RG-VLN: BENCHMARKING UNDERSPECIFIED REQUEST GROUNDING IN VISION-LANGUAGE NAVIGATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language navigation (VLN) and embodied navigation benchmarks span specified goals, route instructions, target descriptions, and user intentions. In everyday interaction, users may express an underspecified request that combines intent, multiple entities, and partial spatial cues while providing incomplete navigation information. Fulfilling such requests jointly requires target grounding, spatial reasoning, and route planning from the request and online observations. We formulate this complementary setting as **Request Grounding VLN**. To bridge this research gap, we introduce **RG-VLN**, a benchmark with over 6,000 episodes across MP3D and HM3D scenes, with both VLM-generated and human-written requests. RG-VLN associates trajectories with detailed route instructions, route-preserving paraphrases, and underspecified requests containing progressively less route information, thereby enabling controlled analysis of navigation specification. Replacing detailed instructions with underspecified requests reduces the success rates of representative VLN methods by 68.5%–74.2%, whereas route-preserving paraphrases cause only minor changes. Performance is also consistently lower for requests with lower navigation-detail density, indicating that reduced navigation specification is a major source of difficulty. Building on these findings, we further propose **HiNav**, a hierarchical framework that connects request reasoning with navigation execution through closed-loop, scene-grounded textual guidance. This interface resolves request ambiguity from observations and enables existing VLN policies to serve as plug-and-play executors. It achieves the highest success rate across all RG-VLN splits, outperforming the strongest baseline by 7.2%–21.2% relatively. Code and datasets will be publicly released.

1 INTRODUCTION

Embodied navigation requires agents to interpret task specifications and act in complex environments. Existing embodied navigation tasks differ substantially in how navigation objectives are communicated, spanning goal-specified navigation, route-oriented VLN, target-oriented navigation, and intent-driven navigation (Yadav et al., 2023; Anderson et al., 2018; Qi et al., 2020; Zhu et al., 2021; Wang et al., 2023; Qian et al., 2026; Wang et al., 2025c). Among them, vision-language navigation (VLN) studies navigation from natural-language inputs, ranging from detailed route instructions to increasingly high-level language guidance.

However, everyday users may issue underspecified requests that combine intent, multiple entities, and partial spatial cues without providing complete navigation information. As illustrated in Fig. 1, a user may ask, "Could you please check if I left my phone on the sofa? It is on your left," rather than describing every action leading to the destination. Such a request provides useful semantic and spatial cues, yet the intended target must still be grounded in the scene and the route must be inferred from online observations as the agent acquires new visual evidence during navigation.

We formulate this setting as **Request Grounding VLN**, where an agent must jointly perform target grounding, spatial reasoning, and route planning from an underspecified everyday request and online visual observations. We further instantiate this setting in continuous environments, where agents navigate from egocentric observations using primitive actions rather than selecting among

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.

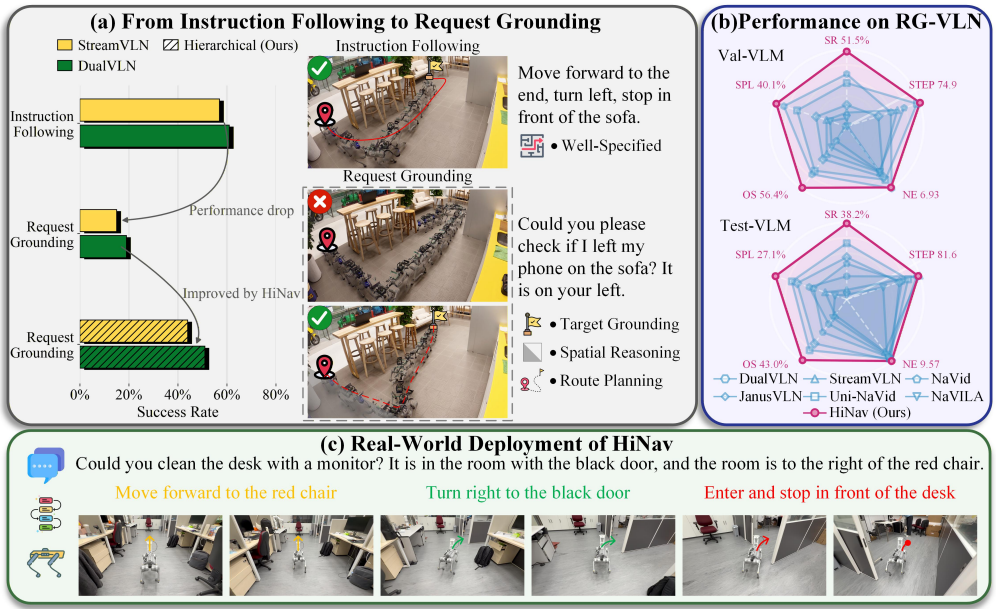


Figure 1: Overview of Request Grounding VLN. (a) From instruction following to request grounding: the illustrated shift from a detailed route instruction to an underspecified everyday request requires target grounding, spatial reasoning, and route planning from incomplete navigation information. Motivated by this gap, we construct the RG-VLN benchmark. The evaluated VLN methods suffer significant success-rate drops when transferred to RG-VLN, whereas HiNav improves their performance through closed-loop textual guidance. (b) HiNav achieves the highest SR and OSR across the MP3D-Val-VLM and HM3D-Test-VLM splits of RG-VLN. (c) Real-world experiments show that HiNav can ground requests and execute navigation on a physical robot.

predefined navigable viewpoints. To make this setting measurable at scale, we introduce **RG-VLN**, a benchmark spanning MP3D (Chang et al., 2017) and HM3D (Ramakrishnan et al., 2021), with approximately 5,000 validation episodes, more than 1,000 test episodes, and over 200 episodes with independently human-written requests. For matched trajectories, RG-VLN pairs detailed route instructions, route-preserving paraphrases, and underspecified requests, enabling controlled comparison while keeping the scene, trajectory, and target fixed. Matched-trajectory experiments show that route-preserving paraphrases cause only minor performance changes, whereas replacing detailed instructions with underspecified requests reduces the success rates of representative VLN methods by 68.5%–74.2%. Performance is also lower for requests with lower navigation-detail density, while human evaluation shows that VLM-generated requests closely align with human-written ones. Together, these results indicate that reduced navigation specification, rather than superficial linguistic variation, is a major source of difficulty.

To address this challenge, we further propose **HiNav**, a hierarchical framework that connects request reasoning with navigation execution through closed-loop, scene-grounded textual guidance. A Request Reasoner resolves target ambiguity from selected observations and generates short-horizon textual subtasks that can be directly followed by a VLN Executor. This explicit language interface enables plug-and-play reuse of different executors, yielding relative SR gains of 199.3% and 162.7% with frozen StreamVLN (Wei et al., 2026b) and DualVLN (Wei et al., 2026a), respectively. Across RG-VLN, HiNav achieves the highest success rate on all splits, outperforming the strongest baseline by 7.2%–21.2% relatively. We further demonstrate zero-shot deployment on a physical robot.

Our main contributions are summarized as follows:

- We formulate Request Grounding VLN, a navigation setting that jointly requires target grounding, spatial reasoning, and route planning from underspecified everyday requests with incomplete navigation information.

- We introduce RG-VLN, containing over 6,000 episodes across MP3D and HM3D with multiple trajectory and instruction sources.
- Through matched-trajectory intervention and request-detail density analysis, we show that reduced navigation specification poses a substantial challenge to existing VLN methods.
- We propose HiNav, which connects request reasoning and navigation execution through closed-loop, scene-grounded textual subtask guidance, enabling plug-and-play reuse of existing VLN executors.

2 Related Work

2.1 Task Specifications in Embodied Navigation

Embodied navigation tasks differ in how goals and navigation information are specified. Goal-specified navigation provides the goal directly: PointNav supplies a geometric destination (Cai et al., 2025; Peng et al., 2025), while ObjectNav specifies a semantic category and requires exploration to find a suitable instance (Yadav et al., 2023; Yokoyama et al., 2024). Route-oriented VLN, including R2R and VLN-CE, uses language primarily to guide the agent along a route (Anderson et al., 2018; Krantz et al., 2020). ULN extends this setting to reduced or coarsened route instructions (Feng et al., 2022). In contrast, RG-VLN shifts the communicative form from route-oriented instructions to everyday task requests, where navigation information remains incomplete. Target-oriented navigation, exemplified by REVERIE (Qi et al., 2020) and SOON (Zhu et al., 2021), uses high-level descriptions to identify a target or scene referent. Agents must interpret these descriptions, explore the environment, and ground the target without step-by-step route guidance. Here, language primarily serves to identify the target referent, whereas RG-VLN starts from an everyday request in which the intended target must be grounded while only partial spatial guidance is available. At the execution level, REVERIE and SOON operate on predefined navigation graphs, whereas RG-VLN uses egocentric observations and primitive actions in continuous environments.

Intent-driven navigation uses language to express a human need, intention, or request. DDN (Wang et al., 2023) and IntentionNav (Qian et al., 2026) primarily leave the semantic object goal unspecified: they infer a suitable object goal from an implicit need or intention and then search for an instance in the scene. NavRAG (Wang et al., 2025c) also starts from user demands, but uses retrieved scene context to construct informative navigation instructions. RG-VLN is closest to this line of work, but differs in what remains unspecified. Rather than reducing the request to an inferred object goal or enriching it into a more complete navigation instruction, RG-VLN evaluates navigation directly from an underspecified everyday request with incomplete navigation information. The target must be grounded from the request and scene, while partial spatial cues must be combined with online observations to plan the route. Thus, RG-VLN jointly requires target grounding, spatial reasoning, and route planning.

2.2 VLN Reasoning and Execution

Recent VLN methods increasingly use vision-language models (VLMs) for zero-shot instruction interpretation (Zhou et al., 2024), spatial reasoning (Yin et al., 2025; Jiang & Wang, 2025), and planning (Chen et al., 2024; Qiao et al., 2025). Other methods further fine-tune VLMs on navigation trajectories to predict actions directly (Zhang et al., 2024; 2025b; Cheng et al., 2025; Zhang et al., 2025a; Wei et al., 2026b; Wang et al., 2025a; Zeng et al., 2026), yielding strong end-to-end executors for specified goals and routes. Recent dual-system methods further decouple high-level reasoning from low-level execution. DualVLN (Wei et al., 2026a) grounds intermediate pixel goals with a VLM and executes them using a diffusion policy, while TIC-VLA (Huang et al., 2026) connects a reasoning VLM and an action expert through latent semantic features. SeqWalker (Han et al., 2026) dynamically selects sub-instructions from a specified long instruction for low-level execution. In contrast, HiNav generates scene-grounded textual subtasks from an underspecified request and online visual history, which serve as an explicit interface to pretrained VLN executors.

