnection stems from two coupled limitations: intermediate reasoning is not explicitly grounded in visual evidence, and high-level decisions lack precise spatial goals to guide low-level motion. Inspired by human navigation, which naturally links cognition and locomotion through landmarks and spatial goals, we propose **GroundingVLN**, a framework that uses visual grounding as a shared interface between reasoning and action. GroundingVLN first reasons with grounding by anchoring task-relevant visual evidence to precise image locations throughout structured reasoning. It then acts through grounding by predicting a progress-aligned pixel goal, which is translated into primitive actions by a geometric planner. To learn these capabilities, we construct **GroundingCOTVLN-188K**, a dataset of visually grounded reasoning traces synchronized with navigation progress, and introduce **Grounded and Execution-Aware Reinforcement Learning (GEAR)**, which aligns grounded reasoning and spatial decisions with downstream execution. Experiments demonstrate that GroundingVLN achieves state-of-the-art performance (**69.9%** SR on R2R-CE and **75.1%** SR on RxR-CE) with superior data efficiency, requiring only **0.9%** of the training data used by the strongest baseline. It also generalizes strongly across datasets, attaining 59.9% SR on RxR-CE when trained solely on R2R, a gain of 20.1% over the strongest baseline. The full code, dataset, and model will be released after review.

1 INTRODUCTION

Vision-and-Language Navigation (VLN) requires an embodied agent to follow natural-language instructions in continuous, partially observable environments (Anderson et al., 2018b; Krantz et al., 2020). Successful navigation requires grounding language in egocentric observations, tracking instruction progress, planning sub-tasks, and executing actions. Although recent vision-language models (VLMs) exhibit strong visual understanding and reasoning capabilities (Qwen Team, 2026; Wang et al., 2025b), translating them into reliable long-horizon navigation remains challenging.

Existing methods still struggle to bridge high-level semantic reasoning and low-level action execution. A dominant line of work fine-tunes VLMs to directly predict discrete actions, such as moving forward, turning, and stopping (Zhang et al., 2024; 2025a; Cheng et al., 2025). However, directly mapping visual observations to low-level actions offers limited interpretability and requires substantial navigation trajectories to learn motor control, rather than fully leveraging the pretrained reasoning capabilities of VLMs. Alternatively, waypoint-based methods select traversable candidates generated by an external waypoint predictor (Shi et al., 2025; Qiao et al., 2025), avoiding direct low-level action prediction but restricting action space to predicted candidates, which may limit overall performance. Recent approaches introduce Chain-of-Thought (CoT) reasoning to improve long-horizon decision-making and interpretability (Guo et al., 2026; Zuo et al., 2026). Nevertheless, their intermediate reasoning relies primarily on text or latent features, making it difficult to bind referenced evidence to precise regions in the current view or to express an executable spatial target. Consequently, two coupled limitations persist: the reasoning process lacks explicit visual evidence anchored through spatial grounding, and high-level decisions lack a geometrically precise, training-friendly interface to guide low-level action execution.

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.

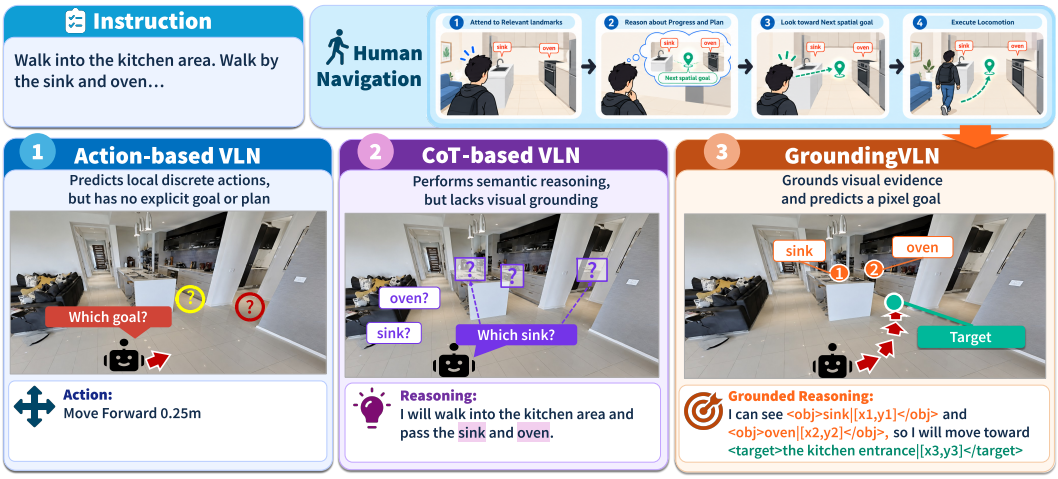


Figure 1: Human navigation grounds planning in relevant landmarks and spatial goals while separating cognition from motor control. Action-based VLN lacks explicit spatial goals, while text-only CoT leaves reasoning evidence ungrounded. GroundingVLN bridges these two coupled gaps by binding task-relevant evidence to image locations and predicting progress-aligned pixel goals, providing a shared grounding interface between high-level reasoning and low-level execution.

Cognitive science suggests that human navigation follows a hierarchical organization, in which higher-level cognitive systems plan routes and spatial goals (Balaguer et al., 2016; Epstein et al., 2017), while brainstem and spinal circuits execute locomotion (Leiras et al., 2022). Crucially, humans selectively encode relevant landmarks at decision points (Janzen & Van Turennout, 2004) and proactively direct their gaze toward upcoming landing targets before stepping (Patla & Vickers, 2003). Together, these findings motivate visual grounding as the interface between cognition and action: Grounding instruction-relevant landmarks can support localization, progress estimation, and planning, while grounding the next intended destination provides a spatial objective for low-level execution. Such an interface is feasible with modern VLMs, which already possess strong visual understanding, reasoning, and coordinate-level pointing capabilities (Qwen Team, 2026; Zhang et al., 2026b). We therefore cast the VLM as a high-level cognitive system that grounds reasoning in visual evidence and decisions in spatial goals, leaving motor execution to a low-level planner.

Building on this insight, we propose **GroundingVLN**, a grounding-centric framework that integrates grounding during reasoning and decision making. During reasoning, the VLM follows a structured process of subtask decomposition, history recall, current perception, progress estimation, and future-goal planning, while binding key evidence in its reasoning chain to precise image locations. During decision making, it predicts a pixel goal aligned with the current semantic subtask, which a low-level planner converts into an executable trajectory and actions. To train these capabilities, we construct **GroundingCOTVLN-188K** for supervised finetuning and enforce temporal alignment of grounding across changing viewpoints. We further introduce **Grounded and Execution-Aware Reinforcement Learning (GEAR)**, which exploits the measurable geometry of pixel goals to provide fine-grained rewards that jointly align reasoning, spatial decisions, and downstream execution. Extensive experiments show that GroundingVLN achieves state-of-the-art performance on both R2R-CE and RxR-CE using only **0.9%** of the training data used by the strongest baseline, while demonstrating strong cross-dataset generalization.

Our main contributions are as follows:

- We introduce a grounding-centric VLN paradigm that decouples high-level cognition from low-level motion: grounding during reasoning binds intermediate thoughts to visual evidence, while grounding during decision making converts semantic subgoals into executable spatial goals.
- We build an automatic data engine to construct **GroundingCOTVLN-188K**, with a mechanism for temporal alignment of grounding that preserves grounded entities across changing

viewpoints. We further propose **GEAR**, which exploits the continuous geometry of pixel goals to provide fine-grained rewards that align reasoning, spatial decisions, and execution.

- Extensive experiments demonstrate that GroundingVLN achieves state-of-the-art performance on R2R-CE and RxR-CE with only **0.9%** of the training data used by the strongest baseline, while exhibiting strong data efficiency and cross-dataset generalization.

2 Related Work

2.1 Action Representations in Vision-and-Language Navigation

Recent VLN systems increasingly adapt pretrained VLMs as navigation policies. NaVid, Uni-NaVid, and StreamVLN directly generate low-level actions (Zhang et al., 2024; 2025a; Wei et al., 2025), while NaVILA outputs language-form mid-level actions for a locomotion policy (Cheng et al., 2025). Such action prediction offers a simple interface, but provides limited supervision for task progress and visual evidence, often requires substantial training data, and may compromise the VLM's general-purpose capabilities through catastrophic forgetting (Hancock et al., 2026). Waypoint-based methods instead select among traversable candidates proposed by a dedicated predictor (Krantz et al., 2021; An et al., 2025; Shi et al., 2025); although avoiding direct low-level action prediction, their action space and performance remain bounded by candidate quality. DualVLN further predicts the farthest visible pixel goal, but conditions its diffusion executor on latent VLM features (Wei et al., 2026). In contrast, GroundingVLN uses a progress-aligned pixel goal as an explicit interface between semantic reasoning and low-level planning.

2.2 Chain-of-Thought Reasoning in Vision-and-Language Navigation

To improve interpretability and long-horizon decision making, NavGPT-2 and NavCoT generate textual chains of thought before predicting actions (Zhou et al., 2024; Lin et al., 2025). However, free-form reasoning can be verbose, hallucinate visual details, and increase inference cost. Recent methods therefore adopt more structured reasoning: EvolveNav combines formalized CoT supervision with self-improvement (Lin et al., 2026), AwareVLN reasons about agent state and task progress at key decision points (Guo et al., 2026), and FantasyVLN incorporates visual imagination into multimodal CoT (Zuo et al., 2026). Nevertheless, their intermediate conclusions are not explicitly bound to image regions, making spatial descriptions difficult to verify against current observations. GroundingVLN instead introduces coordinate-based grounding into structured CoT, directly linking each reasoning step to its visual evidence.

2.3 Reasoning with Grounding

Multimodal reasoning is shifting from text-only CoT toward explicit interaction with visual evidence. Visual CoT Prompting iterates through seeing, reasoning, and verification (Chen et al., 2024b), while V* introduces guided visual search for task-relevant regions (Wu & Xie, 2024). More recent methods directly ground intermediate thoughts: ViGoRL anchors each reasoning step to image coordinates (Sarch et al., 2025), and Thinking with Visual Grounding interleaves language with point or box references (Zhang et al., 2026b). However, these methods primarily address single-image reasoning and cannot preserve entity identity across changing viewpoints. GroundingVLN extends grounded reasoning to closed-loop navigation through temporally aligned grounding, allowing visual evidence to be tracked and reused throughout a trajectory.

