HiNav: A Human-Inspired Framework for Zero-Shot Vision-and-Language Navigation

Anonymous ACL submission

Abstract

Vision-and-Language Navigation (VLN) requires an agent to interpret natural language instructions and navigate complex environments. Existing approaches often fail to stop at targets due to incorrect endpoint recognition or fail to reach targets in long-distance tasks. Inspired by human navigation, we devise a solution to these challenges, proposing Human-Inspired Navigation (HiNav), a modular framework that mimics human cognitive processes for efficient navigation. HiNav integrates four components that emulate key human abilities: HiView for optimal viewpoint selection; HiMem for selective memory and map maintenance, enhancing long-range exploration; HiSpace for spatial reasoning and object relationship inference, improving endpoint recognition; and HiDecision for Large Language Model (LLM)-based path planning. We also introduce an Instruction-Object-Space (I-O-S) dataset and fine-tune the Qwen3-4B model into Qwen-Spatial (Qwen-Sp), which outperforms leading commercial LLMs (e.g., GPT-4o, Gemini-2.5-Flash, Grok3) in object list extraction, achieving higher F1 and NDCG scores on the I-O-S test set. Extensive experiments on the Room-to-Room (R2R) and REVERIE datasets demonstrate HiNav's state-of-the-art performance with significant improvements in Success Rate (SR) and Success weighted by Path Length (SPL).

1 Introduction

Humans navigate complex environments with remarkable efficiency, relying on precise observation to select informative viewpoints, selective memory to retain task-relevant information, spatial reasoning to infer object locations from linguistic cues, and precise decision-making. For example, when instructed to "reach the kitchen's refrigerator," humans visualize the kitchen layout, focus on key landmarks, and filter out irrelevant details from memory. However, VLN agents, which aim to replicate this capability by interpreting natural language instructions and analyzing visual observations, often struggle to perform effectively. Existing LLM-based Zero-Shot VLN (ZS-VLN) solutions frequently falter in complex, long-distance tasks and scenarios with ambiguous endpoints (Zhou et al., 2023; Long et al., 2023; Chen et al., 2024).

Inspired by human navigation and the failures in prior VLN solutions, we propose Human-Inspired Navigation (HiNav), a modular ZS-VLN framework that mimics human cognitive processes for efficient navigation. HiNav comprises four components: (1) HiView, drawing from how humans center objects of interest in their visual field, optimizes viewpoint selection by identifying the most informative perspectives; (2) HiMem, based on the forgetting mechanism in human memory, creates a dynamic topological map that selectively retains important spatial information while discarding outdated or irrelevant details; (3) HiSpace, reflecting human spatial imagination and reasoning capabilities, analyzes instructions and infers spatial layouts from linguistic cues, enhancing the agent's ability to understand environmental context; and (4) HiDecision, modeled after human decision-making processes, leverages advanced LLMs to determine navigation actions based on instructions, processed observations, object spatial layouts, and map information.

Additionally, we construct an Instruction-Object-Space (I-O-S) dataset, derived from oracle paths across indoor environments, to support instruction analysis and spatial reasoning. We have also fine-tuned the Qwen3-4B model(Yang et al., 2025) on this I-O-S dataset to create the Qwen-Sp model, which can analyze language instructions for VLN tasks, extract and reason about objects along the navigation path, and infer spatial layouts of objects at the destination. Extensive experiments on the Room-to-Room (R2R) and REVERIE datasets (Anderson et al., 2018; Qi et al., 2020b) demonstrate Hi-

Nav's state-of-the-art performance, with significant gains in Success Rate (SR) and Success weighted by Path Length (SPL).