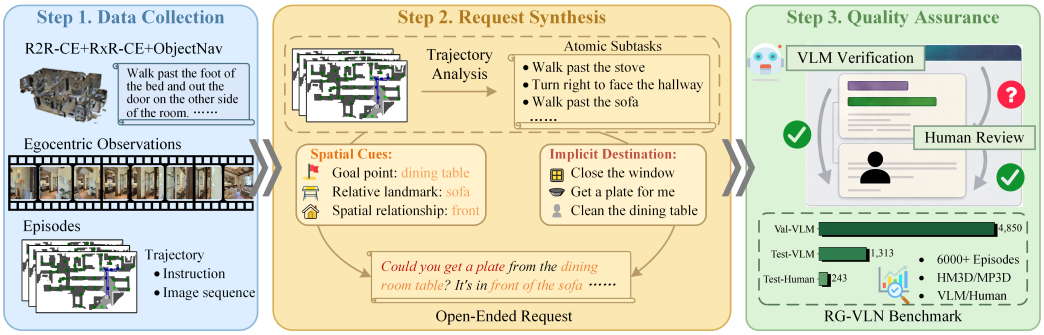


Figure 2: RG-VLN construction pipeline. Diverse trajectories are converted into underspecified requests through route decomposition, spatial grounding, and intent abstraction, followed by automatic verification and human review.

3 RG-VLN Benchmark

RG-VLN makes Request Grounding VLN measurable at scale while preserving diversity and quality. It combines diverse trajectories, scenes, and request sources. To reduce annotation cost, we use a VLM-assisted construction pipeline (Fig. 2) tailored to RG-VLN: it converts trajectories into underspecified requests through route decomposition, spatial grounding, and intent abstraction, followed by automatic verification and targeted human review. Human evaluation in Fig. 3 shows that VLM-generated requests are closely aligned with human-written requests.

3.1 Task Definition

In Request Grounding VLN, an agent is given an underspecified everyday request I, which the benchmark associates with a specific endpoint in the current scene. At step t, the agent observes a monocular RGB image o_t and predicts a navigation action according to

$$\pi : (I, o_{1:t}) \longmapsto a_t \in \mathcal{A}, \quad \tau_\pi = (a_1, \ldots, a_T), \tag{1}$$

where $\mathcal{A} = \{\text{Move Forward, Turn Left, Turn Right, Stop}\}$ and the final action a_T is Stop. An episode is successful when the agent stops within the prescribed distance of the request-grounded endpoint. The request I expresses an everyday intent and may mention multiple entities or partial spatial relations, but does not provide a complete route. Its semantic destination may be apparent or expressed indirectly; jointly performing target grounding, spatial reasoning, and route planning from I and online observations is required in either case.

3.2 Construction Pipeline

Construction comprises three stages: data collection, request synthesis, and quality assurance. We use Qwen3-VL-235B-A22B (Bai et al., 2025a) for VLM-based synthesis and verification.

Data Collection. We replay reference trajectories in Habitat (Savva et al., 2019) with MP3D and HM3D scenes. MP3D trajectories are selected from the validation-unseen splits of R2R-CE and RxR-CE within VLN-CE (Krantz et al., 2020) and retain their paired route instructions. HM3D trajectories are independently generated by a shortest-path follower from ObjectNav (Yadav et al., 2023) start and goal locations. For each episode, we record egocentric RGB observations and actions. Following NaVILA (Cheng et al., 2025), consecutive identical actions are merged to produce a compact visual sequence for request generation.

Request Synthesis. Directly prompting a VLM to write a request from raw image sequences can introduce hallucinated landmarks or unstable goals. To improve grounding, each trajectory is first decomposed into grounded atomic subtasks that summarize local movement and landmark relations. The VLM then analyzes these subtasks together with the visual context to synthesize natural human requests that remain grounded to the benchmark endpoint, preserve key spatial relations, and avoid complete action-level route descriptions. Multiple request variants are generated for each trajectory to improve linguistic diversity while keeping the endpoint fixed.

Algorithm 1 Closed-Loop Inference in HiNav

Input: Request I; CLIP encoders f_I, f_T; Request Reasoner R_θ; Executor E_ϕ; guidance interval H_r; executor refresh interval H_e; action horizon H_a; similarity threshold τ
Output: Action trajectory $\mathcal{A}$

Observe o_1; $t \leftarrow 1$, $k \leftarrow 0$, $h_t \leftarrow \emptyset$, $\mathcal{A} \leftarrow \emptyset$, $B \leftarrow \emptyset$
while the episode is not terminated **do**
  if $t = 1$ **or** $(t-1) \bmod H_r = 0$ **then**
    $c_i \leftarrow \cos(f_I(o_{i-1}), f_I(o_i))$, for $2 \le i \le t$
    $q_i \leftarrow \cos(f_I(o_i), f_T(I))$, for $1 \le i \le t$
    $\bar{c}_{2:t} \leftarrow \mathrm{Normalize}(c_{2:t})$; $\mathcal{S}_t \leftarrow \mathrm{Split}(o_{1:t}; \bar{c}_i < \tau)$
    $m_S \leftarrow \max(1, \mathrm{round}(|S| N_{\max}/t))$, $S \in \mathcal{S}_t$
    $\mathcal{C}_t \leftarrow \bigcup_{S \in \mathcal{S}_t} \mathrm{Top}_{m_S}(S; q_i)$
    Keep the top-$N_{\max}$ frames in $\mathcal{C}_t$ by q_i
    $M_t \leftarrow$ kept frames in temporal order
    $k \leftarrow k + 1$; $g_k \leftarrow R_\theta(I, o_t, M_t)$
  end if
  if $t = 1$ **or** $(t-1) \bmod H_e = 0$ **then**
    $B \leftarrow \mathrm{Queue}(E_\phi(g_k, o_t, h_t; H_a))$
  end if
  $a_t \leftarrow \mathrm{PopFront}(B)$; execute a_t and observe o_{t+1}
  $h_{t+1} \leftarrow \mathrm{Update}(h_t, o_t, a_t)$; $\mathcal{A} \leftarrow \mathcal{A} \,\|\, a_t$; $t \leftarrow t + 1$
end while
return $\mathcal{A}$

Quality Assurance. We use a two-stage quality assurance process to maintain request quality while minimizing manual effort. First, generated requests are iteratively verified by the VLM for target consistency, spatial correctness, and trajectory compatibility. If inconsistencies are detected, the VLM revises the request and re-evaluates the output for up to three cycles. Second, flagged samples are independently reviewed by two human annotators, with each correction or rejection cross-checked. In total, 16.7% of generated samples enter human review; 21.4% of the flagged samples are corrected, and the remainder are removed. This process retains requests with consistent endpoints and valid spatial cues.

3.3 Benchmark Splits

RG-VLN has three complementary splits with different scene, trajectory, and request sources.

MP3D-Val-VLM (4,850 episodes). VLM-generated requests are grounded to VLN-CE trajectories in unseen MP3D scenes. Because each request is paired with the original route instruction on the same scene, endpoint, and path, this split supports controlled matched-trajectory comparisons.

HM3D-Test-VLM (1,313 episodes). VLM-generated requests are grounded to HM3D trajectories from ObjectNav tasks. This split evaluates scene and trajectory transfer beyond VLN-CE paths.

HM3D-Test-Human (243 episodes). For each trajectory sample from the HM3D-Test, three human annotators independently write one request after viewing the trajectory image sequence, implying the unique endpoint through natural intent and spatial cues. This split evaluates generalization to independently written language. More dataset statistics are provided in the supplementary material.

4 HiNav: Hierarchical Navigation

HiNav addresses Request Grounding VLN through an explicit textual interface between a Request Reasoner and a VLN Executor. The Reasoner converts an underspecified request and accumulated visual evidence into short-horizon, scene-grounded textual guidance, which is refreshed in closed loop as new observations arrive. Because the interface is expressed in natural language, existing instruction-following VLN policies can be reused as plug-and-play executors without modifying their core architectures. A dual-similarity history grounding module further selects request-relevant observations for the Reasoner.

Table 1: Performance across request-detail densities on MP3D-Val-VLM. Lower density indicates less linguistic guidance per unit geodesic distance and is used as a proxy for available navigation information. SR, SPL, and OSR are reported in percentage points, and NE in meters. Within each group, the best results are **bold** and second-best underlined.

	High Density				Medium Density				Low Density			
Method	**SR↑**	**SPL↑**	**OSR↑**	**NE↓**	**SR↑**	**SPL↑**	**OSR↑**	**NE↓**	**SR↑**	**SPL↑**	**OSR↑**	**NE↓**
NaVid (Zhang et al., 2024)	30.9	24.0	46.5	6.02	21.7	18.5	31.4	7.77	13.9	11.9	19.1	11.9
Uni-NaVid (Zhang et al., 2025a)	50.0	41.2	62.8	5.33	42.1	37.1	49.9	6.84	27.8	24.3	32.7	10.5
StreamVLN (Wei et al., 2026b)	34.0	28.4	52.0	6.40	30.6	27.2	40.9	7.47	18.1	16.5	24.0	11.8
NaVILA (Cheng et al., 2025)	33.2	29.2	47.4	6.29	26.2	23.9	35.6	7.79	15.8	14.5	21.7	12.0
JanusVLN (Zeng et al., 2026)	40.0	35.6	55.1	5.78	29.6	26.9	40.9	7.55	20.8	18.8	27.6	11.0
DualVLN (Wei et al., 2026a)	52.9	**45.6**	64.0	**4.42**	46.0	41.7	52.1	**5.74**	28.7	26.0	31.6	10.0
HiNav (Ours)	**62.9**	45.1	**69.5**	4.65	**54.1**	**43.9**	**59.2**	6.16	**37.4**	**31.4**	**40.4**	**9.97**

4.1 Closed-Loop Inference Pipeline

At inference time, HiNav alternates between request reasoning and action execution through scene-grounded textual guidance. As summarized in Algorithm 1, a slow Request Reasoner infers the destination and missing route cues, while a fast VLN Executor translates the generated guidance into primitive actions.

At replanning round k, which begins at navigation step t, HiNav first selects a compact visual memory M_t and then generates the next subtask guidance g_k:

$$M_t = K(I, o_{1:t}), \qquad g_k = R_\theta(I, o_t, M_t), \tag{2}$$

where I is the original request, K is the history-selection function, and R_θ is the Request Reasoner. A VLN Executor E_ϕ receives g_k as its language instruction and predicts a short action sequence:

$$a_{t:t+H_a-1} = E_\phi(g_k, o_t, h_t), \tag{3}$$

where h_t denotes the executor-specific navigation history and H_a is its action-prediction horizon. The guidance describes the next action, spatial relation, and landmark rather than a full episode plan. New observations update the history after execution, allowing the Reasoner to revise its guidance when landmarks appear or the trajectory deviates. By default, the Reasoner updates every four executed steps, while Uni-NaVid predicts four actions and refreshes them every two steps following its original strategy (Zhang et al., 2025a).

Because g_k is natural language, the same Reasoner can guide different instruction-following policies without architecture changes or parameter updates. We instantiate the Reasoner with Qwen3-VL-8B (Bai et al., 2025a) and use Uni-NaVid (Zhang et al., 2025a) as the default Executor.