3 Method

3.1 Problem Formulation

We consider vision-and-language navigation in an unknown continuous 3D environment. Given a natural-language instruction $I = (w_1, \ldots, w_L)$, an agent starts from state s_0 and receives an egocentric visual observation o_t at each time step t. Based on the instruction, current observation, and interaction history $\mathcal{H}_t = \{(o_\tau, a_\tau)\}_{\tau=0}^{t-1}$, the policy π_θ produces a navigation decision $a_t \sim \pi_\theta(a_t \mid I, o_t, \mathcal{H}_t)$. The agent must reach the destination and stop within a finite horizon.



Figure 2: **Overview of GroundingVLN.** (A) The instruction is first decomposed into ordered subtasks. The high-level VLM performs grounded reasoning and predicts the current subtask, action, and target, which the low-level execution module converts into primitive actions. (B) GroundingCOTVLN-188K provides grounded CoT supervision through subtask–goal alignment and temporal alignment of grounding. (C) GEAR combines an execution-aware reward map with multi-level rewards for group-relative policy optimization.

3.2 Framework of GroundingVLN

Figure 2 summarizes GroundingVLN's decoupled, closed-loop design. (A) After decomposing the instruction into ordered subtasks, the VLM performs structured, visually grounded reasoning to recall past execution, perceive the scene, assess progress, and plan the next goal. It anchors relevant evidence with coordinate-tagged `<obj>` and `<prev_obj>` references. It then predicts the current subtask, a high-level action (`MOVE`, `TURN`, or `STOP`), and a target pixel as a decision. For `MOVE`, the low-level module back-projects the 2D pixel goal into 3D, plans a path, and executes it. (B) To train grounded reasoning and pixel-goal decision-making, we construct the GroundingCOTVLN-188K dataset. Subtask–goal alignment ensures that each decision's pixel goal matches subtask progress, while temporal alignment of grounding keeps visual references consistent across observations during reasoning. (C) GEAR (Grounded and execution-aware reinforcement learning) combines an execution-aware pixel-goal reward map with multi-level rewards for group-relative policy optimization, further improving grounding and decision-making.

3.3 Reasoning and Acting with Grounding

At each decision step, GroundingVLN first binds evidence to image locations during structured reasoning, then grounds the resulting decision to a pixel goal. A low-level planner converts this goal into primitive actions, closing the loop between reasoning, decision making, and execution.

Structured Reasoning with Grounding. Inspired by the hierarchical and goal-directed nature of human navigation, we decompose the instruction once at initialization into ordered subtasks, identifying the turns and visual evidence required by each. At each subsequent decision step, the VLM performs four ordered reasoning stages:

1. **History Recall:** Summarize the previous execution, including traveled distance, heading changes, and observed landmarks.
2. **Current Perception:** Describe the current scene and orientation while identifying visible instruction-relevant evidence.

3. **Progress Estimation:** Determine which requirements have been satisfied, whether the current subtask is complete, and what remains unresolved.
4. **Future Planning:** Select the next local objective according to the earliest unfinished instruction condition.

To bind this reasoning process to the observation, we embed visual grounding directly into the CoT. A newly confirmed entity is represented as `<obj>l|[x,y]</obj>`, whereas a previously grounded entity that reappears in the current view is marked as `<prev_obj>l|[x,y]</prev_obj>`. Here, l denotes a semantic label, and all grounding and target locations use normalized integer coordinates $(x, y) \in [0, 1000]^2$ in the current view. These references make each visual claim spatially verifiable and allow historical evidence to be explicitly reused during progress estimation and planning.

Progress-Aligned Pixel-Goal Decision. After structured reasoning, the VLM predicts a high-level action $a_t^h \in \{\texttt{MOVE}, \texttt{TURN}, \texttt{STOP}\}$. A `MOVE` always includes a pixel goal $g_t = (c_t, l_t, p_t)$, specifying the current subtask c_t, target label l_t, and location $p_t = (x_t, y_t)$. Its walkable destination is marked in the CoT as `<target>l|[x,y]</target>`, distinct from the evidential landmarks denoted by `<obj>` and `<prev_obj>`. A `TURN` specifies $d_t \in \{\texttt{left}, \texttt{right}, \texttt{around}\}$ and includes the same target tag and pixel goal only when its intended post-turn destination is visible in the current observation. The current step executes only an in-place rotation of 90° for `left`/`right` or 180° for `around`. `STOP` is emitted only when all instruction conditions are satisfied.

Unlike DualVLN, which supervises the farthest visible trajectory point (Wei et al., 2026), we align the pixel goal with semantic progress. We partition the reference trajectory into ordered subtask segments and restrict candidates at step t to the current segment, terminating at its next decision boundary. Among candidates that are continuously visible and traversable, we select the point with the greatest forward path progress and project it into the current view. The model jointly predicts (c_t, l_t, p_t) at inference, coupling what remains to be completed with where the agent should move next without prematurely crossing into a future subtask.

Low-Level Planning and Execution. The execution module handles each decision according to its action type: `TURN` triggers a fixed in-place rotation, with or without a pixel goal, and `STOP` ends navigation. Only `MOVE` uses its pixel goal to plan and execute a path. RTAB-Map (Labbé & Michaud, 2019) estimates the camera pose online from the RGB-D stream, while accumulated point clouds form a map of ground support, obstacles, and elevation changes, including stairs. For each `MOVE`, the module converts the target to local pixel coordinates, then uses the corresponding depth, camera intrinsics, and estimated pose to recover its 3D world position. A* searches the map for a collision-free path to this position, and a path follower translates the path into low-level turns and forward movements.

3.4 GROUNDINGCOTVLN-188K: AUTOMATIC GROUNDED SFT DATA GENERATION

To train structured reasoning and pixel-goal prediction, we construct **GroundingCOTVLN-188K**, comprising 188K grounded image–CoT samples collected from 21K R2R and RxR trajectories.

Trajectory Collection. For each instruction–reference-path pair, we replay the trajectory in Habitat (Savva et al., 2019). Densely spaced reference points are merged, long segments are interpolated, and the low-level planner drives the agent between successive points while recording its actual executed trajectory. We sample three types of decision anchors: *arrival anchors* at reference points, *turn anchors* around substantial in-place rotations, and *mid-distance anchors* after sufficient travel. Additional anchors are inserted across large gaps, whereas near-static or redundant samples are removed. Each retained anchor stores stitched left-front-right RGB-D observations, camera pose, reference-point index, and executed position, providing the geometric basis for progress annotation, visual projection, and pixel-goal construction.

Subtask–Goal Alignment. We first use the teacher VLM to decompose the instruction into ordered subtasks $\{u_k\}_{k=1}^{K}$ and extract landmarks and turns. Using replayed observations and reference actions, each u_k is aligned with a contiguous path segment $\mathcal{S}_k$, allowing every anchor t to be assigned a current progress segment $\mathcal{S}_{k_t}$ and a set of completed subtasks.

Pixel-goal candidates are further constrained by actual execution between the current and next anchors. Let $\mathcal{T}_{t:t+1}$ denote this executed trajectory segment; we select

$$q_t^* = \underset{q \in \mathcal{T}_{t:t+1} \cap \mathcal{S}_{k_t}}{\arg\max} \ \mathrm{Prog}(q), \quad \text{s.t.} \quad \mathrm{Visible}(q, o_t) \wedge \mathrm{Navigable}(q), \tag{1}$$

where $\mathrm{Prog}(q)$ measures forward progress along the path, and $\mathrm{Visible}(q, o_t)$ denotes continuous visibility along the trajectory up to q in o_t. Depth-based occlusion checks ensure that q_t^* lies on a genuinely visible ground region, after which camera projection produces $p_t = \Pi_t(q_t^*) \in [0, 1000]^2$. This construction prevents targets from crossing into a future subtask while preserving their physical executability.

Grounded CoT Synthesis. For each anchor, annotation proceeds in two stages. The teacher VLM first describes the agent orientation, spatial layout, visible landmarks, and task-relevant evidence in the current observation; it then performs directed grounding over referenced entities and inserts `<obj>l|[x,y]</obj>` tags.

Unlike grounding in a static image, VLN requires temporal alignment of grounding as the viewpoint changes. The same entity moves across image coordinates and may become occluded, creating two risks: grounding may drift to a similar instance, or a temporarily invisible entity may be forgotten. We therefore back-project every new grounding into a uniquely identified 3D world marker and initialize SAM 3 tracking (Carion et al., 2026). At later anchors, historical markers are first reprojected into the current view; when reprojection is invalid, SAM 3 recovers the instance mask and valid depth points are back-projected to update its world position. A track is accepted only when its new 3D estimate satisfies spatial continuity with the stored marker. Invisible entities retain their identity in memory and, when observed again, are rebound to the current pixel location using `<prev_obj>`.

Finally, deterministic programs assemble executed motion, current observation, progress state, action, and pixel goal, while the teacher VLM verbalizes these facts as a structured CoT. Samples are retained only if they pass coordinate visibility, depth occlusion, action–reasoning consistency, format completeness, and annotation-leakage checks, yielding the final GroundingCOTVLN-188K dataset. Appendix A.7 provides further implementation details.

3.5 Grounded and execution-aware reinforcement learning

The continuous and measurable geometry of pixel goals enables finer post-training signals than discrete action correctness alone. We therefore introduce **GEAR** (Grounded and execution-aware reinforcement learning), which constructs execution-aware 2D goal reward maps and combines them with multi-level rewards through Group Relative Policy Optimization (GRPO) (Shao et al., 2024). Appendix A.9 gives the exact reward definitions, and Appendix A.10 provides the optimization details.