Our contributions are:

- We introduce HiNav, achieving state-of-the-art performance with a 5.1% improvement in SR and 5.0% in SPL on the R2R subset (Anderson et al., 2018). We will release the HiNav codebase.
- We demonstrate the versatility of the HiSpace module, which can be seamlessly integrated into other VLN frameworks to enhance their performance, whether map-based or not.
- We present the I-O-S dataset, comprising 28,414 samples, enabling fine-grained analysis of navigation instructions. We will release this dataset to foster VLN research and LLM spatial inference.
- We develop Qwen-Sp, outperforming leading commercial LLMs (e.g., GPT-4o, Gemini-2.5-Flash, Grok3) (OpenAI, 2024; Google DeepMind, 2025; xAI, 2025) in the task of object extraction, achieving a higher F1 score (0.316 vs. 0.270 for GPT-4o) and NDCG score (0.388 vs. 0.325 for GPT-4o) on the I-O-S test set. We will release Qwen-Sp to support research on the spatial inference ability of LLMs.

2 Related Work

Vision-and-Language Navigation Vision-and-Language Navigation (VLN) requires agents to follow natural language instructions in 3D environments (Anderson et al., 2018; Krantz et al., 2020; Chen et al., 2019; Qi et al., 2020b). Early VLN research predominantly involved supervised learning, focusing on cross-modal alignment between vision and language (Hao et al., 2020; Hong et al., 2021a; Chen et al., 2021b), often leveraging visual-linguistic representations (Chen et al., 2020; Li et al., 2020). Data augmentation techniques (Fried et al., 2018; Tan et al., 2019; Wang et al., 2023) and specific training strategies (Wang et al., 2019; Huang et al., 2019) were also explored. Other research focused on state memorization (Chen et al., 2021c; Deng et al., 2020), self-correction mechanisms (Ke et al., 2019; Ma et al., 2019), and the use of external knowledge (Gao et al., 2021; Qi et al., 2020a). A significant subfield is Zero-Shot VLN (ZS-VLN), where agents navigate without task-specific training, heavily relying on Large Language Models (LLMs). NavGPT (Zhou et al., 2023) showed that LLMs can make navigation decisions from prompted inputs. DiscussNav (Long et al., 2023) used a multi-expert LLM system. MapGPT (Chen et al., 2024) integrated an online linguistic map for LLM-based global planning.

Large Language Models in VLN Large Language Models (LLMs) (Brown et al., 2020; Touvron et al., 2023; OpenAI, 2023; Yang et al., 2025) are central to modern VLN due to their strong language understanding and reasoning. In ZS-VLN, they primarily act as decision-makers (Zhou et al., 2023; Chen et al., 2024; Long et al., 2023; Huang et al., 2023). Beyond zero-shot applications, LLMs are also fine-tuned on VLN data. LangNav (Pan et al., 2023) and NavCoT (Lin et al., 2024) fine-tuned LLaMA models for navigation tasks. Effective prompting techniques (Wei et al., 2022; Kojima et al., 2022; Yao et al., 2022) remain key to LLM performance in ZS-VLN.

Human-Inspired Approaches and Navigation Maps Human-inspired navigation strategies leverage cognitive processes to enhance robotic navigation, with memory-adaptive models filtering historical data to improve decision-making (He et al., 2024), datasets like Touchdown emphasizing spatial reasoning for complex instructions (Chen et al., 2019), and VLN frameworks incorporating dynamic human activities for social navigation (Li et al., 2024). Map-based methods provide spatial memory, where metric maps built via Simultaneous Localization and Mapping (SLAM) offer detailed geometry at high computational cost (Thrun, 1998; Fuentes-Pacheco et al., 2015), and topological maps abstract environments into efficient graphs (Chen et al., 2021a, 2022). Recent efforts, such as MapGPT, integrate topological maps with LLMs to enable spatial reasoning in ZS-VLN (Chen et al., 2024).