4.2 Dual-Similarity History Grounding

The Request Reasoner needs historical evidence without processing the full stream. Dual-Similarity History Grounding (DS-HG) constructs a bounded memory M_t that combines temporal coverage with request relevance. Let f_I and f_T denote the image and text encoders of a pretrained CLIP model (Radford et al., 2021). For each observation o_i, adjacent-frame similarity $c_i = \cos(f_I(o_{i-1}), f_I(o_i))$ captures visual continuity, while request relevance $q_i = \cos(f_I(o_i), f_T(I))$ identifies task-relevant evidence. As detailed in Algorithm 1, DS-HG segments history using normalized $\bar{c}_i$, allocates frames in proportion to segment length, and retains the most relevant candidates under the $N_{\max}$-frame budget. The resulting memory preserves temporal coverage and prioritizes request-relevant observations. We set $N_{\min} = 8$ for target coverage and $N_{\max} = 36$ for memory capacity.

4.3 Training

Training the Reasoner and Executor requires temporally aligned subtask descriptions and actions, which are not available in existing VLN training data. We synthesize this supervision from R2R-CE training trajectories (Krantz et al., 2020): each underspecified request serves as I, while aligned

Table 2: Overall performance on RG-VLN. SR, SPL, and OSR are reported in percentage points, and NE in meters. Within each split, the best results are **bold** and second-best underlined.

	MP3D-Val-VLM				HM3D-Test-VLM				HM3D-Test-Human			
Method	**SR↑**	**SPL↑**	**OSR↑**	**NE↓**	**SR↑**	**SPL↑**	**OSR↑**	**NE↓**	**SR↑**	**SPL↑**	**OSR↑**	**NE↓**
NaVid (Zhang et al., 2024)	22.2	18.1	32.3	8.55	27.7	19.5	33.2	9.62	35.4	25.0	44.4	8.28
Uni-NaVid (Zhang et al., 2025a)	40.0	34.2	48.5	7.55	33.2	24.1	40.1	9.51	40.3	29.5	49.4	8.43
StreamVLN (Wei et al., 2026b)	27.6	24.0	39.0	8.55	19.0	16.5	25.4	10.9	32.9	27.9	44.4	9.34
NaVILA (Cheng et al., 2025)	25.1	22.6	34.9	8.70	17.2	14.0	26.0	10.5	33.3	27.9	42.0	8.80
JanusVLN (Zeng et al., 2026)	30.1	27.1	41.2	8.12	29.1	23.0	37.2	**9.44**	39.9	**33.0**	49.0	**7.97**
DualVLN (Wei et al., 2026a)	42.5	37.8	49.2	**6.73**	23.6	20.6	28.9	10.3	30.5	26.2	43.6	9.48
HiNav (Ours)	**51.5**	**40.1**	**56.4**	6.93	**38.2**	**27.1**	**43.0**	9.57	**43.2**	29.8	**53.1**	8.42

subtasks and actions provide prediction targets. Training scenes are disjoint from all RG-VLN evaluation scenes, and no intermediate annotations from evaluation episodes are used. The training set contains 9,927 episodes and approximately 420K image–action pairs. The Reasoner and the default Executor are fine-tuned independently as next-token prediction models. At a subtask boundary t, the Reasoner receives (I, o_t, M_t) and predicts g_k using only observations available up to t, while the Executor receives (g_k, o_t, h_t) and predicts the next four actions. Details of dataset construction and optimization are provided in the supplementary material.

5 Experiments

In this section, we first diagnose why request grounding is difficult through request-detail density analysis and matched-trajectory interventions. Next, we evaluate whether HiNav closes this gap across all three splits, and validate the quality of VLM-generated requests through human evaluation. Finally, we analyze HiNav's design choices, including plug-and-play executor transfer, hierarchical learning, history grounding, and zero-shot real-world deployment. Additional analyses are provided in the supplementary material.

Table 3: Matched request intervention on a subset of MP3D-Val-VLM. The three language-input columns evaluate the same episodes with different levels of navigation specification. All values are SR in percentage points.

Method	**VLN**	**Paraphrase**	**RG-VLN**
StreamVLN	57.0	55.0	14.7
DualVLN	61.3	60.7	19.3

5.1 Experimental Setup

Metrics. We report Success Rate (SR) and Success weighted by Path Length (SPL) as the primary metrics. An episode is successful if the agent stops within 3 m of the target. We additionally report Oracle Success Rate (OSR), Navigation Error (NE), and the number of executed steps (STEP).

Baselines. We compare HiNav with NaVid (Zhang et al., 2024), Uni-NaVid (Zhang et al., 2025a), StreamVLN (Wei et al., 2026b), NaVILA (Cheng et al., 2025), JanusVLN (Zeng et al., 2026), and the dual-system method DualVLN (Wei et al., 2026a). Details are provided in the appendix.

5.2 Diagnosing the Difficulty of Request Grounding

Performance across Request-Detail Density. We first examine how performance varies with the navigation information available in a request. For request I and reference trajectory τ, we define request-detail density as $D(I, \tau) = |I|/L(\tau)$, where $|I|$ is the request length and $L(\tau)$ is the geodesic path length. We partition MP3D-Val-VLM episodes into three equal-sized groups, using density as a proxy for the amount of linguistic navigation guidance available per unit distance. Table 1 reveals a consistent relationship between request detail and navigation performance. Across all seven methods, SR decreases by 15.9–25.5 points from the high-density to the low-density group.

Matched Request Intervention. On a fixed subset of MP3D-Val-VLM, we evaluate two representative methods with three language inputs grounded to the same reference path: the original VLN route instruction, a route-preserving paraphrase, and the RG-VLN request. Following the VLM-assisted

procedure described in the benchmark construction, the paraphrase is prompted to change surface wording while preserving route information. As reported in Table 3, route-preserving paraphrases change SR by only 1.0%–3.5%, while RG-VLN requests reduce SR by 68.5%–74.2%. Together with the density trend, this controlled comparison supports reduced navigation specification, rather than surface wording alone, as a major source of difficulty in RG-VLN.

5.3 Overall Performance

We next evaluate the overall system performance on RG-VLN. As shown in Table 2, HiNav achieves the highest SR and OSR on all three RG-VLN splits, exceeding the strongest SR baseline by 21.2%, 15.1%, and 7.2% on MP3D-Val-VLM, HM3D-Test-VLM, and HM3D-Test-Human, respectively. The three splits establish complementary strengths: performance at scale, transfer across scene and trajectory sources, and generalization to independently written language. Fig. 4 illustrates how HiNav grounds an open-ended request into scene-grounded subtasks and reaches the target.

Table 4: Transfer across frozen VLN Executors. HiNav supplies subtask guidance to each frozen executor through the same textual interface. SR, SPL, and OSR are reported in percentage points, and NE in meters.

Executor	SR↑	SPL↑	OSR↑	NE↓
StreamVLN	14.7	12.6	28.7	8.77
+HiNav (Ours)	44.0	40.5	51.0	5.04
DualVLN	19.3	16.5	29.3	7.11
+HiNav (Ours)	50.7	46.6	54.7	4.36

Table 5: Analysis of the hierarchical learning on R2R-derived MP3D evaluation set. All fine-tuning variants use the same limited training set. SR, SPL, and OSR are reported in percentage points, and NE in meters. Best results are **bold** and second-best underlined.

Method	SR↑	SPL↑	OSR↑	NE↓
Uni-NaVid	30.7	27.2	39.3	**7.41**
Vanilla fine-tuning	26.7	21.5	38.3	8.29
HiNav (Ours)	**36.4**	**29.5**	**43.5**	7.49

5.4 Request Quality and Human Validity

We validate VLM-generated requests through blind comparison with human-written counterparts, collecting 250 valid ratings per source. We also evaluate trajectory alignment with Qwen3-VL-Embedding (Li et al., 2026), computed as the cosine similarity between trajectory-video and request embeddings. As shown in Fig. 3, VLM requests achieve comparable task validity (3.98 ± 0.79 vs. 3.87 ± 0.88) and alignment (0.555 ± 0.057 vs. 0.546 ± 0.065). When asked to identify which group was human-written, participants answered correctly in only 45.2% of cases. Together with the HM3D-Test-Human results, these findings validate the quality of the generated requests.

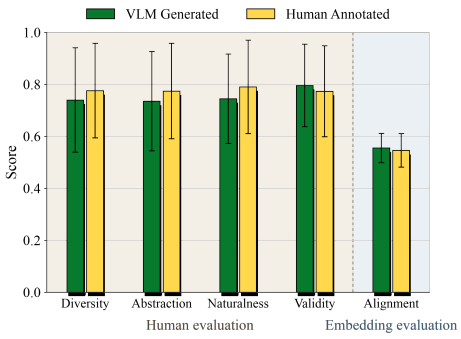


Figure 3: Quality comparison between human-written and VLM-generated requests. Human ratings are normalized from 1–5 to $[0, 1]$; error bars show SD. Embedding scores measure trajectory–request alignment.

5.5 Analysis of HiNav

Transfer across Frozen Executors. To test the generality of HiNav's textual interface, we connect the same trained Reasoner to StreamVLN and DualVLN. Each executor retains its official weights and remains frozen; HiNav supplies only the textual subtask guidance g_k. As shown in Table 4, HiNav raises StreamVLN from 14.7 to 44.0 SR and DualVLN from 19.3 to 50.7 SR on the same fixed MP3D-Val-VLM subset used for the matched-request intervention. These improvements support scene-grounded textual guidance as a reusable interface across executor architectures.

Analysis of the Hierarchical Learning. For the training-based ablations below, we use a separate fixed R2R-derived evaluation set in MP3D scenes. We compare direct and hierarchical adaptation

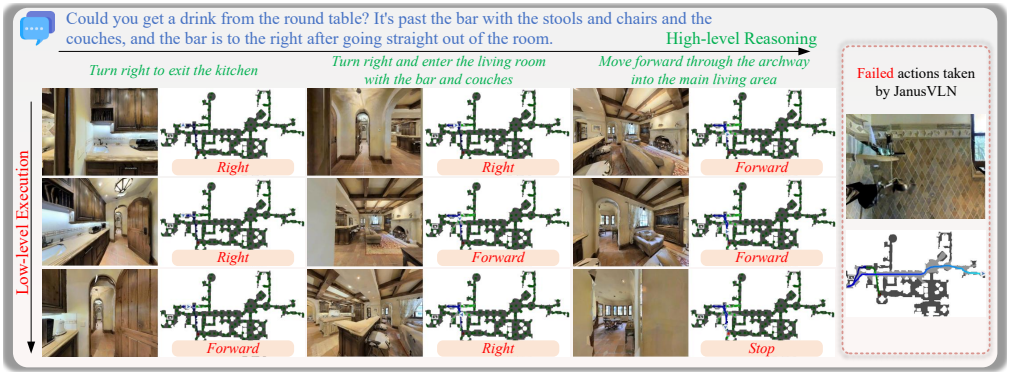


Figure 4: Qualitative simulation result. HiNav links high-level request reasoning with low-level execution to reach the intended destination, whereas JanusVLN deviates under the same request.

using the same training episodes in Table 5. Vanilla fine-tuning directly adapts Uni-NaVid from instruction following to request grounding using request–action pairs, yielding 26.7 SR compared with 30.7 for the original pretrained model. One possible explanation is that directly mapping underspecified requests to low-level actions introduces a substantial task shift and may disrupt the model's pretrained instruction-following prior, especially with limited supervision. HiNav instead converts request reasoning into aligned short-horizon subtask supervision, allowing the Executor to operate on guidance closer to its original instruction-following objective. HiNav achieves 36.4 SR, together with higher SPL and OSR, suggesting that hierarchical adaptation is more effective than direct request-to-action fine-tuning under the same training episodes.