Execution-Aware Pixel-Goal Reward Maps. Rewarding only the 2D distance to a reference pixel ignores its mismatch with physical distance and cannot reflect obstacles or terrain. For each anchor t, we therefore construct a reward map $\mathcal{M}_t$ over the image domain. A candidate pixel p is back-projected to obtain an execution endpoint $P_t(p)$; endpoints on walls, furniture, or other non-traversable surfaces are removed using the reconstructed local traversability map. For each valid candidate, we define

$$Q_t(p) = \lambda_r \exp\left(-\frac{d_\perp(p)^2}{2\sigma_r^2}\right) + \lambda_p \exp\left(-\frac{(s(p) - s_t^*)^2}{2\sigma_p^2}\right) + \lambda_e \exp\left(-\frac{d_e(p)^2}{2\sigma_e^2}\right), \tag{2}$$

where $d_\perp(p)$ is the lateral distance to the reference route, $s(p) - s_t^*$ is the route-progress difference from the reference target, and $d_e(p) = \|P_t(p) - P_t^*\|_2$ measures execution-endpoint error. Thus, Q_t jointly scores route consistency, semantic progress, and physical executability; invalid pixels receive the minimum map value. Figure 2(C) visualizes the resulting spatial distribution of Q_t.

Multi-Level Navigation Reward and Group-Relative Policy Optimization. Beyond the pixel-goal reward map, we design complementary rewards for output validity, action selection, subtask progress, and visual grounding. Invalid outputs receive the minimum reward; otherwise, for each action–target case y, we compute

$$R_y = \alpha_y^a R_{\mathrm{act}} + \alpha_y^t R_{\mathrm{task}} + \alpha_y^g R_{\mathrm{goal}} + \lambda_g R_{\mathrm{grd}} - P, \qquad R_{\mathrm{goal}}(p) = \phi(Q_t(p)), \tag{3}$$

where R_{act}, R_{task}, and R_{grd} evaluate the action, current subtask, and the semantic and spatial correctness of grounded evidence, respectively; P penalizes inconsistencies between reasoning and decisions. For each anchor, we sample G outputs and optimize their group-normalized rewards with a clipped group-relative policy objective, encouraging responses that jointly improve reasoning, grounding, and executable navigation.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Benchmarks. We evaluate GroundingVLN on R2R-CE and RxR-CE under the continuous-environment VLN setting (Anderson et al., 2018b; Krantz et al., 2020; Ku et al., 2020). Both benchmarks are implemented in Habitat (Savva et al., 2019) using scenes from Matterport3D (Chang et al., 2017), and we report results on their `val_unseen` splits. Following standard protocols (Anderson et al., 2018a; Ilharco et al., 2019), we report Navigation Error (NE), Oracle Success Rate (OS), Success Rate (SR), and Success weighted by Path Length (SPL), where success requires stopping within 3 m of the goal; for RxR-CE, we additionally report normalized Dynamic Time Warping (nDTW).

Implementation Details. We use Qwen3.5-397B-A17B (Qwen Team, 2026) to annotate GroundingCOTVLN-188K and initialize GroundingVLN from Qwen3.5-4B. Training consists of two stages. We first perform one epoch of supervised fine-tuning on 188K grounded image–CoT samples covering 21K R2R and RxR trajectories, which takes approximately 8 hours on 8 NVIDIA H200 GPUs. We then apply GEAR to 16K samples for 2,000 optimization steps with a learning rate of 1×10^{-6}, taking approximately 20 hours on the same hardware.

Table 1: Comparison on R2R-CE and RxR-CE `val_unseen`. Data reports the number of individual training samples stated by each method; counting details are provided in Appendix A.1. Params reports backbone scale, excluding additional learned components (+). † includes 13.1M samples inherited from NaVILA and 1.9M AwareVLN-specific samples. ‡ comprises 16.9M expert trajectory samples and 5.0M reasoning samples.

Method	Data	Params	R2R-CE Val-Unseen				RxR-CE Val-Unseen			
			NE↓	OS↑	SR↑	SPL↑	NE↓	SR↑	SPL↑	nDTW↑
HPN+DN Krantz et al. (2021)	–	–	6.31	40.0	36.0	34.0	–	–	–	–
CMA Hong et al. (2022)	–	–	6.20	52.0	41.0	36.0	8.76	26.5	22.1	47.0
GridMM Wang et al. (2023a)	–	–	5.11	61.0	49.0	41.0	–	–	–	–
ETPNav An et al. (2025)	–	–	4.71	65.0	57.0	49.0	5.64	54.7	44.8	61.9
ScaleVLN Wang et al. (2023b)	4.9M	–	4.80	–	55.0	51.0	–	–	–	–
InstructNav Long et al. (2025)	0	–	6.89	–	31.0	24.0	–	–	–	–
SmartWay Shi et al. (2025)	0	–	7.01	51.0	29.0	22.5	–	–	–	–
OpenNav Qiao et al. (2025)	0	–	6.70	23.0	19.0	16.1	–	–	–	–
HSGM Li et al. (2026)	0	–	5.42	58.7	47.9	32.8	7.43	41.8	25.1	54.9
VLN-R1 Qi et al. (2025)	1.83M	7B	7.00	41.2	30.2	21.8	9.10	22.7	17.6	–
NaVid Zhang et al. (2024)	1.28M	7B	5.47	49.1	37.4	35.9	–	–	–	–
NaVILA Cheng et al. (2025)	13.1M	8B	5.22	62.5	54.0	49.0	6.77	49.3	44.0	58.8
Uni-NaVid Zhang et al. (2025a)	3.58M	7B	5.58	53.5	47.0	42.7	6.24	48.7	40.9	–
StreamVLN Wei et al. (2025)	∼26.3M	7B	4.98	64.2	56.9	51.9	6.22	52.9	46.0	61.9
NavFoM Zhang et al. (2026a)	12.7M	7B	4.61	72.1	61.7	55.3	4.74	64.4	56.2	65.8
MapDream Lian et al. (2026)	1.7M	2B+	4.59	64.4	59.8	54.4	4.96	59.4	49.2	–
DualVLN (S2)+SPF Wei et al. (2026)	∼12.0M	7B	4.25	68.3	60.9	55.2	5.71	63.5	55.0	46.8
DualVLN Wei et al. (2026)	∼12.0M	7B+	4.05	70.7	64.3	58.5	4.58	61.4	51.8	70.0
JanusVLN Zeng et al. (2026)	10.69M	7B	4.78	65.2	60.5	56.8	6.06	56.2	47.5	62.1
AwareVLN Guo et al. (2026)	15.0M†	8B	4.02	73.5	65.4	55.1	3.95	67.6	56.1	65.7
ABot-N0 Chu et al. (2026)	21.9M‡	4B+	3.78	70.8	66.4	63.9	3.83	69.3	60.0	–
GroundingVLN (Ours)	**188K**	**4B**	**3.66**	**74.8**	**69.9**	**64.1**	**3.54**	**75.1**	**62.0**	**75.3**

4.2 Main Results

Results on VLN-CE benchmark. As shown in Table 1, GroundingVLN achieves the best results across all reported metrics on R2R-CE and RxR-CE, with absolute SR gains of 3.5% and 5.8% over ABot-N0, respectively. It also surpasses ETPNav, the strongest waypoint-based baseline, and pixel-goal-based DualVLN. This advantage persists against DualVLN (System 2) + SPF, which uses Habitat's Shortest Path Follower for idealized execution of System 2's predicted pixel goals. Together, these findings suggest that visually grounded reasoning and spatial goals aligned with semantic progress and execution constraints support more reliable navigation. Notably, GroundingVLN achieves these results with only 188K training samples, approximately 0.9% of ABot-N0's total reported training data, demonstrating strong sample efficiency.

Table 2: R2R-to-RxR transfer on the RxR-CE `val_unseen` split without using RxR-CE training data.

Method	NE↓	OS↑	SR↑	SPL↑
NaVid Zhang et al. (2024)	8.41	34.5	23.8	21.2
MonoDream Wang et al. (2026a)	8.57	35.9	25.1	21.6
Progress-Think Wang et al. (2026b)	8.30	38.3	27.5	22.7
NaVILA Cheng et al. (2025)	8.78	46.8	34.3	28.2
AwareVLN Guo et al. (2026)	7.15	51.0	39.8	36.0
GroundingVLN (Ours)	**5.21**	**65.8**	**59.9**	**48.2**

Cross-Dataset Generalization. As shown in Table 2, GroundingVLN substantially outperforms all baselines in direct R2R-to-RxR transfer without any RxR-CE training data. It achieves 59.9% SR and 48.2% SPL, exceeding the strongest baseline, AwareVLN, by absolute margins of 20.1% and 12.2%, respectively. These pronounced gains suggest that explicit subtask progress, visually grounded evidence, and the semantic pixel-goal interface provide a highly transferable navigation representation rather than overfitting to the instruction style and trajectories of R2R.

4.3 Ablation Study

Table 3: Ablation of key components on the R2R-CE `val_unseen` split.

Method	NE↓	OS↑	SR↑	SPL↑
Full model	**3.66**	74.8	**69.9**	**64.1**
w/o <obj> and <prev_obj>	3.94	68.8	66.1	62.8
w/o temporal alignment of grounding	4.48	67.8	62.2	58.1
w/o GEAR	4.65	66.0	57.2	49.5
w/o Subtask Decomposition	4.05	**74.9**	68.1	62.2
w/ 2D pixel-distance reward	4.07	73.2	66.2	60.7

Key Component Ablations. As shown in Table 3, all components contribute to navigation success. Removing GEAR causes the largest absolute SR drop (12.7%), while replacing its execution-aware reward map with 2D pixel-distance supervision also degrades performance, highlighting the importance of aligning optimization with physical execution. Removing <obj> and <prev_obj> lowers SR by 3.8 percentage points, supporting the role of grounded reasoning in linking navigation decisions to visual evidence. Disabling temporal alignment of grounding during data annotation and retraining on the resulting annotations causes a larger drop, suggesting that inconsistent cross-view evidence can mislead reasoning. Subtask decomposition further improves success and path efficiency by organizing long instructions into coherent stages.