3 Method

Task Description VLN tasks require an agent to interpret a natural language instruction $I = \{w_1, w_2, \ldots, w_L\}$ and navigate a 3D environment to a target location. At each step t, given the current pose p_t, the simulator provides several neighboring viewpoints that are currently navigable. The agent observes its state s_t, including a set of navigable viewpoints $\mathcal{V}_t = \{v_{t,i}\}_{i=1}^{K}$, where K is the number of navigable viewpoints, and visual observations

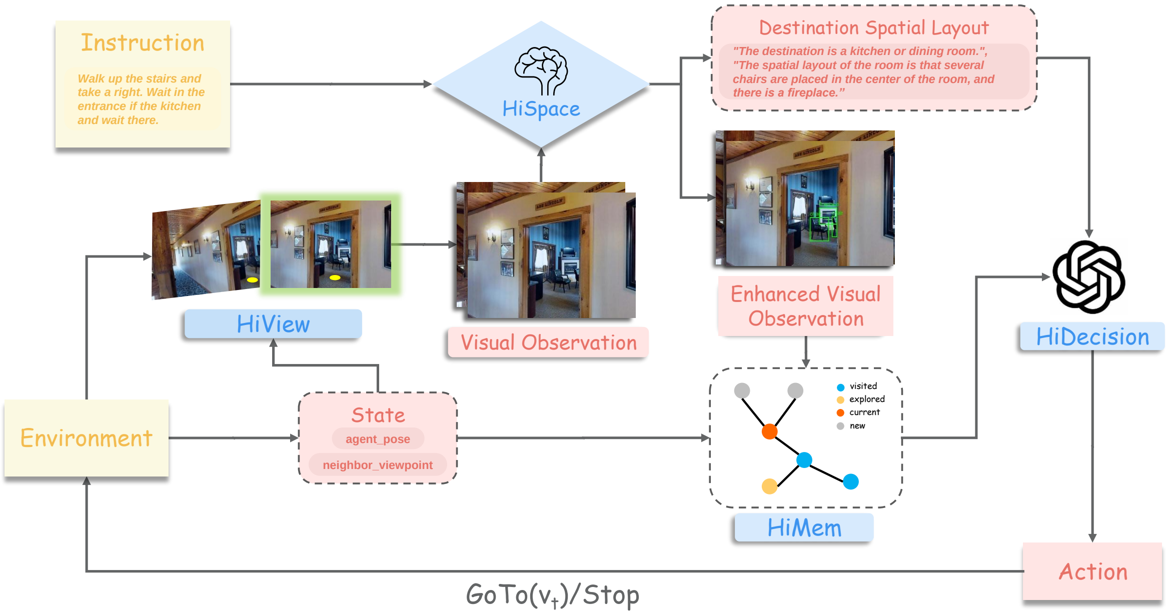


Figure 1: The HiNav architecture, integrating HiView, HiMem, HiSpace, and HiDecision to emulate human observation, memory, spatial reasoning, and decision-making. The agent receives visual observations from the environment, where HiView optimizes viewpoint selection by centering navigable viewpoints (yellow dots), enhanced by HiSpace to represent nodes in HiMem's topological map, while HiSpace processes instructions and images to generate destination spatial layouts and enhanced visual observations. HiDecision then uses these layouts and HiMem's map prompts to output navigation actions (stop or proceed to a selected viewpoint).

O, and selects an action a_t from a discrete action space A_t (e.g., navigate to an adjacent viewpoint or stop). The action is sent to the control module to execute the corresponding movement. The challenge lies in grounding linguistic instructions in visual scenes to generate an action sequence $A = \{a_1, a_2, \ldots, a_T\}$.

HiNav adopts a human-inspired modular framework that integrates four key cognitive processes: observation, memory, spatial reasoning, and decision-making. The framework comprises HiView (Section 3.1), HiMem (Section 3.2), HiSpace (Section 3.3), and HiDecision (Section 3.4), which work together to process visual and linguistic inputs efficiently, as illustrated in Figure 1. These modules emulate human navigation strategies to achieve robust and effective navigation in complex indoor environments.

3.1 HiView: Observation Module

To emulate human visual behavior of centering objects of interest (Skaramagkas et al., 2023), the HiView module selects the visual observation closest to each navigable viewpoint's direction as its representation. From a predefined set of 36 observation directions (12 horizontal at 30-degree intervals and three vertical: up, middle, down), only the direction aligned with the camera's orientation is processed, reducing computational overhead.