History Grounding Ablation. We analyze the history-grounding component on a subset of MP3D episodes by comparing DS-HG with a sliding window, image-only selection, and request-only selection under the same memory capacity. As shown in Table 6, DS-HG achieves the highest SR and SPL while using the fewest steps among all variants. These results indicate that both temporal coverage and request relevance are useful for estimating task progress.

Table 6: History grounding ablation on R2R-derived MP3D evaluation set. All variants use the same keyframe capacity.

Configuration	SR↑	SPL↑	OSR↑	STEP↓
Sliding window	29.3	20.5	**50.2**	122.5
Image similarity only	<u>32.8</u>	<u>24.9</u>	38.9	89.4
Request similarity only	31.9	22.6	35.8	<u>79.0</u>
DS-HG (Ours)	**36.4**	**29.5**	<u>43.5</u>	**67.6**

5.6 Real-World Deployment

We further deploy HiNav without additional adaptation on a system composed of an RTX 4090 server and a Unitree Go2 robot equipped with an RGB camera. Visual observations are streamed to the Reasoner and Executor on the server, and predicted actions are converted to low-level robot commands. In physical indoor environments, HiNav grounds open-ended requests and reaches the intended destinations; additional visualizations are provided in the supplementary material. These qualitative results demonstrate the feasibility of zero-shot physical deployment.

6 Conclusion

We formulate Request Grounding VLN as a complementary setting that jointly requires target grounding, spatial reasoning, and route planning from underspecified everyday requests with incomplete navigation information. RG-VLN evaluates this setting across MP3D and HM3D, multiple trajectory sources, and VLM-generated and human-written requests. Its matched-trajectory analysis reveals a substantial performance gap between detailed route instructions and underspecified requests. HiNav addresses this gap by connecting high-level request reasoning with low-level execution through scene-grounded subtask guidance. It achieves the highest success rate on every RG-VLN split and substantially improves frozen VLN executors through the same textual interface, establishing an effective and extensible foundation for open-ended request navigation.

Ethics statement

This work involves human annotators for request collection and human evaluators for request-quality assessment. The study protocol was conducted in accordance with the relevant institutional research ethics requirements. All participants provided informed consent. No sensitive personal information was collected, and evaluation responses were stored anonymously. Participants were volunteers.

Reproducibility statement

We provide detailed benchmark construction procedures, evaluation protocols, training and inference configurations, and model hyperparameters in Sections 3–5 and Appendix A. The prompts used for benchmark construction, training-data synthesis, and inference are provided in Appendix D, together with the human annotation and evaluation interfaces in Appendix E. Code and benchmark annotations will be released to facilitate reproduction.

AI use statement

We used Qwen3-VL to synthesize and verify navigation requests and to generate training supervision, as described in the paper and supplementary material. Generated benchmark requests underwent automated checks and targeted human review following the procedures described in the benchmark construction section. We also used OpenAI Codex to assist with refining the presentation and adjusting LaTeX formatting. The authors take responsibility for the final content of this work, including all AI-assisted text and artifacts.

References

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.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report, 2025a. URL https://arxiv.org/abs/2511.21631.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025b. URL https://arxiv.org/abs/2502.13923.

Wenzhe Cai, Jiaqi Peng, Yuqiang Yang, Yujian Zhang, Meng Wei, Hanqing Wang, Yilun Chen, Tai Wang, and Jiangmiao Pang. Navdp: Learning sim-to-real navigation diffusion policy with privileged information guidance, 2025. URL https://arxiv.org/abs/2505.08712.

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, 2017. URL https://arxiv.org/abs/1709.06158.

Jiaqi Chen, Bingqian Lin, Ran Xu, Zhenhua Chai, Xiaodan Liang, and Kwan-Yee Wong. Mapgpt: Map-guided prompting with adaptive path planning for vision-and-language navigation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9796–9810, 2024.

An-Chieh Cheng, Yandong Ji, Zhaojing Yang, Xueyan Zou, Jan Kautz, Erdem Biyik, Hongxu Yin, Sifei Liu, and Xiaolong Wang. Navila: Legged robot vision-language-action model for navigation. In *RSS*, 2025.

Mengfei Du, Binhao Wu, Zejun Li, Xuan-Jing Huang, and Zhongyu Wei. Embspatial-bench: Benchmarking spatial understanding for embodied tasks with large vision-language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 346–355, 2024.

Weixi Feng, Tsu-Jui Fu, Yujie Lu, and William Yang Wang. Uln: Towards underspecified vision-and-language navigation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 6394–6412, 2022.

Zebin Han, Xudong Wang, Baichen Liu, Qi Lyu, Zhenduo Shang, Jiahua Dong, Lianqing Liu, and Zhi Han. Seqwalker: Sequential-horizon vision-and-language navigation with hierarchical planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18279–18287, 2026.

Zhiyu Huang, Yun Zhang, Johnson Liu, Rui Song, Chen Tang, and Jiaqi Ma. Tic-vla: A think-in-control vision-language-action model for robot navigation in dynamic environments. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2026.

Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 9902–9915. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/janner22a.html.

Zhiqiang Jiang and Xin Wang. Spatialgpt: Zero-shot vision-and-language navigation via spatial cot over structured spatial memory. In *Proceedings of the 33rd ACM International Conference on Advances in Geographic Information Systems*, pp. 423–435, 2025.

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

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

Mingxin Li, Yanzhao Zhang, Dingkun Long, Keqin Chen, Sibo Song, Shuai Bai, Zhibo Yang, Pengjun Xie, An Yang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Qwen3-vl-embedding and qwen3-vl-reranker: A unified framework for state-of-the-art multimodal retrieval and ranking, 2026. URL https://arxiv.org/abs/2601.04720.

Chenyang Ma, Guangyu Yang, Kai Lu, Shitong Xu, Bill Byrne, Niki Trigoni, and Andrew Markham. Cyclevla: Proactive self-correcting vision-language-action models via subtask backtracking and minimum bayes risk decoding. *arXiv preprint arXiv:2601.02295*, 2026.

Yuhang Peng, Sidong Wang, Jihaoyu Yang, Shilong Li, Han Wang, and Jiangtao Gong. Bench2freead: A benchmark for vision-based end-to-end navigation in unstructured robotic environments, 2025. URL https://arxiv.org/abs/2503.12180.

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.

Lin Qian, Shijie Li, Sihao Lin, Xuan Zhang, Bangya Liu, Yanran Li, and Hujun Yin. Intentionnav: A benchmark for intent-driven object navigation from implicit human instruction, 2026. URL https://arxiv.org/abs/2605.23187.

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.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John M Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. Habitat-matterport 3d dataset (HM3d): 1000 large-scale 3d environments for embodied AI. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2021. URL https://arxiv.org/abs/2109.08238.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9339–9347, 2019.

Lucy Xiaoyang Shi, Brian Ichter, Michael Robert Equi, Liyiming Ke, Karl Pertsch, Quan Vuong, James Tanner, Anna Walling, Haohuan Wang, Niccolo Fusai, Adrian Li-Bell, Danny Driess, Lachy Groom, Sergey Levine, and Chelsea Finn. Hi robot: Open-ended instruction following with hierarchical vision-language-action models. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 54919–54933. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/shi25d.html.

Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, Steven Bohez, Konstantinos Bousmalis, Anthony Brohan, Thomas Buschmann, Arunkumar Byravan, Serkan Cabi, Ken Caluwaerts, Federico Casarini, Oscar Chang, Jose Enrique Chen, Xi Chen, Hao-Tien Lewis Chiang, Krzysztof Choromanski, David D'Ambrosio, Sudeep Dasari, Todor Davchev, Coline Devin, Norman Di Palo, Tianli Ding, Adil Dostmohamed, Danny Driess, Yilun Du, Debidatta Dwibedi, Michael Elabd, Claudio Fantacci, Cody Fong, Erik Frey, Chuyuan Fu, Marissa Giustina, Keerthana Gopalakrishnan, Laura Graesser, Leonard Hasenclever, Nicolas Heess, Brandon Hernaez, Alexander Herzog, R. Alex Hofer, Jan Humplik, Atil Iscen, Mithun George Jacob, Deepali Jain, Ryan Julian, Dmitry Kalashnikov, M. Emre Karagozler, Stefani Karp, Chase Kew, Jerad Kirkland, Sean Kirmani, Yuheng Kuang, Thomas Lampe, Antoine Laurens, Isabel Leal, Alex X. Lee, Tsang-Wei Edward Lee, Jacky Liang, Yixin Lin, Sharath Maddineni, Anirudha Majumdar, Assaf Hurwitz Michaely, Robert Moreno, Michael Neunert, Francesco Nori, Carolina Parada, Emilio Parisotto, Peter Pastor, Acorn Pooley, Kanishka Rao, Krista Reymann, Dorsa Sadigh, Stefano Saliceti, Pannag Sanketi, Pierre Sermanet, Dhruv Shah, Mohit Sharma, Kathryn Shea, Charles Shu, Vikas Sindhwani, Sumeet Singh, Radu Soricut, Jost Tobias Springenberg, Rachel Sterneck, Razvan Surdulescu, Jie Tan, Jonathan Tompson, Vincent Vanhoucke, Jake Varley, Grace Vesom, Giulia Vezzani, Oriol Vinyals, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Fei Xia, Ted Xiao, Annie Xie, Jinyu Xie, Peng Xu, Sichun Xu, Ying Xu, Zhuo Xu, Yuxiang Yang, Rui Yao, Sergey Yaroshenko, Wenhao Yu, Wentao Yuan, Jingwei Zhang, Tingnan Zhang, Allan Zhou, and Yuxiang Zhou. Gemini robotics: Bringing ai into the physical world, 2025. URL https://arxiv.org/abs/2503.20020.

Hongcheng Wang, Andy Guan Hong Chen, Xiaoqi Li, Mingdong Wu, and Hao Dong. Find what you want: Learning demand-conditioned object attribute space for demand-driven navigation. *Advances in Neural Information Processing Systems*, 36:16353–16366, 2023.