Grounding Accuracy Analysis. To assess GEAR's effect on grounding quality and its relationship with navigation accuracy, we manually annotate 100 episodes from R2R-CE `val_unseen`. Figure 3(a) compares SFT with SFT+GEAR, showing improved grounding recall and F1 after GEAR. Figure 3(b) compares grounding accuracy between successful and failed episodes, showing

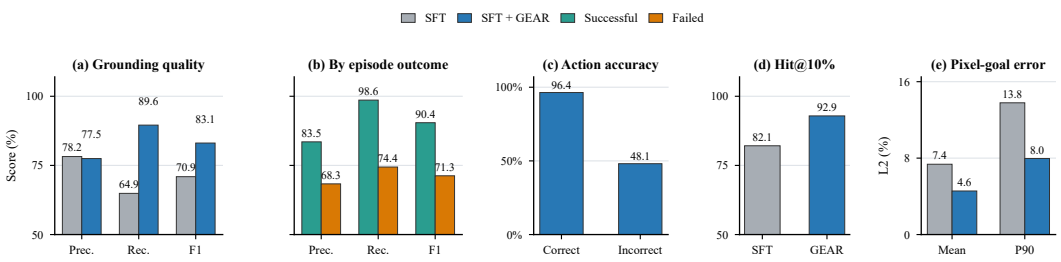
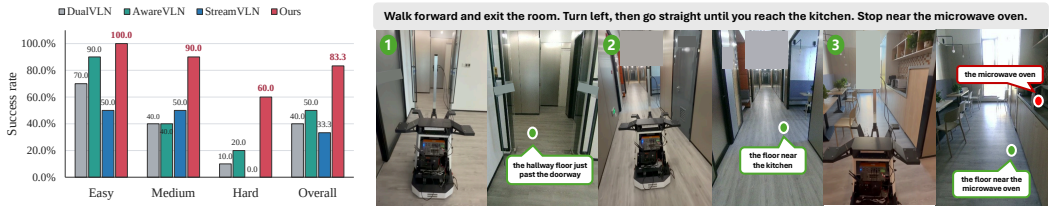


Figure 3: Grounding accuracy and pixel-goal prediction. (a) Grounding quality under SFT and SFT+GEAR. (b) Grounding by episode outcome. (c) Action accuracy conditioned on grounding correctness. (d) The fraction of goals with normalized L2 error $\leq 10\%$. (e) Localization error.



(a) Real-world success rates

(b) A real-world navigation example

Figure 4: Real-World Performance Analysis. (a) Success rates over 10 episodes per difficulty level. (b) Visualization of a real-world navigation example.

higher grounding accuracy in successful episodes. Figure 3(c) examines the association between grounding correctness in reasoning and action correctness across all decision steps. Correct grounding is associated with nearly twice the action accuracy, linking reliable visual evidence to better decisions. Higher Hit@10% and lower mean and tail localization errors in Figure 3(d,e) further indicate that GEAR improves spatial goal precision alongside grounding accuracy. These results show that GEAR substantially strengthens grounding during both reasoning and decision making, yielding a large improvement in final navigation success.

4.4 Real-World Deployment

We conduct real-world experiments on an AgileX TRACER 2.0 wheeled robot equipped with an Insta360 X5 and an Intel RealSense D435. The camera setup uses a mounting height of 1.2 m and a downward tilt of 30°. GroundingVLN inference runs on a single NVIDIA GeForce RTX 4090 GPU. As shown in Figure 4(a), GroundingVLN outperforms all three baselines across every difficulty level. Its advantage is especially pronounced on medium and hard tasks. GroundingVLN is substantially more robust in translating compositional language instructions into temporally consistent navigation behaviors, particularly when successful execution requires maintaining intermediate goals, respecting spatial constraints, and performing long-horizon action transitions.

5 Conclusion

We introduced **GroundingVLN**, a grounding-centric framework that connects high-level reasoning and low-level execution through visual grounding. It anchors task-relevant evidence during structured reasoning and translates semantic progress into executable pixel goals. **GroundingCOTVLN-188K** provides temporally aligned grounded supervision, while **GEAR** aligns reasoning and spatial decisions with downstream execution. Experiments demonstrate state-of-the-art performance, strong data efficiency, and cross-dataset generalization, showing that visual grounding is an effective interface for adapting pretrained VLMs to embodied navigation.

AI USE STATEMENT

In this work, we used generative AI tools for generating synthetic data sets, implementing methods, assisting with translation, and cleaning and reformatting data sets. We have not used generative AI tools for developing theoretical models or conceptual frameworks, proposing or refining hypotheses, designing or providing feedback on research methodology or experiments, or interpreting results; formulating mathematical claims, providing critical ingredients for proving them, writing proofs, and qualitative or thematic data analysis are not applicable to this work. Additionally, we used generative AI tools for searching for information, identifying relevant literature, creating or editing software code, and editing the paper to improve readability. We have reviewed all AI-assisted work. The core research idea and method were developed by the authors. AI-generated code was inspected and tested before use. Qwen3.5-397B-A17B served as the teacher VLM for GroundingCOTVLN-188K; its generated annotations were validated and filtered as described in Section 3.4 and Appendix A.7. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Section 3 describes the model, data-generation pipeline, low-level execution, and GEAR objective. Section 4.1 specifies the benchmarks, metrics, model initialization, and training setup. Appendix A.7 details trajectory replay, annotation, target construction, and filtering; Appendices A.8, A.9, and A.10 provide the SFT formulation, reward definitions, and GEAR optimization settings. The inference code is included in the supplementary material submitted with this paper. The complete codebase, GroundingCOTVLN-188K dataset, and model weights will be released after the review process.

REFERENCES

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*, 47(7):5130–5145, 2025. doi: 10.1109/TPAMI.2024.3386695.

Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018a.

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 *2018 IEEE/CVF conference on computer vision and pattern recognition*, pp. 3674–3683. IEEE, 2018b.

Jan Balaguer, Hugo Spiers, Demis Hassabis, and Christopher Summerfield. Neural mechanisms of hierarchical planning in a virtual subway network. *Neuron*, 90(4):893–903, 2016.

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris Coll-Vinent, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, Jie Lei, Tengyu Ma, Baishan Guo, Arpit Kalla, Markus Marks, Joseph Greer, Meng Wang, Peize Sun, Roman Rädle, Triantafyllos Afouras, Effrosyni Mavroudi, Katherine Xu, Tsung-Han Wu, Yu Zhou, Liliane Momeni, RISHI HAZRA, Shuangrui Ding, Sagar Vaze, Francois Porcher, Feng Li, Siyuan Li, Aishwarya Kamath, Ho Kei Cheng, Piotr Dollar, Nikhila Ravi, Kate Saenko, Pengchuan Zhang, and Christoph Feichtenhofer. Sam 3: Segment anything with concepts. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 138846–138923, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/e0982cbc81401df3430ee1ff780dc7a2-Paper-Conference.pdf.

Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from RGB-D data in indoor environments. In *2017 International Conference on 3D Vision*, pp. 667–676. IEEE, 2017.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Zehui Chen, Haodong Duan, Jiaqi Wang, Yu Qiao, Dahua Lin, et al. Are we on the right way for evaluating large vision-language models? *Advances in Neural Information Processing Systems*, 37:27056–27087, 2024a.

Zhenfang Chen, Qinhong Zhou, Yikang Shen, Yining Hong, Zhiqing Sun, Dan Gutfreund, and Chuang Gan. Visual chain-of-thought prompting for knowledge-based visual reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 1254–1262, 2024b.

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*, LosAngeles, CA, USA, June 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, et al. Abot-n0: Technical report on the vla foundation model for versatile embodied navigation. *arXiv preprint arXiv:2602.11598*, 2026.

Russell A Epstein, Eva Zita Patai, Joshua B Julian, and Hugo J Spiers. The cognitive map in humans: spatial navigation and beyond. *Nature neuroscience*, 20(11):1504–1513, 2017.

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 (CVPR)*, pp. 4065–4075, June 2026.

Asher Hancock, Xindi Wu, Lihan Zha, Olga Russakovsky, and Anirudha Majumdar. Actions as language: Fine-tuning vlms into vlas without catastrophic forgetting. In *International Conference on Learning Representations*, volume 2026, pp. 75333–75360, 2026.

Yicong Hong, Zun Wang, Qi Wu, and Stephen Gould. Bridging the gap between learning in discrete and continuous environments for vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15439–15449, June 2022.

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.

Gabriele Janzen and Miranda Van Turennout. Selective neural representation of objects relevant for navigation. *Nature neuroscience*, 7(6):673–677, 2004.

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

Jacob Krantz, Aaron Gokaslan, Dhruv Batra, Stefan Lee, and Oleksandr Maksymets. Waypoint models for instruction-guided navigation in continuous environments. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15162–15171, October 2021.

Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4392–4412, 2020.

Mathieu Labbé and François Michaud. Rtab-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation. *Journal of field robotics*, 36(2):416–446, 2019.

Roberto Leiras, Jared M Cregg, and Ole Kiehn. Brainstem circuits for locomotion. *Annual review of neuroscience*, 45:63–85, 2022.

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 (CVPR)*, pp. 15243–15252, June 2026.

Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, et al. Mvbench: A comprehensive multi-modal video understanding benchmark. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 22195–22206, 2024.

Guoxin Lian, Shuo Wang, Yucheng Wang, Yongcai Wang, Maiyue Chen, Kaihui Wang, Bo Zhang, Zhizhong Su, Deying Li, and Zhaoxin Fan. Mapdream: Task-driven map learning for vision-language navigation. *arXiv preprint arXiv:2602.00222*, 2026.

Bingqian Lin, Yunshuang Nie, Ziming Wei, Jiaqi Chen, Shikui Ma, Jianhua Han, Hang Xu, Xiaojun Chang, and Xiaodan Liang. Navcot: Boosting llm-based vision-and-language navigation via learning disentangled reasoning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(7):5945–5957, 2025.