For a set of reachable viewpoints $\mathcal{V}_t$ from the agent's current pose p_t, HiView computes the direction vector $\vec{d} = p_c - p_t$ for each candidate viewpoint $v_c \in \mathcal{V}_t$ with position p_c. This yields the target heading θ_{tg} and elevation ϕ_{tg}. The module then identifies the optimal view index k^* from the available views, each characterized by orientations (θ_k, ϕ_k), by minimizing the L_1 angular distance:

$$k^* = \underset{k \in \text{available views}}{\arg\min} \; \mathcal{D}((\theta_k, \phi_k), (\theta_{\text{tg}}, \phi_{\text{tg}})). \quad (1)$$

The selected visual observation O_{k^*}, captured in the direction of the camera's orientation, serves as the visual representation of the viewpoint v_c after being enhanced by the HiSpace module, ensuring that the representation encapsulates sufficient spatial information and visual features for effective navigation.

3.2 HiMem: Memory Module

Humans navigate complex environments efficiently by selectively retaining task-relevant information and discarding irrelevant details through mechanisms like cognitive maps (Epstein et al., 2017) and working memory (Baddeley and Hitch, 1974; Malleret et al., 2024). Inspired by this, we investigated VLN failures, noting that existing methods (Zhou et al., 2023; Long et al., 2023; Chen et al., 2024) falter in prolonged tasks, particularly beyond 13 steps in the R2R dataset (Anderson et al., 2018). Excessive node accumulation in topological maps overwhelms LLM context limits, reducing success rates. Unlike prior approaches that retain all observations in an expanding map (Chen et al., 2022; Chen et al., 2024), our HiMem framework dynamically filters irrelevant nodes, sustaining performance and mitigating LLM context constraints. The overall workflow of HiMem is illustrated in Figure 2.

3.2.1 Map Construction

In VLN, the agent builds a real-time map of an unfamiliar environment using observations from exploration. Following prior work (Chen et al., 2022; Chen et al., 2024), we use a topological graph $G_t = (V_t, E_t)$, where $V_t = \{v_{t,i}\}_{i=1}^{K}$ represents viewpoint nodes observed up to time step t, and E_t denotes navigable connections between them.

At each step t, the agent records new viewpoints and their connections based on the simulator's feedback about neighboring nodes. These are added to an intermediate graph G_t^{tmp}, updated from the previous graph G_{t-1}. After obtaining the intermediate graph G_t^{tmp}, it is dynamically pruned to produce the final graph G_t.

3.2.2 Dynamic Map Pruning

To maintain a compact and task-relevant topological map, the HiMem module dynamically evaluates and prunes nodes from the intermediate graph G_t^{tmp} that are no longer pertinent to the navigation task. This process begins after an initial exploration phase ($t \geq t_{\text{start}}$), ensuring the map remains efficient by removing outdated or irrelevant information. By selectively filtering nodes, HiMem reduces memory overhead and mitigates interference from obsolete data, producing the final graph G_t.

The set $T_t \subseteq V_t^{\text{tmp}}$ represents all viewpoint nodes that the agent has visited up to time step t. This set tracks the agent's exploration history and is used to assess node relevance.

HiMem identifies a subset of nodes $\mathcal{V}_{\text{assess}} \subseteq V_t^{\text{tmp}}$ for relevance evaluation based on three criteria: nodes must be *non-current*, meaning they are not the agent's current viewpoint ($v_{t,i} \neq v_t$); they must be *previously visited*, having been explored ($v_{t,i} \in T_t$); and they must be *temporally stale*, not revisited recently, satisfying $t - \tau(v_{t,i}) > \theta_{\text{keep}}$ and $t - \tau(v_{t,i}) > \theta_{\text{age}}$, where $\tau(v_{t,i})$ is the time step when node $v_{t,i}$ was last visited, θ_{keep} is the minimum time elapsed since the last visit to consider a node for pruning, and θ_{age} is the threshold for determining node staleness based on its age.