Shaoan Wang, Jiazhao Zhang, Minghan Li, Jiahang Liu, Anqi Li, Kui Wu, Fangwei Zhong, Junzhi Yu, Zhizheng Zhang, and He Wang. Trackvla: Embodied visual tracking in the wild, 2025a. URL https://arxiv.org/abs/2505.23189.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, Zhaokai Wang, Zhe Chen, Hongjie Zhang, Ganlin Yang, Haomin Wang, Qi Wei, Jinhui Yin, Wenhao Li, Erfei Cui, Guanzhou Chen, Zichen Ding, Changyao Tian, Zhenyu Wu, Jingjing Xie, Zehao Li, Bowen Yang, Yuchen Duan, Xuehui Wang, Zhi Hou, Haoran Hao, Tianyi Zhang, Songze Li, Xiangyu Zhao, Haodong Duan, Nianchen Deng, Bin Fu, Yinan He, Yi Wang, Conghui He, Botian Shi, Junjun He, Yingtong Xiong, Han Lv, Lijun Wu, Wenqi Shao, Kaipeng Zhang, Huipeng Deng, Biqing Qi, Jiaye Ge, Qipeng Guo, Wenwei Zhang, Songyang Zhang, Maosong Cao, Junyao Lin, Kexian Tang, Jianfei Gao, Haian Huang, Yuzhe Gu, Chengqi Lyu, Huanze Tang, Rui Wang, Haijun Lv, Wanli Ouyang, Limin Wang, Min Dou, Xizhou Zhu, Tong Lu, Dahua Lin, Jifeng Dai, Weijie Su, Bowen Zhou, Kai Chen, Yu Qiao, Wenhai Wang, and Gen Luo. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency, 2025b. URL https://arxiv.org/abs/2508.18265.

Zihan Wang, Yaohui Zhu, Gim Hee Lee, and Yachun Fan. NavRAG: Generating user demand instructions for embodied navigation through retrieval-augmented LLM. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 8430–8440. Association for Computational Linguistics, 2025c. doi: 10.18653/v1/2025.findings-acl.442. URL https://aclanthology.org/2025.findings-acl.442/.

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

Meng Wei, Chenyang Wan, Xiqian Yu, Tai Wang, Yuqiang Yang, Xiaohan Mao, Chenming Zhu, Wenzhe Cai, Hanqing Wang, Yilun Chen, Xihui Liu, and Jiangmiao Pang. Streamvln: Streaming vision-and-language navigation via slowfast context modeling, 2026b. URL https://arxiv.org/abs/2507.05240.

Karmesh Yadav, Jacob Krantz, Ram Ramrakhya, Santhosh Kumar Ramakrishnan, Jimmy Yang, Austin Wang, John Turner, Aaron Gokaslan, Vincent-Pierre Berges, Roozbeh Mootaghi, Oleksandr Maksymets, Angel X Chang, Manolis Savva, Alexander Clegg, Devendra Singh Chaplot, and Dhruv Batra. Habitat challenge 2023. https://aihabitat.org/challenge/2023/, 2023.

Hang Yin, Xiuwei Xu, Linqing Zhao, Ziwei Wang, Jie Zhou, and Jiwen Lu. Unigoal: Towards universal zero-shot goal-oriented navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19057–19066, 2025.

Naoki Yokoyama, Ram Ramrakhya, Abhishek Das, Dhruv Batra, and Sehoon Ha. Hm3d-ovon: A dataset and benchmark for open-vocabulary object goal navigation. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5543–5550. IEEE, 2024.

Shuang Zeng, Dekang Qi, Xinyuan Chang, Feng Xiong, Xie Shichao, Xiaolong Wu, Shiyi Liang, Mu Xu, and Xing Wei. JanusVLN: Decoupling semantics and spatiality with dual implicit memory for vision-language navigation. In *The Fourteenth International Conference on Learning Representations*, 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. *Robotics: Science and Systems*, 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.

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

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand, 2024. Association for Computational Linguistics. URL http://arxiv.org/abs/2403.13372.

Gengze Zhou, Yicong Hong, and Qi Wu. Navgpt: Explicit reasoning in vision-and-language navigation with large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 7641–7649, 2024.

Zewei Zhou, Tianhui Cai, Yun Zhao, Seth Z.and Zhang, Zhiyu Huang, Bolei Zhou, and Jiaqi Ma. Autovla: A vision-language-action model for end-to-end autonomous driving with adaptive reasoning and reinforcement fine-tuning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Fengda Zhu, Xiwen Liang, Yi Zhu, Qizhi Yu, Xiaojun Chang, and Xiaodan Liang. Soon: Scenario oriented object navigation with graph-based exploration. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12689–12699, 2021.

APPENDIX: SUPPLEMENTARY MATERIAL

This Supplementary Material accompanies the main text with additional implementation details, benchmark statistics, extended experimental results, and human-facing interface examples. Specifically, section A elaborates on technical configurations, section B analyzes the benchmark statistics, and section C presents extended quantitative analyses and qualitative visualizations. section D details the specific prompts utilized in our framework, and section E presents the request annotation interface and the human evaluation interface used to assess request quality. Finally, section F delineates the limitations of our current work and outlines potential directions for future research.

A MORE IMPLEMENTATION DETAILS

A.1 EVALUATION METRICS

To comprehensively evaluate the navigation performance of baselines and our HiNav, we employ five standard metrics:

- Success Rate (SR): This is our primary evaluation metric. A navigation task is deemed successful only if the agent terminates within a 3m radius of the target location. Formally, for each episode, the success status is a binary value $\{0, 1\}$, and the SR is calculated as the average success across all episodes.
- Oracle Success Rate (OSR): This metric measures if the agent passes within 3m of the target at any point during its trajectory. Formally, for each episode, the oracle success status is a binary value $\{0, 1\}$, and the OSR is calculated as the average oracle success across all evaluation episodes.
- Success weighted by Path Length (SPL): This metric jointly considers navigation success and efficiency, defined as:

$$\text{SPL} = \frac{1}{N_{\text{ep}}} \sum S_i \frac{L_i}{\max(P_i, L_i)}, \tag{4}$$

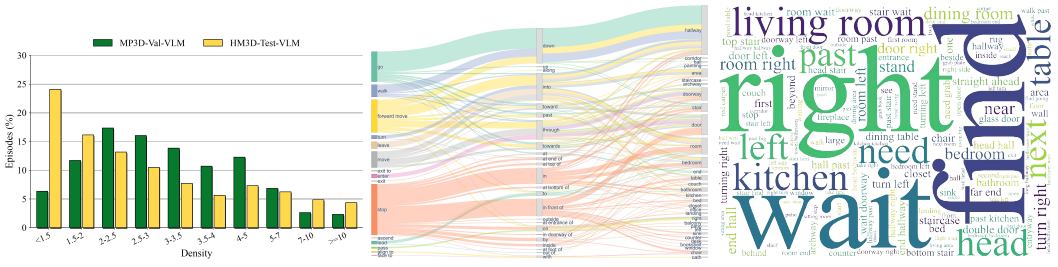
  where $S_i \in \{0, 1\}$ is a binary indicator of success for episode i, L_i is the shortest path length to the target, and P_i is the actual path length taken by the agent, N_{ep} is the total number of episodes in the validation set.
- Navigation Error (NE): This metric represents the average geodesic distance between the agent's final stopping point and the target destination across all evaluation episodes. It serves as an indicator of the termination precision.
- STEP: This metric indicates the average number of navigation steps taken by the agent from the start to termination across all evaluation episodes.

Among these metrics, SR, OSR, and SPL are higher-is-better metrics, indicating higher navigation success; conversely, NE and STEP are lower-is-better metrics, reflecting lower navigation error and motion cost.

A.2 BASELINES CONFIGURATION

To evaluate the generalization and adaptability of state-of-the-art (SOTA) VLM-based methods to underspecified requests, we conduct a comprehensive benchmarking on our RG-VLN dataset, including:

- NaVid (Zhang et al., 2024): The inaugural video-based VLM for VLN, utilizing monocular RGB streams without reliance on explicit maps or odometry.
- NaVILA (Cheng et al., 2025): A dual-system approach combining a VLM trained on diverse datasets, including real-world videos, simulated navigation, question-answering pairs, with a vision-based locomotion policy.
- Uni-NaVid (Zhang et al., 2025a): A SOTA embodied agent that achieves superior performance across multiple tasks, including VLN, object navigation, embodied question answering, and human following, by leveraging massive navigation and question-answer datasets.



(a) Request-detail density (b) Subtask structure (c) Keyword distribution

Figure 5: Statistical analysis of the RG-VLN benchmark. Subfigures show (a) request-detail density, (b) subtask transition structure, and (c) request keyword distribution.

- StreamVLN (Wei et al., 2026b): A high-performance VLN framework that models navigation as a multi-turn dialogue. By incorporating an efficient KV cache management mechanism, it enables effective long-range history management.
- JanusVLN (Zeng et al., 2026): This method introduces a dual-stream architecture that decouples spatial-geometric and visual-semantic memory. JanusVLN achieves efficient memory management and robust spatial understanding, resulting in state-of-the-art success rates on standard navigation benchmarks.
- DualVLN (Wei et al., 2026a): A dual-system navigation method that grounds intermediate visual goals with a VLM and executes them using a low-level action expert.

For each baseline, we adopt their official code, weights, and recommended hyperparameters to ensure a fair and reproducible assessment. This evaluation aims to highlight the performance gap between existing navigation paradigms and the demands of real-world request grounding, thereby establishing a clear baseline for the effectiveness of our proposed HiNav framework.

A.3 HiNav Fine-tuning and Hyperparameters

Training Configurations. For the Request Reasoner, we fine-tune the Qwen3-VL-8B (Bai et al., 2025a) using LoRA targeting $\{q, v, o, k, gate, up, down\}$ projections with a rank of 8 via the LLaMA-Factory (Zheng et al., 2024). The batch size is set to 64, with a learning rate of 3.0×10^{-5} using a cosine annealing schedule for one epoch. As for the Executor, it is initialized with Uni-NaVid (Zhang et al., 2025a) weights, and the vision encoder remains frozen. We fine-tune for 3 epochs with a batch size of 256 and an initial learning rate of 5.0×10^{-6} under a cosine decay. Both models are trained on a cluster of 8 NVIDIA A100 GPUs.

Training Data Construction. The fine-tuning dataset is derived exclusively from R2R-CE training trajectories (Krantz et al., 2020). Existing VLN data do not provide the temporally aligned subtask descriptions required by HiNav, so we synthesize hierarchical supervision from image-action trajectories. Directly asking a VLM to segment a full trajectory into complete subtasks is unstable; therefore, inspired by subtask-centric robotic data pipelines (Ma et al., 2026), we first perform coarse segmentation over the action sequence by aggregating consecutive identical low-level actions into action chunks. These chunks are often fragmented because navigation trajectories contain short corrective turns and noisy local adjustments. We then use Qwen3-VL-235B-A22B (Bai et al., 2025a) to merge and refine action chunks according to image semantics, landmark changes, and the original route instruction, producing complete subtask spans and concise subtask descriptions. To ensure that the model learns reliable termination, the final subtask of each episode is constrained to be a Stop subtask. We further apply rule-based checks to detect frame-index overlaps, gaps, and temporal reversals, followed by VLM-based semantic verification for instruction-trajectory consistency. The resulting training set contains 9,927 episodes and approximately 420K image-action pairs. Training scenes are disjoint from all RG-VLN evaluation scenes, and no intermediate annotations from evaluation episodes are used.