Bingqian Lin, Yunshuang Nie, Khun Loun Zai, Ziming Wei, Mingfei Han, Rongtao Xu, Minzhe Niu, Jianhua Han, Hanwang Zhang, Liang Lin, Bokui Chen, Cewu Lu, and Xiaodan Liang. Evolvenav: Empowering llm-based vision-language navigation via self-improving embodied reasoning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(8):9782–9796, 2026. doi: 10.1109/TPAMI.2026.3682514.

Yuxing Long, Wenzhe Cai, Hongcheng Wang, Guanqi Zhan, and Hao Dong. Instructnav: Zero-shot system for generic instruction navigation in unexplored environment. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2049–2060. PMLR, 06–09 Nov 2025. URL https://proceedings.mlr.press/v270/long25b.html.

Aftab E Patla and Joan N Vickers. How far ahead do we look when required to step on specific locations in the travel path during locomotion? *Experimental brain research*, 148(1):133–138, 2003.

Zhangyang Qi, Zhixiong Zhang, Yizhou Yu, Jiaqi Wang, and Hengshuang Zhao. Vln-r1: Vision-language navigation via reinforcement fine-tuning. *arXiv preprint arXiv:2506.17221*, 2025.

Yanyuan Qiao, Wenqi Lyu, Hui Wang, Zixu Wang, Zerui Li, Yuan Zhang, Mingkui Tan, and Qi Wu. Open-nav: Exploring zero-shot vision-and-language navigation in continuous environment with open-source llms. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6710–6717. IEEE, 2025.

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

Gabriel Sarch, Snigdha Saha, Naitik Khandelwal, Ayush Jain, Michael Tarr, Aviral Kumar, and Katerina Fragkiadaki. Grounded reinforcement learning for visual reasoning. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 150977–151013. Curran Associates, Inc., 2025. doi: 10.52202/085713-5049. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/ddbd83ac1ad27304a72b873124c2dac2-Paper-Conference.pdf.

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

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

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.

Shuo Wang, Yongcai Wang, Wanting Li, Xudong Cai, Yucheng Wang, Maiyue Chen, kaihui.wang, Zhizhong Su, Deying Li, and Zhaoxin Fan. Aux-think: Exploring reasoning strategies for data-efficient vision-language navigation. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 30892–30916. Curran Associates, Inc., 2025a. doi: 10.52202/085713-1040. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/2c90bf5bffe6497730ecade3a3a37458-Paper-Conference.pdf.

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

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

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025b.

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

Zun Wang, Jialu Li, Yicong Hong, Yi Wang, Qi Wu, Mohit Bansal, Stephen Gould, Hao Tan, and Yu Qiao. Scaling data generation in vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 12009–12020, October 2023b.

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

Meng Wei, Chenyang Wan, Peng 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-language navigation. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 12380–12396, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/14da7aea05debb963b3d8d46449d51a0-Paper-Conference.pdf.

Penghao Wu and Saining Xie. V*: Guided visual search as a core mechanism in multimodal llms. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13084–13094. IEEE, 2024.

Shuang Zeng, Dekang Qi, Xinyuan Chang, Feng Xiong, Shichao Xie, Xiaolong Wu, Shiyi Liang, Mu Xu, and Xing Wei. Janusvln: Decoupling semantics and spatiality with dual implicit memory for vision-language navigation. In *International Conference on Learning Representations*, volume 2026, pp. 33001–33026, 2026.

Jiazhao Zhang, Kunyu Wang, Rongtao Xu, Gengze Zhou, Yicong Hong, Xiaomeng Fang, Qi Wu, Zhizheng Zhang, and He Wang. Navid: Video-based vlm plans the next step for vision-and-language navigation. *arXiv preprint arXiv:2402.15852*, 2024.

Jiazhao Zhang, Kunyu Wang, Shaoan Wang, Minghan Li, Haoran Liu, Songlin Wei, Zhongyuan Wang, Zhizheng Zhang, and He Wang. Uni-navid: A video-based vision-language-action model for unifying embodied navigation tasks. *Robotics: Science and Systems*, 2025a.

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

Junkai Zhang, Yihe Deng, Kai-Wei Chang, and Wei Wang. Thinking with visual grounding. *arXiv preprint arXiv:2606.16122*, 2026b.

Zekai Zhang, Weiye Zhu, Hewei Pan, Xiangchen Wang, Rongtao Xu, Xing Sun, and Feng Zheng. Activevln: Towards active exploration via multi-turn rl in vision-and-language navigation. *arXiv preprint arXiv:2509.12618*, 2025b.

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

Jing Zuo, Lingzhou Mu, Fan Jiang, Chengcheng Ma, Mu Xu, and Yonggang Qi. Fantasyvln: Unified multimodal chain-of-thought reasoning for vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 29674–29683, June 2026.

A Appendix

A.1 Training-Data Efficiency

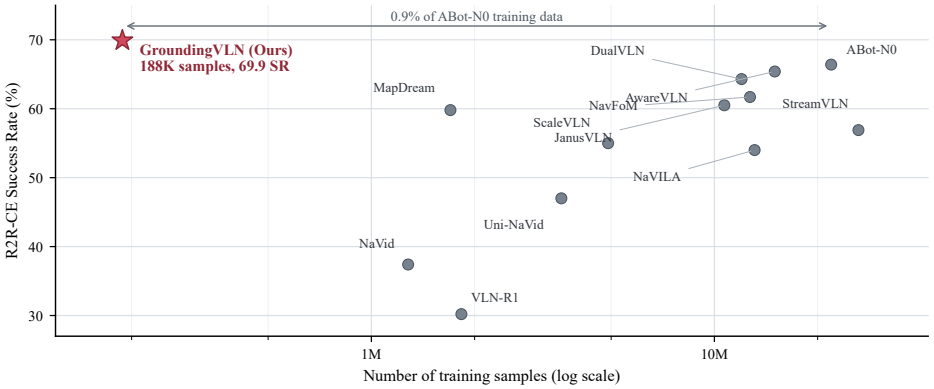


Figure 5: Training-data efficiency on R2R-CE `val_unseen`. The horizontal axis shows each paper's reported training-example count in its native unit on a logarithmic scale; methods without a reported non-zero count are omitted. GroundingVLN achieves the highest SR among the plotted methods with 188K examples, equivalent to 0.9% of ABot-N0's reported total.

We use each paper's reported training count without converting between units or double-counting reused data. Across methods, a reported example may be a trajectory, a clip, or a decision step; a trajectory or clip can contain multiple steps. Our 188K examples are step-level VLM input–output pairs: each pairs the current observation and up to three context images and preceding reasoning turns with one grounded high-level decision. They come from 21K R2R and RxR trajectories and include the 16K GEAR subset. ABot-N0 (Chu et al., 2026) reports 16.9M training trajectories across five navigation paradigms and 5.0M reasoning examples, totaling 21.9M. Its reasoning data include navigability labels, navigation CoTs generated by Qwen-VL-Max, Gemini-3 Pro, and Qwen3-VL, trajectory-verified Object-Goal reasoning, POI-grounding VQA, and public VQA/EQA data.

Figure 5 shows that GroundingVLN reaches the highest R2R-CE `val_unseen` SR among the plotted methods with 188K decision-step examples, or 0.86% (rounded to 0.9%) of ABot-N0's

reported total. Since a training trajectory can contain multiple decision steps, a step-level count for ABot-N0 could be larger and the corresponding proportion smaller than 0.9%.

A.2 Comparison without Additional Training Data

Table 4: Comparison with the benchmark-only and full-data variants of StreamVLN and JanusVLN on the `val_unseen` splits. Benchmark-only variants are trained using R2R and RxR data without additional navigation corpora; their exact sample counts are not reported.

			R2R-CE Val-Unseen			RxR-CE Val-Unseen			
Method	Training corpus	Samples	NE↓	SR↑	SPL↑	NE↓	SR↑	SPL↑	nDTW↑
StreamVLN (Wei et al., 2025)	R2R+RxR only	–	5.98	45.6	42.3	–	–	–	–
StreamVLN (Wei et al., 2025)	Full corpus	∼26.3M	4.98	56.9	51.9	6.22	52.9	46.0	61.9
JanusVLN Base (Zeng et al., 2026)	R2R+RxR only	–	5.17	52.8	49.2	6.46	51.4	44.3	59.1
JanusVLN (Zeng et al., 2026)	Full corpus	10.69M	4.78	60.5	56.8	6.06	56.2	47.5	62.1
GroundingVLN (Ours)	R2R+RxR only	**188K**	**3.66**	**69.9**	**64.1**	**3.54**	**75.1**	**62.0**	**75.3**

Table 4 further controls for the use of external navigation corpora. With only 188K R2R and RxR samples, GroundingVLN achieves absolute SR gains of 24.3% and 17.1% over the benchmark-only variants of StreamVLN and JanusVLN on R2R-CE, respectively, and 23.7% over JanusVLN Base on RxR-CE. More importantly, it remains substantially stronger than their full-data variants: GroundingVLN improves R2R-CE SR by 13.0% over StreamVLN and 9.4% over JanusVLN, and improves RxR-CE SR by 18.9% over JanusVLN, while using only 0.7% and 1.8% of their respective training samples. These results show that the gains arise from more effective grounded supervision and execution alignment rather than from scaling the training corpus.

A.3 General-Capability Retention

To assess whether navigation tuning preserves the pretrained model's general multimodal capabilities, we evaluate zero-shot performance on MMStar (Chen et al., 2024a) and MVBench (Li et al., 2024) under the same deterministic protocol.

Table 5: Zero-shot accuracy (%) on general multimodal benchmarks after navigation tuning.

Model	MMStar	MVBench
Qwen3.5-4B	**63.67**	**50.93**
GroundingVLN	57.20	45.55
AwareVLN (Guo et al., 2026)	0.07	0.00
DualVLN System 2 (Wei et al., 2026)	0.00	0.00

As shown in Table 5, GroundingVLN largely preserves general multimodal competence without mixing general-purpose data into navigation training, unlike the severe specialization often observed in end-to-end VLN agents. Compared with the original Qwen3.5-4B, it shows absolute accuracy drops of only 6.47% on MMStar and 5.38% on MVBench. Moreover, navigation tuning yields absolute accuracy gains of 27.5% on Object Shuffle, 5.0% on Scene Transition, 4.5% on Counterfactual Inference, 3.5% on State Change, and 3.0% on Egocentric Navigation. These gains are concentrated in spatial-change tracking, temporal transition understanding, and egocentric motion reasoning, suggesting that navigation data strengthen capabilities directly relevant to embodied interaction.