To rigorously evaluate the generalization capability of HiNav across both navigation tasks and linguistic variations, we deliberately restrict the fine-tuning process to only the first request type, inten-

Table 7: Performance analysis across diverse requests on the subset of MP3D-Val-VLM. Metrics include Success Rate (SR), Success weighted by Path Length (SPL), Oracle Success Rate (OSR), Navigation Error (NE), and Steps Taken (STEP). For SR, values in parentheses denote the deviation from the model's Overall SR. Avg. Diff. represents the mean of the absolute deviations, which serves as a metric to evaluate the robustness of different methods across varied request types. Bold values indicate the best performance in each row.

Req.	Metrics	**NaVid**(Zhang et al., 2024)	**Uni-NaVid**(Zhang et al., 2025a)	**StreamVLN**(Wei et al., 2026b)	**NaVILA**(Cheng et al., 2025)	**JanusVLN**(Zeng et al., 2026)	**HiNav (Ours)**
1	**SR(%)↑**	21.4(-2.9)	26.6(-4.1)	29.9(-2.5)	22.5(-4.2)	19.2(-10.9)	35.3(-1.1)
	SPL(%)↑	17.7	23.8	26.7	21.0	16.6	28.3
	OSR(%)↑	29.2	34.5	37.9	28.2	32.3	42.9
	NE(m)↓	7.73	7.81	7.39	7.72	8.03	7.57
	STEP↓	77.0	60.5	83.6	98.3	113.7	69.7
2	**SR(%)↑**	24.7(+0.4)	32.7(+2.0)	33.7(+1.3)	27.7(+1.0)	35.2(+5.1)	36.5(+0.1)
	SPL(%)↑	20.4	28.8	29.6	24.9	32.0	29.7
	OSR(%)↑	34.0	41.4	43.3	36.8	44.6	42.7
	NE(m)↓	7.27	7.23	7.14	7.54	6.90	7.38
	STEP↓	80.8	65.8	87.4	121.6	106.5	65.8
3	**SR(%)↑**	26.9(+2.6)	33.2(+2.5)	33.8(+1.4)	30.3(+3.6)	36.7(+6.6)	37.5(+1.1)
	SPL(%)↑	22.8	29.1	29.7	26.9	33.1	30.5
	OSR(%)↑	35.7	42.2	44.9	40.9	47.8	45.1
	NE(m)↓	7.14	7.15	7.10	7.23	6.84	7.53
	STEP↓	77.5	66.1	93.1	118.7	106.3	67.2
Overall SR(%)↑		24.3	30.7	32.4	26.7	30.1	**36.4**
Avg. Diff.(%)↓		1.97	2.87	1.73	2.93	7.53	**0.77**

tionally excluding the other two more diverse request variations. This experimental design allows us to assess whether HiNav can effectively interpret and follow unseen, complex, or diverse requests despite being trained on a constrained request set.

Inference Setup. During inference, the Request Reasoner updates subtasks every 4 steps, while the Executor performs action generation every 2 steps, following the configuration in (Zhang et al., 2025a). The step size for forward movement is set to 0.25m, and the rotation angle is set to $30°$. To ensure trajectory coverage and prevent out-of-memory errors, we set the minimum and maximum keyframe limits to $N_{\min} = 8$ and $N_{\max} = 36$, respectively. The Request Reasoner is deployed using the vLLM serving framework (Kwon et al., 2023) to optimize inference throughput and resource utilization.

Real-World Deployment. For our real-world experiments, we deploy HiNav on a Unitree Go2 quadruped robot paired with an RTX 4090 server. The robot is equipped with an Intel RealSense D435 RGB camera for image acquisition and an NVIDIA Jetson Orin module for robot-side communication and locomotion control. We follow the workflow (Wei et al., 2026b): the robot streams live visual data to the server via a high-speed WiFi connection, while the server executes the Request Reasoner and Executor to generate subtasks and actions, which are then transmitted back to the robot for execution. Additional physical robot rollouts are provided in the supplementary video.

B Benchmark Statistical Analysis

The benchmark comprises the MP3D-Val-VLM, HM3D-Test-VLM, and HM3D-Test-Human splits described in the main text. To quantify linguistic diversity and task complexity, we analyze request-detail density, subtask structure, and lexical coverage, following the order of Figure 5.

Request-Detail Density. Following the main text, we define request-detail density for a request I and reference trajectory τ as $D(I, \tau) = |I|/L(\tau)$, where $|I|$ is the request length and $L(\tau)$ is the geodesic length of the reference trajectory. This metric measures how much linguistic guidance is provided per unit navigation distance, and we use it to evaluate whether the information density of RG-VLN requests is reasonable. As shown in Figure 5(a), the density distribution is concentrated in a moderate range rather than collapsing to either extremely sparse or overly verbose requests. This indicates that most requests provide sufficient target and spatial cues for grounding, while still avoiding step-by-step route specification. The distribution therefore supports the intended benchmark design: RG-VLN should reduce explicit navigation information without making the request unrealistically ambiguous or excessively detailed.

Subtask Structural Balance. We analyze subtask transitions in Figure 5(b). The generated requests cover diverse subtask structures, reflecting variation in landmark progression, turning behavior, and stopping conditions across episodes. This structural diversity complements the density analysis

Table 8: Navigation performance of various VLM-based reasoners on a subset of MP3D-Val-VLM. Best and second-best results are shown in **bold** and underlined, respectively.

Method	SR (%)↑	SPL (%)↑	OSR (%)↑	NE (m)↓	STEP↓
InternVL3.5-8B (Wang et al., 2025b)	15.3	8.4	49.2	10.13	145.6
Qwen2.5-VL-7B (Bai et al., 2025b)	14.1	7.8	49.5	10.58	150.9
Qwen3-VL-30B-A3B (Bai et al., 2025a)	17.5	9.9	**50.4**	9.80	138.1
Qwen3-VL-8B (Bai et al., 2025a)	21.3	13.7	47.7	9.40	120.2
HiNav (Ours)	**36.4**	**29.5**	43.5	**7.49**	**67.6**

Table 9: Comparison of task completion time and navigation performance on a subset of MP3D-Val-VLM. Avg. Time denotes the average time to complete a single evaluation episode. Best and second-best results are shown in **bold** and underlined, respectively.

Method	SR (%)↑	SPL (%)↑	OSR (%)↑	STEP ↓	Avg. Time (s)↓
StreamVLN (Wei et al., 2026b)	32.4	28.6	41.9	87.9	15.05
HiNav (Ours)	**36.4**	**29.5**	**43.5**	**67.6**	**15.00**

by showing that the benchmark contains varied navigation layouts while maintaining a controlled amount of language information per path distance.

Linguistic Diversity and Intent Coverage. As illustrated in Figure 5(c), the keyword distribution spans both navigation primitives and landmark-based spatial references, while goal-oriented verbs such as *find* and *wait* indicate that the requests are intent-driven rather than purely route-following. This lexical pattern is consistent with the RG-VLN construction goal: requests should expose target intent and spatial cues, but should not collapse back into dense, action-by-action VLN route instructions.

C Additional Experimental Results

C.1 Performance Analysis across Diverse Requests

As previously described, our benchmark leverages a multi-stage VLM prompting strategy to generate three distinct types of template-free requests with varied linguistic styles. Request type 1 typically begins with a question and omits explicit motion primitives, instead relying on spatial-geometric descriptions to guide the agent (e.g., *"Could you turn on the kitchen lights? The light switches are on the stone wall before the kitchen island with the sink, and the island is in the kitchen which is ahead past the curio cabinet."*). Request types 2 and 3 still preserve the high-level intent of RG-VLN requests, while incorporating additional action cues that provide different levels of navigation specification (e.g., *"Head into the kitchen past the big display cabinet and find the light switches on the stone pillar right before the island with the sink, I need to brighten up the space for dinner prep."* and *"Go past the curio cabinet and into the kitchen, then stop at the light switch on the stone wall next to the main counter, I want to check the food I left in the oven."*).

To rigorously evaluate the generalization and robustness of the baselines and our HiNav, we present a detailed performance breakdown in Table 7, which reports a subset of MP3D-Val-VLM. Several key observations can be made. First, across all three request types, HiNav maintains a clear lead, achieving the highest SR in every category. This consistently superior performance underlines the effectiveness of our framework in tackling the complexities of request-grounded VLN. Second, all methods exhibit a performance dip on request type 1 compared to the other two types. This suggests that request type 1, which prioritizes spatial reasoning over explicit motion cues, presents a higher cognitive challenge for existing agents. Conversely, the additional action cues in request types 2 and 3 facilitate easier goal alignment, resulting in higher SRs across the board. Most notably, the Avg. Diff. metric reveals that HiNav exhibits the highest stability across diverse requests, with an average deviation of only 0.77%, lower than all baselines. It is crucial to emphasize that HiNav is fine-tuned exclusively on request type 1. The fact that it generalizes so effectively to unseen request styles (2 and 3) stems from its hierarchical decomposition. By decoupling complex navigation into high-

level request reasoning and low-level action execution, HiNav alleviates the reasoning bottleneck, enabling deeper semantic understanding and more robust execution in request-grounded scenarios.

C.2 Performance Comparison of VLM-based Reasoning

Table 8 presents the performance of various open-source VLMs on a subset of MP3D-Val-VLM when integrated as high-level reasoners within the HiNav framework, including InternVL3.5-8B (Wang et al., 2025b), Qwen2.5-VL-7B (Bai et al., 2025b), Qwen3-VL-8B (Bai et al., 2025a), and Qwen3-VL30B-A3B (Bai et al., 2025a). Experimental results show that off-the-shelf VLMs exhibit lower SRs than our method. Meanwhile, all VLM-based reasoners yield significantly higher OSR and STEP counts, indicating that these models primarily struggle with subtask termination uncertainty, leading to excessive and redundant trajectories. Among the open-source candidates, Qwen3-VL-8B delivers the best performance, consistent with its superior results on embodied benchmarks such as EmbSpatial-Bench (Du et al., 2024) and ERQA (Team et al., 2025). Ultimately, our method, which benefits from precise subtask supervision, achieves superior performance and outperforms Qwen3-VL-8B by a substantial 15.1% in SR.

C.3 Quantitative Evaluation of Runtime Efficiency

We evaluate the runtime efficiency of HiNav against StreamVLN (Wei et al., 2026b) on a subset of MP3D-Val-VLM using a single NVIDIA A100 GPU. As summarized in Table 9, HiNav outperforms StreamVLN by 4.0% in SR while achieving a marginally shorter mission duration (by a mere 0.05s). HiNav reaches the goal with fewer steps than StreamVLN, implying that its per-step operational latency is slightly higher due to the introduction of the high-level Request Reasoner. These results demonstrate that our hierarchical architecture effectively balances overall efficiency and navigation performance. By decoupling subtask reasoning from action generation, HiNav leverages the temporal coordination between the slow Request Reasoner and the fast Executor. This design allows the agent to avoid redundant steps through more deliberate decision-making, without sacrificing real-time operational requirements in request-grounded scenarios.

C.4 Reasoner-Executor Coordination

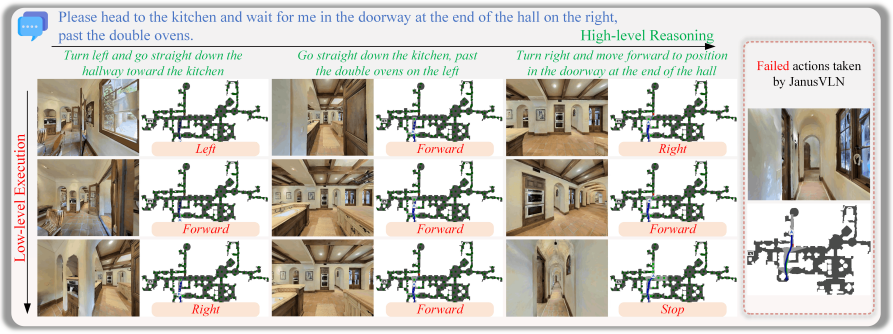
Table 10 studies the temporal coordination between the slow Request Reasoner and the fast Executor on a subset of MP3D-Val-VLM by varying the reasoner invocation interval. A very small interval, such as 2 steps, allows frequent re-reasoning but can make the system overly reactive to short-term observations; it also increases high-level reasoning overhead and yields a lower SR than the 4-step setting. In contrast, larger intervals reduce reasoner calls but delay subtask refreshes. When the interval grows to 8 or 12 steps, OSR remains competitive because the agent can still pass near the target, but SR and SPL decrease and STEP increases, indicating weaker stopping precision and more redundant execution. Invoking the Request Reasoner every 4 steps achieves the best SR and SPL while keeping the number of steps close to the minimum. We therefore use this setting as the default, matching the intended slow-fast design: the Reasoner provides stable short-horizon semantic guidance, while the Executor handles local action generation at a higher frequency.

Table 10: Reasoner-executor coordination on a subset of MP3D-Val-VLM. Frequency denotes the interval of reasoner invocation, measured in navigation steps. Best and second-best results are shown in **bold** and underlined, respectively.

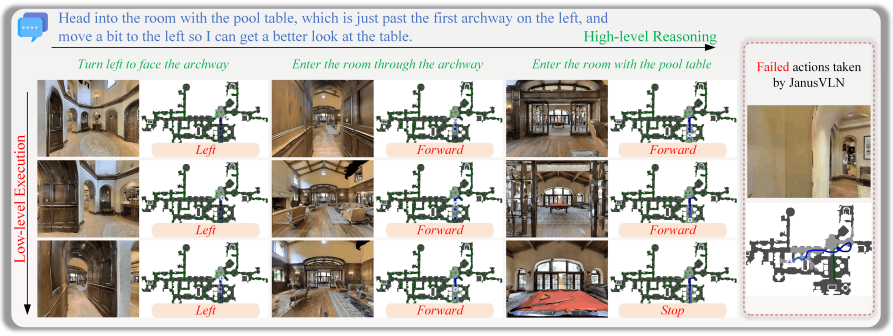
Freq.	SR↑	SPL↑	OSR↑	STEP↓
2	33.6	<u>27.7</u>	41.3	**65.6**
4	**36.4**	**29.5**	43.5	<u>67.6</u>
8	<u>34.2</u>	27.5	<u>44.0</u>	71.8
12	33.6	26.6	**45.0**	75.2

C.5 Visualizations of Navigation Process

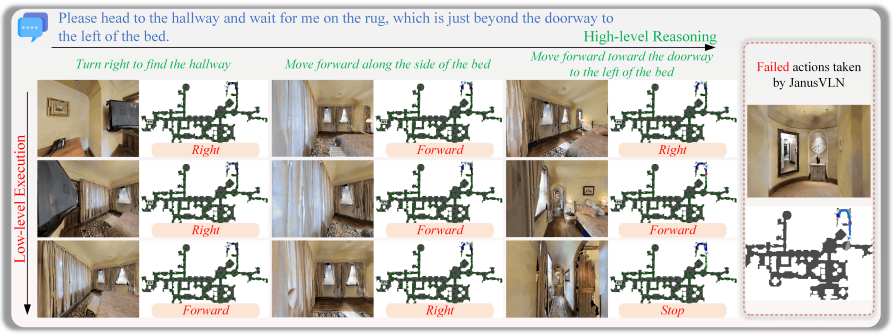
Figure 6 shows HiNav's trajectories in three request-grounding scenarios. The Request Reasoner decomposes underspecified requests (blue) into subtasks (green), which the Executor translates into atomic actions (red). The rightmost panels show failed JanusVLN (Zeng et al., 2026) trajectories, illustrating the benefit of connecting high-level request reasoning with low-level execution.



(a) Evaluation Episode 1.



(b) Evaluation Episode 2.



(c) Evaluation Episode 3.

Figure 6: Qualitative visualization of HiNav's navigation process. The input underspecified request is decomposed by the Request Reasoner into sequential subtasks, which are then translated by the Executor into actions. The rightmost panel displays failed trajectories from the baseline JanusVLN (Zeng et al., 2026) for comparison. Note that only keyframes and their corresponding subtasks are presented here for brevity. More complete navigation episodes are provided in RG-VLN.mp4.

D VLM PROMPTS

In this section, we provide the detailed prompts used in our RG-VLN benchmark construction and in the synthesis of training data. These prompts are designed to exploit the reasoning and multimodal understanding abilities of Qwen3-VL-235B-A22B (Bai et al., 2025a). We also present the prompt configurations used by the Request Reasoner and Executor during inference.

D.1 BENCHMARK CONSTRUCTION

To ensure the high-quality generation of free-form, diverse, and goal-oriented requests, we design prompts for each stage of the construction pipeline. The specific prompts for subtask analysis, request generation, quality verification, and self-correction are illustrated below.

Subtask Analysis

Role: You are a Research Expert in Robot Vision-Language Navigation. You have just been provided with a sequence of robot navigation images (above) and an original route instruction.

Task: Decompose the entire navigation path shown in the images into a sequence of Atomic Subtasks based on the Global Route Instruction.

Input:
1. Global Route Instruction: "<instruction>"
2. Image Sequence: Provided above. Each image has a Red Step ID marked at the top-left; you MUST use these IDs to ground your temporal reasoning.

Reasoning Process:
Before generating the final JSON, you must analyze the image stream by following these steps:
1. Scan for Boundaries: Look at the Red Step IDs. Identify ranges of frames where the robot performs a consistent action.
2. Landmark Alignment: Check if the landmarks visible in these frame ranges match the original route instruction.
3. Merge Noise: If you see a very short segment of "Turning" in the middle of a "Straight" path, ignore it as sensor noise.

Rules for Subtasks:
1. Action Types: The "action" description implies one of three types:
- Straight: (e.g., "Go straight past the sofa")
- Turn: (e.g., "Turn left towards the bedroom")
- Stop: (e.g., "Stop in front of the bed")
2. Format: The "action" string must use the format: "[Verb] [Preposition] [Landmark]".
- Good: "Go straight down the hallway"
- Bad: "Move forward", "Turn", "See a door"
3. Merge Approach and Stop. If the robot is moving towards the final goal and then immediately stops at that goal within a few frames, merge these into a single "Stop" action.
- Bad: Step 4: "Go straight to the sofa", Step 5: "Stop at the sofa".
- Good: Step 4: "Stop in front of the sofa".
4. Mandatory Final Action: "Stop"
- Since the image sequence records the full path, the final subtask MUST describe the arrival at the destination using the "Stop" action.
- Even if the robot is still slightly moving towards the goal in the last few frames, interpret this as the final arrival.
- Example: Change "Go straight to the sofa" (at the end of video) -> "Stop in front of the sofa".

Output Format:
Output ONLY a JSON object containing a single key "subtasks", where each subtask includes a "step" number and an "action" description.

Request Generation

Role: You are a professional in Vision-Language Navigation (VLN) and human-computer interaction, expert at generating human-like underspecified navigation requests for robots to execute. You will be provided with decomposed atomic subtasks of a robot's navigation path and the corresponding navigation image sequence. Your core task is to generate a natural request that mimics human daily direction-giving: focus on "where the destination is" instead of "how to move there", retain all key spatial relationships, and avoid any specific movement descriptions.

Input:
1. A list of subtasks: <subtasks>
2. Image sequence (the step ID is marked in red text at the top-left corner of each image): <img_list>

Core Request Generation Rules:
1. Key Spatial Relationship Retention: Reserve all critical spatial correlations between landmarks and the final destination from subtasks and images.
2. No Specific Movement Descriptions: Strictly avoid mentioning any robot locomotion actions (e.g., move forward/turn left/step, straight/turn/stop, backtracking) or movement-related adverbs.
3. Human-like Direction Logic: Fully follow how humans naturally give directions in daily life, using colloquial spatial description words.

Two Mandatory Step-by-Step Tasks:
Task 1: Infer a Realistic Navigation Purpose (Expressed as a Question)
- Infer a scenario-matching, realistic purpose based on the final destination (from subtasks) and landmark context (e.g., hallway, bathroom, sofa).
- The purpose must align with daily life logic and cover one of three dimensions (match the destination's typical usage):
1. Daily chores (e.g., clean the rug, tidy the coffee table).
2. Item-related needs (e.g., fetch a jacket, get a plate).
3. Interaction requirements (e.g., wait for me, come to me).
- If no specific items are in the subtasks, select an item from the image of the final destination (the item must be clearly visible in the final destination image and fit the daily usage of the destination).
- Ensure the destination implied by the purpose is consistent with the final destination in subtasks; express the purpose as a natural short question (no complex clauses).

Task 2: Describe the Final Destination's Spatial Position (Expressed as a Statement)
- Describe the precise spatial orientation and landmark correlation of the final destination only based on image visual information and subtask spatial relationships.
- Reference critical landmarks in the path and their relative positions to the destination.
- Use human-like spatial prepositions and description words; avoid robot/technical terms and any image/frame/step ID references.

Output Format:
Output only two parts in sequence, with no separation by line breaks or commas: a question for the navigation purpose (Task 1) followed by a statement for the spatial position of the final destination (Task 2).

Check Generated Requests

Role: You are a Vision Language Navigation (VLN) expert. Your task is to evaluate the generated request against the original route instruction and score it.