For comparison, we evaluate AwareVLN and DualVLN System 2, two VLN models likewise adapted from general-purpose VLM backbones. Despite using prompts adapted to the VQA tasks, both models remain largely unable to provide the required answer choices: AwareVLN continues to emit its navigation protocol, while DualVLN System 2 returns directional actions or `STOP`. This collapse toward a navigation-specific output space is a clear manifestation of catastrophic forgetting

under task-specific fine-tuning. Taken together, GroundingVLN combines state-of-the-art VLN performance with strong retention of its pretrained general capabilities, highlighting its potential as a general-purpose embodied foundation model.

A.4 Inference Efficiency

Table 6: Average inference time per navigation episode. Lower is better.

Method	Time (s/episode)↓
NaVILA Cheng et al. (2025)	56.84
Aux-Think Wang et al. (2025a)	58.04
Progress-Think Wang et al. (2026b)	**36.60**
ActiveVLN Zhang et al. (2025b)	106.90
GroundingVLN (Ours)	37.41

We measure inference time using a single NVIDIA RTX 4090 GPU. As shown in Table 6, GroundingVLN requires 37.41 seconds per episode, comparable to the fastest Progress-Think result of 36.60 seconds, while reducing latency by 34.2%, 35.5%, and 65.0% relative to NaVILA, Aux-Think, and ActiveVLN, respectively. GroundingVLN invokes the VLM only 9.26 times per episode on average, corresponding to 33.25% of navigation steps. The low-level planner handles execution between these sparse high-level decisions, avoiding repeated VLM inference at every primitive action while retaining structured reasoning at key decision points.

A.5 Robustness to Depth Noise

To assess sensitivity to imperfect depth sensing, we evaluate GroundingVLN on R2R-CE `val_unseen` with a disparity-domain noise model configured using the D435 stereo baseline and subpixel error characteristics documented by Intel RealSense. For each finite ground-truth depth Z satisfying $0.17 < Z < 5\,\mathrm{m}$, we compute

$$\begin{aligned} d &= \frac{f_x B}{Z}, \qquad \tilde{d} = d + \epsilon_d, \qquad \epsilon_d \sim \mathcal{N}(0, \sigma_d^2), \\ \hat{d} &= q\ \mathrm{round}(\tilde{d}/q), \quad \hat{Z} = \frac{f_x B}{\hat{d}}, \end{aligned} \tag{4}$$

where f_x is each camera's original horizontal focal length in pixels, $B = 0.05\,\mathrm{m}$ is the stereo baseline, $\sigma_d = 0.08$ pixels, and $q = 1/32$ pixel is the disparity quantization step. Invalid input depths, non-positive disparities, and reconstructed depths that are non-finite or outside the same valid range are set to zero. Out-of-range depths are not clamped to $5\,\mathrm{m}$, avoiding artificial distant surfaces. Ignoring quantization and range filtering, the first-order depth uncertainty grows quadratically with distance, $\sigma_Z \approx Z^2\sigma_d/(f_x B)$. For the front camera with $f_x \approx 277.13$ pixels, this corresponds to approximately 0.58, 5.20, and $14.43\,\mathrm{cm}$ at 1, 3, and near $5\,\mathrm{m}$, respectively.

Table 7: Depth-noise robustness on R2R-CE `val_unseen`. NE is in meters; OS, SR, and SPL are percentages.

Depth input	NE↓	OS↑	SR↑	SPL↑
Ground-truth depth	3.66	74.8	69.9	64.1
D435-like noisy depth	4.14	71.2	65.5	59.7

As shown in Table 7, GroundingVLN retains an SR of 65.5% and an SPL of 59.7% under noisy depth, with an absolute decrease of 4.4% in both metrics relative to ground-truth depth. The moderate degradation suggests that its pixel-goal execution pipeline tolerates the tested distance-dependent measurement errors, although reliable depth remains important for geometric planning.

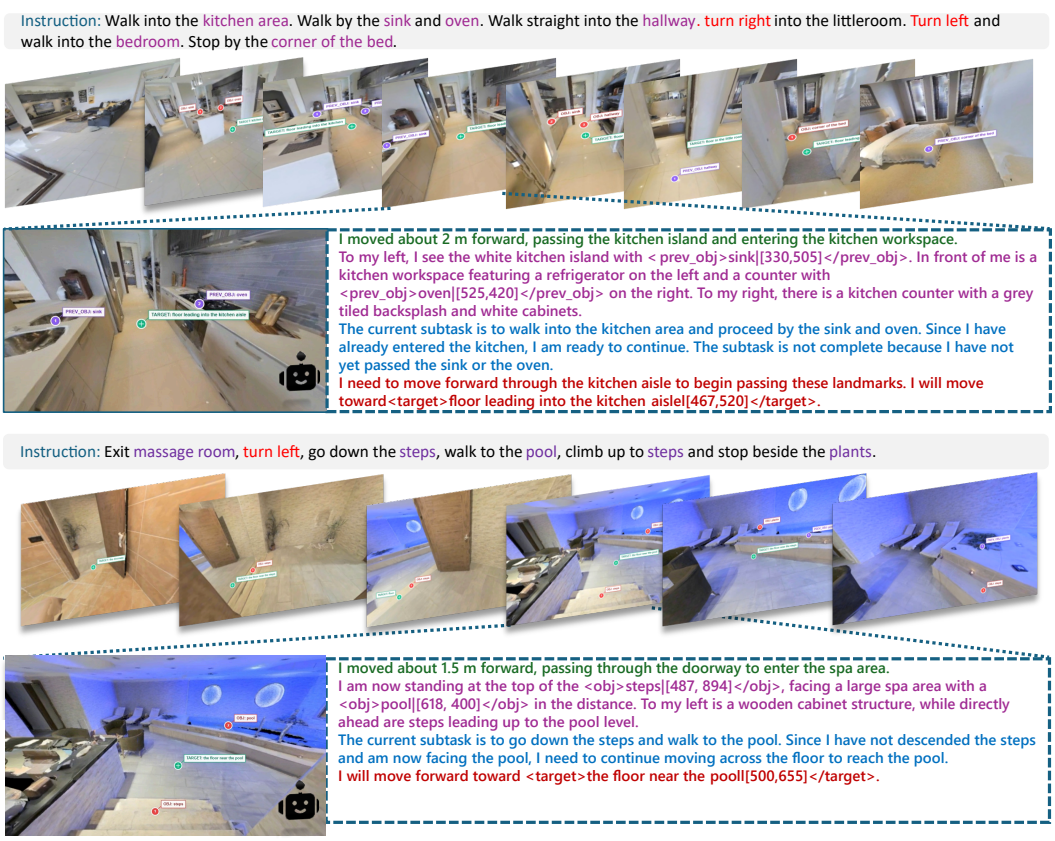


Figure 6: Two qualitative examples of GroundingVLN. The upper example follows a route from the kitchen to the bedroom, while the lower example navigates from a massage room toward a pool and nearby plants. Each example shows the instruction, representative trajectory observations, and an enlarged decision step with grounded reasoning. Newly grounded evidence, recalled evidence, and pixel goals are marked by `<obj>`, `<prev_obj>`, and `<target>`, respectively.

A.6 Qualitative Results

Figure 6 illustrates how grounded reasoning guides navigation in two distinct environments. In the kitchen-to-bedroom example, the model recalls the sink and oven as historical evidence despite viewpoint changes. At the highlighted step, it distinguishes entering the kitchen from passing these landmarks and selects a pixel goal further along the aisle. In the massage-room-to-pool example, the model grounds the steps and pool, recognizes that it has entered the spa area but remains at the top of the steps, and selects a floor target toward the pool while identifying descent and approach as unfinished parts of the subtask. Both examples connect visual evidence and completed motion to the remaining subtask, illustrating how temporal grounding and semantic progress guide local goal selection along longer routes.

A.7 GroundingCOTVLN-188K Data-Engine Details

This section details the automatic data engine introduced in Section 3.4. The pipeline first replays reference trajectories to collect temporally ordered anchors. It then annotates each anchor with its current subtask, progress state, visual observation, and instruction-relevant entities. Finally, it aligns grounded evidence across time, derives action and pixel-goal labels from the executed geometry, and filters invalid examples before export.

Replay and anchor construction. For each episode, the replay stage executes the reference route with the same low-level planner used by the navigation system and caches the *executed* trajectory rather than teleporting between reference poses. The cache additionally retains camera intrinsics, poses, actions, waypoint indices, and the complete left–front–right frame stream required by temporal tracking. Annotation then runs in a strict cache-only mode, which decouples simulator throughput from VLM and SAM 3 inference.

Before replay, dense routes containing more than ten points or an adjacent pair closer than $1\,\mathrm{m}$ are resampled at approximately uniform arc-length intervals. The endpoints and the strongest route turns above 45° are explicitly retained. If merging is unnecessary, only segments longer than $4\,\mathrm{m}$ are interpolated, with a maximum spacing of $3\,\mathrm{m}$. We record arrival anchors at reference points, paired pre-/post-turn anchors for executed in-place rotations of at least 60°, and mid-distance anchors after approximately $3\,\mathrm{m}$ of travel. Candidate anchors triggered by 45° accumulated rotation or 15 low-level steps are used to fill retained trajectory gaps exceeding $4\,\mathrm{m}$. Near-static anchors immediately preceding an arrival or turn are removed. To diversify termination states, the final stopping radius is sampled independently for every episode from $[0, 1]\,\mathrm{m}$.