Scoring Criteria:
1. The goal must be relevant to the original destination and scenario-matching. The implied destination in the generated request must match the original route instruction's destination.
2. The spatial logic in the generated request must be fully consistent with the original route instruction (all key landmarks retained, correct spatial relationships). Correcting obvious typos in the original route instruction does NOT count as a violation.
3. The generated request should focus on the spatial relationships from the third-person view, avoiding specific movement verbs.

Scoring Scale:
- 1.0: No errors.
- 0.0: Any core error.

Output JSON only: {"score": your request score, "reason": a brief reason (less than 100 words)}.

Input:
- Original Route Instruction: <original>
- Generated Request: <request>
- Image sequences: <img_list>

Self-Correct Generated Requests

Role: You are a VLN (Vision-Language Navigation) request reviser. Your task is to revise the provided generated request based on the given evaluation JSON and the original route instruction.

Inputs:
- Original route instruction: <instruction>
- Generated request: <request>
- Evaluation JSON: <evaluation_json> // schema: { "score": 1.0—0.0, "reason": "..." }

Repair policy:
- If score is 1.0: Return the generated request unchanged.
- If score is 0.0: Fix only what is needed to solve the violation based on the original route instruction. Do not add objects/locations/actions not present in the instruction. Preserve the original path logic. Prefer minimal edits.
- If ambiguity arises, choose the least assumptive phrasing that remains faithful to the original route instruction (e.g., keep original nouns; avoid inventing adjectives or counts).
- Keep wording close to the generated request; avoid paraphrasing unless needed for compliance.
- Address typos (spelling, grammar, or punctuation errors) as part of violation fixes: Apply minimal edits to correct obvious typos (e.g., "dinning" -¿ "dining", "coutch" -¿ "couch", "on the to the right" -¿ "on the right") without altering the original intent or spatial references. Follow the principle of minimal edit distance to avoid unnecessary changes.

Procedure:
1. Read the original route instruction and infer the path logic.
2. Revise the generated request to resolve violations.
3. Output only the corrected request.

Only output the corrected request (no extra explanations, labels, or formats).

D.2 Fine-Tuning Dataset Synthesis

For the synthesis of our fine-tuning dataset, we employ Qwen3-VL-235B-A22B (Bai et al., 2025a) to facilitate the refinement and validation of subtask chunks. The corresponding prompt for chunk refinement is provided below.

Refine Subtask Chunks

Role: You are an expert Robot Navigation Video Analyst. Your goal is to Group raw physical actions into high-level semantic subtasks.

Inputs:
- A Global Instruction (the high-level goal).
- A list of Raw Chunks (based on action).
- Each Chunk has an ID (e.g., 0, 1, 2...).
- Each Chunk covers a specific time range.
- The Chunk IDs are NOT frame IDs. A chunk can contain many frames.

Global Instruction Analysis:
First, understand the user's intent. Identify key landmarks (e.g., "plant", "door", "kitchen") and spatial relations (e.g., "past", "towards", "through").

Processing Rules (Apply strictly):
1. Intelligent Merging
- Scenario: A sequence of actions where the agent is primarily moving towards a goal, but includes short physical actions (e.g., 1-2 frames of "Turn Right" or "Turn Left") at the start, middle, or end of the sequence.
- "Stop" chunk cannot be merged with any other chunk.
2. Rules for Subtasks:
- Action Types: The "action" description implies one of three types: Straight (e.g., "Go

straight past the sofa"), Turn (e.g., "Turn left towards the bedroom"), and Stop (e.g., "Stop in front of the bed").
- Format: The "action" string must use the format: "[Verb] [Preposition] [Landmark]".
- Subtasks should be concise and correct, only depicting the intent reflected by the chunks and ignoring minor adjustments (typically under 2 frames).
3. Matching Rules:
- Each subtask must contain at least one chunk.
- Each chunk can only belong to one subtask.

Your Tasks:
1. Landmark Analysis
- Analyze the image frames associated with each chunk.
- Identify the visible landmarks and their spatial roles.
- Track landmark state changes across chunks (e.g., appearing, disappearing).
- Ignore the physical action labels in this step.
2. Hard Boundary Identification (Landmark-Driven)
- Identify hard boundaries between adjacent chunks.
- Chunks separated by a hard boundary MUST NOT belong to the same subtask.
3. Semantic Chunk Grouping
- Decide whether adjacent chunks should be merged or kept separate based on semantic intent.
- Suppress noise chunks that only support alignment or continuation.
- Treat a "Turn" chunk as semantic ONLY if it causes a landmark change.
4. Subtask Formation and Description
- Each final group of chunks becomes one subtask.
- Describe each subtask in a concise manner.

Output Format:
Output a list of subtasks and list the "chunk_ids" that belong to each subtask in the JSON format.

D.3 Inference Prompts

After fine-tuning, our framework requires only streamlined prompts during inference. For the Request Reasoner, the prompt consists of the global navigation request and sequential visual contexts. For the Executor, which is fine-tuned upon Uni-NaVid (Zhang et al., 2025a), we maintain its original prompt structure but augment it with the generated subtasks as additional guidance.

High-Level Request Reasoner

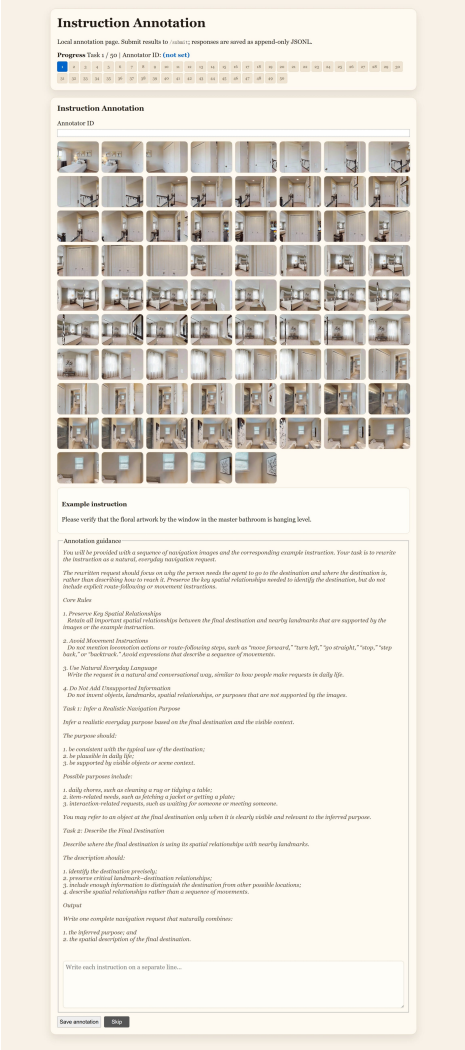
You are an expert Vision-Language Navigation reasoner. Your task is to determine the NEXT high-level semantic subtask, based on the navigation request, the visual observation history and the current visual observation. A subtask is an abstract, goal-oriented action.

Navigation Request:
<request>
Visual Observation History:
<history_img>
Current Visual Observation:
<current_img>

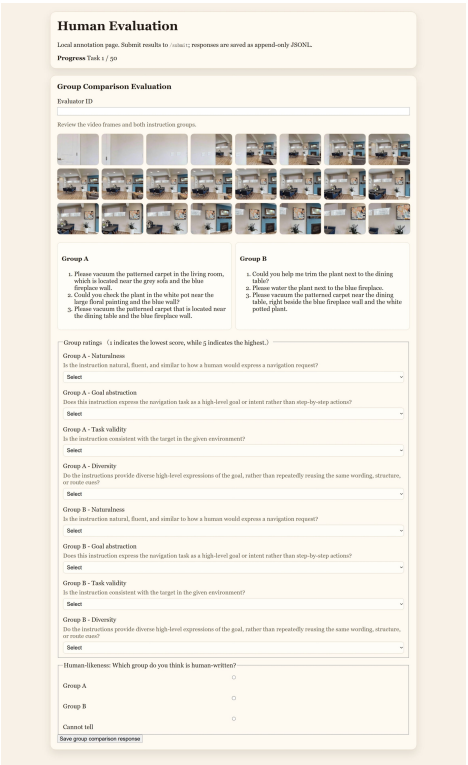
Output the subtask:

Low-Level Executor

Imagine you are a robot programmed for navigation tasks. You have been given a video of historical observations and an image of the current observation <image>. Your assigned request is: "<request>". The current subtask you need to execute is: "<subtask>". Analyze this series of images to determine your next four actions. The predicted action should be one of the following: forward, left, right, or stop.



(a) Request annotation interface.



(b) Human evaluation interface.

Figure 7: Human-facing interfaces used for RG-VLN request annotation and human evaluation. The annotation interface in (a) collects human-written requests from trajectory observations, while the evaluation interface in (b) collects blind judgments on request quality and human-likeness.

E HUMAN EVALUATION AND ANNOTATION INTERFACES

To support human-written request collection and human validity assessment, we build web interfaces for request annotation and human evaluation, as shown in Figure 7. The request annotation interface in Figure 7a presents the navigation observation sequence and asks annotators to write a natural request that implies the target endpoint through intent and spatial cues. The human evaluation interface in Figure 7b presents human-written and VLM-generated requests in an anonymized form, where volunteers rate linguistic diversity, goal abstraction, naturalness, and task validity, and additionally judge whether the request is human-written.

F LIMITATIONS AND FUTURE WORK

Limitations. Despite the promising results achieved by HiNav, several limitations remain. First, our current RG-VLN benchmark is confined to static indoor environments. However, intelligent agents should ideally demonstrate robust navigation capabilities across both indoor and outdoor dynamic settings. Second, the requests in our benchmark are fixed and do not account for human-in-the-loop adjustments. In real-world scenarios, human intent is often volatile, requiring agents to handle interruptions and dynamic re-planning. Finally, HiNav is constrained to discrete action outputs, and the serial coupling between the Request Reasoner and Executor can lead to motion stuttering, limiting the framework's adaptability in dynamic environments.

Future Work. As a foundational step toward request-grounded embodied navigation, we outline several promising directions for future research:

- **Hybrid Indoor-Outdoor Navigation:** We plan to extend our benchmark to encompass seamless transitions between indoor and outdoor environments and introduce complex dynamic entities, such as pedestrians and vehicles, evaluating proactive social-spatial interaction capabilities (Peng et al., 2026).
- **Dynamic Instruction Following:** To further challenge the agent's intelligence, we intend to introduce dynamic command updates during navigation, evaluating the agent's capacity for real-time task re-planning (Shi et al., 2025).
- **Continuous Trajectory Planning:** We aim to integrate continuous trajectory planning techniques (Janner et al., 2022; Zhou et al., 2025) to enable smoother navigation and better adaptation to complex, dynamic environments.
- **Real-world Deployment:** We anticipate that our proposed RG-VLN task and framework will contribute significantly to the development of practical service robotics and intelligent mobility aids, such as robotic guide dogs.