Subtask, progress, and observation annotation. The teacher first divides the instruction into ordered, exact instruction substrings and associates each subtask with a checklist of two or three landmarks or turns. The decomposition is rejected and retried when a subtask is not an instruction substring, a checklist has an invalid structure, or a turn is unsupported by its subtask. A progress annotator then receives at most 20 sampled front-view frames, the executed action trace, waypoint events, and sparse room hints, and aligns every subtask with a contiguous waypoint interval. Turn boundaries are subsequently repaired using the measured signed rotations, preventing visually plausible but temporally incorrect progress labels.

Observation generation is performed before CoT synthesis. For the current anchor, the teacher sees the unannotated stitched panorama; from the second anchor onward, it also receives the preceding successful-anchor image and a coordinate-free version of the preceding observation. Task-relevant object candidates are selected in batches of at most ten anchors. The front view is processed first, and a full left–front–right pass is invoked only when instruction landmarks remain uncovered. Directed grounding is then performed independently within each view, prioritizing the front panel. Panel-local coordinates are normalized to $[0, 1000]^2$ and converted to the common stitched-image coordinate system. Outputs referring to oracle overlays, reference points, target annotations, or other generation artifacts are rejected and regenerated.

Cross-time grounding alignment. Each newly accepted `<obj>` coordinate is paired with depth and back-projected through the anchor camera pose to create a persistent world-space entity record. Initial estimates farther than $5\,\mathrm{m}$ from the agent are discarded. The corresponding box prompt initializes SAM 3 on the complete per-step replay stream; when only a point is available, the engine constructs a small point-centered box. We use a mask-presence threshold of 0.4 and a minimum mask area of 64 pixels. The representative point is chosen near the mask centroid from pixels with valid depth and is then back-projected to refresh the world position.

Temporal identity is accepted only when the refreshed 3D point is within $2\,\mathrm{m}$ of the previous valid estimate. Tracks with the same normalized label and mask IoU above 0.5 are merged, while broad regions such as *floor*, *room*, and *hallway* are excluded from instance tracking. A SAM 3 miss or a mask without usable depth does not overwrite a valid world position; that position is retained for up to five CoT anchors. Consequently, an entity can leave the image and later be reprojected with the same identity. Such reappearances are encoded as `<prev_obj>`, whereas only newly established evidence uses `<obj>`.

Geometry-owned action and pixel-goal labels. After observation grounding, a deterministic assembler derives progress, actions, and pixel goals from the executed trajectory. For `MOVE`, the trajectory is densified to at most $0.25\,\mathrm{m}$ spacing, and the target is the farthest continuously visible executed point before the next retained anchor and within the current semantic segment. Projection searches for a ground-contact pixel and checks depth agreement, occlusion, front-view membership, and traversability; a `MOVE` anchor is dropped when no genuine target is visible. A `TURN` target is optional and is kept only when its view is compatible with the rotation direction and the teacher confirms a visible semantic cue. `STOP` is emitted only after the final subtask is complete and the

executed pose lies within the replay terminal tolerance. The teacher verbalizes these facts and supplies a concise target label, but post-processing overwrites its action and coordinate fields with the deterministic labels. It also enforces one matching <target> tag whenever a pixel goal is present.

Export and quality control. Every anchor exports a raw panorama, a rendered visual-memory image, and a variant without historical key-object markings. The current response is added to visual and textual memory only *after* its input has been rendered, preventing its supervised labels from leaking into the same sample. Each example contains at most three prior successful reasoning turns and three between-anchor context images.

Before export, the engine requires a successful teacher status, a non-empty structured <think> trace, valid grounding markup, action–reasoning agreement, and a visible target for MOVE; it also rejects annotation leakage and malformed turn supervision. An offline audit revalidates the mixed <think>–JSON schema, agreement between parsed and serialized targets, message–image chains, coordinates, and every assistant response retained in the history. It further checks decomposition substring/checklist constraints and removes all remaining samples from a navigation episode if no valid STOP survives row-level filtering.

Table 8: Key implementation settings of the GroundingCOTVLN-188K data engine.

Setting	Value
Route target budget / minimum spacing	10 / 1 m
Turn preservation / turn-anchor threshold	45° / 60°
Anchor stride / gap-insertion threshold	3 m / 4 m
Progress frames / history turns / context frames	20 / 3 / 3
Key-object annotation batch size	10 anchors
SAM 3 presence / minimum mask area	0.4 / 64 px
World-point range / identity distance	5 m / 2 m
Duplicate-mask IoU / state retention	0.5 / 5 anchors
Executed-trajectory target spacing	0.25 m
Sampled stop threshold / tolerance floor	$[0, 1]$ m / 0.35 m

A.8 Coordinate-Aware Supervised Fine-Tuning

In addition to the token-weighted autoregressive cross-entropy $\mathcal{L}_{\mathrm{wCE}}$, SFT uses a differentiable coordinate objective. At each coordinate digit position k, we restrict the logits to digit tokens and compute the expected digit $\hat{d}_k = \sum_{d=0}^{9} d\, p_k(d)$. The expected coordinate is then reconstructed from its decimal place values v_k as $\hat{c} = \sum_k v_k \hat{d}_k$. For all coordinates c appearing in <obj>, <prev_obj>, <target>, and the action JSON, we optimize

$$\mathcal{L}_{\mathrm{SFT}} = \mathcal{L}_{\mathrm{wCE}} + \lambda_{\mathrm{coord}} \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \mathrm{SmoothL1}_{\delta}\left(\frac{\hat{c}}{s_{\mathrm{coord}}}, \frac{c}{s_{\mathrm{coord}}} \right). \tag{5}$$

A.9 Detailed GEAR Reward Definitions

We describe the reward functions used in our implementation. Let $\bar{a}_t$, $\bar{d}_t$, $\bar{c}_t$, and $\bar{p}_t$ denote the annotated action, turn direction, current subtask, and pixel goal, respectively, and let the corresponding predictions be denoted with hats. For pixel-goal supervision, our implementation uses the execution-aware target maps described below.

Strict output validity. Before assigning a semantic reward, we require exactly one non-empty <think> block followed by one action JSON object and no trailing text. The JSON must contain a non-empty current subtask and an action-specific schema. MOVE requires one target with an integer coordinate in $[0, 1000]^2$, TURN requires a valid direction and may contain one target, and STOP must not contain a target. A target-bearing decision must also contain exactly one matching <target> tag in the reasoning. Malformed, truncated, schema-inconsistent, or coordinate-inconsistent outputs receive

$$R_{\mathrm{invalid}} = -1. \tag{6}$$

Action and subtask rewards. For valid outputs, the action reward is the following deterministic matrix:

$$R_{\text{act}} = \begin{cases} 1, & \hat{a}_t = \bar{a}_t \text{ and, for } \texttt{TURN},\ \hat{d}_t = \bar{d}_t, \\ -0.5, & \bar{a}_t = \hat{a}_t = \texttt{TURN},\ \hat{d}_t \neq \bar{d}_t, \\ -0.75, & \bar{a}_t = \texttt{TURN},\ \hat{a}_t = \texttt{MOVE},\ \hat{p}_t \in \mathcal{Q}_{\bar{d}_t}, \\ -0.9, & (\bar{a}_t, \hat{a}_t) \in \{(\texttt{MOVE}, \texttt{TURN}), (\texttt{TURN}, \texttt{MOVE})\}, \\ -1, & \text{otherwise.} \end{cases} \tag{7}$$

Here $\mathcal{Q}_{\bar{d}_t}$ is the outer quarter of the front panel consistent with a required left or right turn. The -0.9 case for a predicted `MOVE` applies when its target is outside this partially aligned region. Premature `STOP` and any wrong action for a gold `STOP` receive -1. For subtask consistency, we normalize Unicode, lowercase text, collapse whitespace, and remove trailing punctuation, denoted by $\nu(\cdot)$, and use

$$R_{\text{task}} = 2\,\mathbf{1}[\nu(\hat{c}_t) = \nu(\bar{c}_t)] - 1. \tag{8}$$

Execution-aware target quality. For a normalized pixel p, depth unprojection gives a 3D endpoint $P_t(p)$. From the stitched RGB-D observation, we reconstruct a connected local support surface that preserves traversable floors, stairs, and landings while excluding occupied or unsupported regions. We set $V_{\text{nav}}(p) = 1$ when the endpoint is sufficiently close to this support surface, and 0 otherwise.

Let s_t^a be the agent's arc-length position on the reference route, $s(p)$ the route projection of $P_t(p)$, and s_t^* the projection of the annotated endpoint P_t^*. We further define the executed distance $d_a(p) = \|P_t(p) - P_t^a\|_2$, forward progress $\Delta s(p) = s(p) - s_t^a$, lateral route distance $d_\perp(p)$, and endpoint error $d_e(p) = \|P_t(p) - P_t^*\|_2$. For a target at distance $d_t^* = \|P_t^* - P_t^a\|_2$, the near-target tightening factor is

$$\kappa_t = \text{clip}\left(\frac{d_t^*}{2.0}, 0.33, 1.0\right). \tag{9}$$

The score-validity indicator is

$$\begin{aligned} V_{\text{score}}(p) = V_{\text{nav}}(p)\,\mathbf{1}[p \in \mathcal{F}_t]\,\mathbf{1}[d_a(p) \geq \eta_d] \\ \cdot\,\mathbf{1}[\Delta s(p) \geq \eta_s]\,\mathbf{1}[d_\perp(p) \leq \kappa_t \eta_\perp]\,\mathbf{1}[|\Delta h(p)| \leq \eta_h], \end{aligned} \tag{10}$$

where $\mathcal{F}_t$ is the front panel for a gold `MOVE` and the full stitched view for a target-bearing `TURN`, and $\Delta h(p)$ is the endpoint's height residual from the local support surface. The thresholds are $\eta_d = \eta_s = 0.30\,\text{m}$, $\eta_\perp = 1.50\,\text{m}$, and $\eta_h = 0.75\,\text{m}$. For a valid point, the continuous target quality is

$$\begin{aligned} Q_t(p) = 0.25 \exp\left[-\frac{d_\perp(p)^2}{2(0.50\kappa_t)^2}\right] + 0.35 \exp\left[-\frac{(s(p) - s_t^*)^2}{2(0.75)^2}\right] \\ + 0.40 \exp\left[-\frac{d_e(p)^2}{2(1.00\kappa_t)^2}\right]. \end{aligned} \tag{11}$$

Invalid points have $Q_t(p) = 0$.

The normalized image is discretized into a 256×68 map. For the set Ω_j of integer normalized coordinates assigned to map cell j, we store the conservative cell score

$$Q_{t,j} = \min_{p \in \Omega_j} Q_t(p), \qquad V_{t,j} = \mathbf{1}\left[\min_{p \in \Omega_j} V_{\text{score}}(p) = 1\right] \mathbf{1}[Q_{t,j} \geq 0.25]. \tag{12}$$

A five-point center-and-corner stencil first rejects clearly invalid cells, after which every integer coordinate in each provisional positive cell is checked. Separately, the cell-level physical indicator $V_{t,j}^{\text{nav}}$ is true when any stencil point has navigable support, without applying route-progress or front-panel gates; this distinguishes a physical obstacle from a walkable but off-route target. The final quality is quantized to 8 bits. We retain a target-bearing training anchor only when its annotated cell satisfies $V_{t,j^*} = 1$ and $Q_{t,j^*} \geq 0.80$. At reward time, the predicted target receives

$$R_{\text{goal}}(\hat{p}_t) = \begin{cases} 2Q_{t,j(\hat{p}_t)} - 1, & V_{t,j(\hat{p}_t)} = 1, \\ -1, & \text{otherwise.} \end{cases} \tag{13}$$

A missing target or a prediction from a different action family also receives -1. Because multiple target labels may be valid, R_{goal} does not consider exact label agreement.

Grounding reward. Teacher-VLM grounding annotations may contain occasional coordinate errors or omissions, while grounding correctness is strongly associated with decision correctness (Figure 3(c)). To avoid misleading supervision from imperfect annotations, we activate the auxiliary grounding bonus only when the navigation decision is also correct.

Let $\mathcal{L}_t^*$ be the set of normalized teacher grounding labels and $\widehat{\mathcal{G}}_t = \{(\hat{l}_k, \hat{p}_k)\}$ the label–coordinate pairs in predicted `<obj>` and `<prev_obj>` tags, with label set $\widehat{\mathcal{L}}_t$. Once the required labels are present, each matched label l is used as an open-vocabulary prompt to SAM 3 (Carion et al., 2026), which segments all of its visible instances. Let $\mathcal{M}_t(l)$ denote the union of these instance masks and define

$$C_t = \mathbf{1}\Big[\forall l \in \mathcal{L}_t^*,\ \exists(\hat{l}_k, \hat{p}_k) \in \widehat{\mathcal{G}}_t : \hat{l}_k = l \ \wedge\ \hat{p}_k \in \mathcal{M}_t(l)\Big]. \tag{14}$$

We deliberately relax coordinate validation by accepting any point inside a visible instance mask for the matching label, rather than requiring agreement with the teacher's exact coordinates. Define $D_t = 1$ when the predicted decision is correct. For a gold `MOVE`, this requires a predicted `MOVE` whose target is in the front panel and physically navigable; for `TURN` and `STOP`, it requires the correct action and, for `TURN`, the correct direction. A target-bearing `TURN` does not require an accurate target to pass D_t because its target is auxiliary shaping. The grounding bonus is

$$B_{\text{grd}} = 0.10\,\mathbf{1}[D_t = 1]\,\mathbf{1}[\emptyset \neq \mathcal{L}_t^* \subseteq \widehat{\mathcal{L}}_t]\,\mathbf{1}[C_t = 1]\,\mathbf{1}[T_t = 1]\,\mathbf{1}[A_t = 0], \tag{15}$$

where $T_t = 1$ for a target-free annotation, or when the predicted target is score-valid and has $Q \geq 0.50$; $A_t = 1$ if any predicted grounding label contains *move* or *turn*. A missing mask or a matched point outside all corresponding masks prevents the bonus. Extra object labels do not prevent the bonus. They are penalized only in a bad decision context:

$$P_{\text{grd}} = 0.10\,\mathbf{1}\big[(R_{\text{act}} < 1)\ \vee\ (\bar{a}_t = \texttt{MOVE} \wedge \neg V_{t,j(\hat{p}_t)})\big]\,\mathbf{1}\Big[\widehat{\mathcal{L}}_t \setminus \mathcal{L}_t^* \neq \emptyset\Big]. \tag{16}$$

Thus, a missing label or spatially invalid coordinate forfeits the bonus without an additional negative penalty, while an unsupported extra label accompanying an incorrect action or out-of-score `MOVE` is penalized.

Reasoning-consistency penalties. We subtract $P_{\text{motion}} = 0.05$ when the first reasoning sentence omits, cannot express, or contradicts the mechanically recorded recent motion. A correct `STOP` whose reasoning still states that navigation is incomplete receives $P_{\text{stop}} = 0.50$. Finally, let ρ_{cross} be the fraction of unique current 4-grams also present in the previous decision's reasoning, after masking markup. If $\rho_{\text{cross}} \geq 0.85$, the final reward is overridden to -1.

Action-conditional composition. Let H_t denote a gold-`MOVE`, same-family prediction whose target is either physically non-navigable or outside the front panel. For all other valid outputs, the base reward is

$$\widetilde{R}_t = \begin{cases} 0.30R_{\text{act}} + 0.20R_{\text{task}} + 0.50R_{\text{goal}}, & \bar{a}_t = \texttt{MOVE}, \\ 0.65R_{\text{act}} + 0.20R_{\text{task}} + 0.15\widetilde{R}_{\text{goal}}, & \bar{a}_t = \texttt{TURN} \text{ with target}, \\ 0.80R_{\text{act}} + 0.20R_{\text{task}}, & \bar{a}_t = \texttt{TURN} \text{ without target, or } \bar{a}_t = \texttt{STOP}, \end{cases} \tag{17}$$

where $\widetilde{R}_{\text{goal}} = 0$ for a same-family `TURN` with the wrong direction; otherwise it equals Equation 13, including -1 for an ineligible target. The complete scalar reward is

$$R_t = \begin{cases} -1, & \text{invalid output or } \rho_{\text{cross}} \geq 0.85, \\ \text{clip}_{[-1,1.1]}(-0.50 - P_{\text{grd}}), & H_t, \\ \text{clip}_{[-1,1.1]}(\widetilde{R}_t + B_{\text{grd}} - P_{\text{motion}} - P_{\text{stop}} - P_{\text{grd}}), & \text{otherwise}. \end{cases} \tag{18}$$

This ordering preserves the absolute -0.50 penalty for a physically invalid or side-view `MOVE`, while allowing the grounding penalty and the final cross-step-copy override to remain effective.

A.10 Group-Relative Optimization Details

Following GRPO (Shao et al., 2024), for each anchor x, we sample G outputs $\{y_i\}_{i=1}^G$ with rewards $\{r_i\}_{i=1}^G$. Their group-relative advantages are

$$A_i = w_x \frac{r_i - \bar{r}_x}{\max(\sigma_x, \epsilon_A)}, \tag{19}$$

Table 9: GEAR optimization hyperparameters.

Parameter	Value
Sampled outputs per anchor G	8
Advantage standard-deviation floor ϵ_A	0.05
Low-variance group threshold τ_{var}	0.02
All-wrong group weight w_{wrong}	0.40
Policy-ratio clipping range ϵ	0.20
Sampled-token KL coefficient β	0.02
Learning rate	1×10^{-6}
Training steps	2,000
`MOVE`/`TURN`/`STOP` sampling ratio	0.50/0.30/0.20

where $\bar{r}_x$ and σ_x are the group mean and standard deviation. We set $w_x = 0$ for groups in which every output is invalid or σ_x falls below a variance threshold; groups containing valid reward variation but no correct decision receive a reduced weight w_{wrong}, and all remaining groups use $w_x = 1$.

Let $y_{i,j}$ be the j-th token of output y_i and define the token-level importance ratio $\rho_{i,j}(\theta) = \pi_\theta(y_{i,j} \mid x, y_{i,<j}) / \pi_{\theta_{\text{old}}}(y_{i,j} \mid x, y_{i,<j})$. To regularize the policy without evaluating the full vocabulary distribution, we use a sampled-token KL proxy. Defining $\Delta_{i,j} = \log \pi_{\text{SFT}}(y_{i,j} \mid x, y_{i,<j}) - \log \pi_\theta(y_{i,j} \mid x, y_{i,<j})$, the proxy is

$$\widehat{D}_{i,j}^{\text{KL}} = \exp(\Delta_{i,j}) - \Delta_{i,j} - 1. \tag{20}$$

The complete GEAR optimization objective is

$$\begin{aligned} \ell_{i,j}^{\text{clip}}(\theta) &= \min\Big(\rho_{i,j}(\theta)A_i, \\ &\qquad \text{clip}(\rho_{i,j}(\theta), 1-\epsilon, 1+\epsilon)A_i\Big), \\ \mathcal{L}_{\text{GEAR}}(\theta) &= -\mathbb{E}_{x,\{y_i\}\sim\pi_{\theta_{\text{old}}}}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|y_i|}\sum_{j=1}^{|y_i|}\ell_{i,j}^{\text{clip}}(\theta)\right] \\ &\quad + \beta\,\mathbb{E}_{x,\{y_i\},j}\left[\widehat{D}_{i,j}^{\text{KL}}\right]. \end{aligned} \tag{21}$$

where ϵ is the policy-ratio clipping range, β controls sampled-token KL regularization, and π_{SFT} is the frozen SFT reference policy. Table 9 lists the optimization parameters used in all experiments.

A.11 Limitations

Although the high-level VLM in GroundingVLN operates solely on RGB observations, the low-level execution module currently relies on sensor-provided metric depth to translate pixel goals into precise and reliable actions. Without training an additional vision-only point-goal planner, depth is required for 2D-to-3D back-projection and RGB-D SLAM-based pose and map estimation. This dependency limits deployment on platforms without reliable depth sensing, motivating future work on an RGB-only low-level planner that does not rely on metric depth while preserving geometric accuracy.