Nodes in $\mathcal{V}_{\text{assess}}$ are assigned a pruning priority score $P(v_{t,i})$, which quantifies their relevance to the ongoing task:

$$P(v_{t,i}) = \lambda_t f_t(v_{t,i}) + \lambda_d f_d(v_{t,i}) + \lambda_f f_f(v_{t,i}) + \lambda_{\text{dist}} f_{\text{dist}}(v_{t,i}) \quad (2)$$

where:

- $f_t(v_{t,i}) = \max(1, t - \tau(v_{t,i}) - \theta_{\text{age}})$: Measures temporal staleness, prioritizing older nodes.
- $f_d(v_{t,i}) = -\deg_{G_t^{\text{tmp}}}(v_{t,i})$: Penalizes nodes with low connectivity, as they are less critical to navigation.
- $f_f(v_{t,i}) = -|\{v_{t,j} \mid (v_{t,i}, v_{t,j}) \in E_t^{\text{tmp}} \wedge v_{t,j} \notin T_t\}|$: Favors nodes with fewer unexplored neighbors, indicating lower exploration potential.
- $f_{\text{dist}}(v_{t,i}) = d_{G_t^{\text{tmp}}}(v_t, v_{t,i})$: Considers the graph distance from the current viewpoint, prioritizing distant nodes.

The coefficients $\lambda_t, \lambda_d, \lambda_f, \lambda_{\text{dist}}$ balance the contributions of each factor. Details on the specific selection of coefficients are provided in the Appendix A.2.

Based on the pruning priority scores, the top N_{remove} nodes with the highest $P(v_{t,i})$ are removed from G_t^{tmp}, yielding the final map $G_t = (V_t, E_t)$. This selective pruning ensures that the topological map remains concise, relevant, and computationally efficient, supporting robust navigation over extended periods.

3.2.3 Map Representation

The HiMem module structures the filtered topological map $G_t = (V_t, E_t)$ into prompts for the HiDecision module, adapting insights from prior work (Chen et al., 2024). The prompts include: (1) Trajectory, listing visited node identifiers in V_t; (2) Map, detailing node connectivity in E_t; and (3) Supplementary Information, linking nodes v_t to

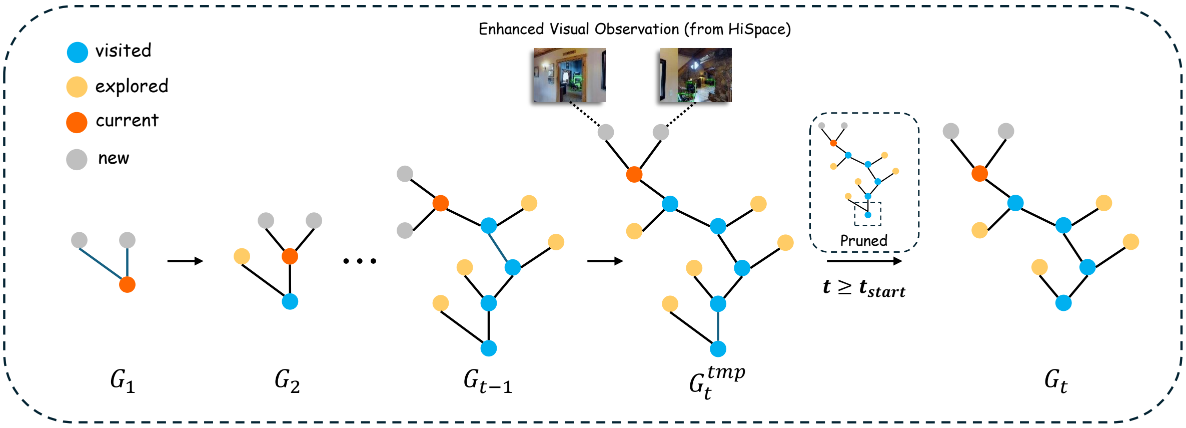


Figure 2: The HiMem architecture, illustrating the dynamic construction and pruning of a task-relevant topological map. At step t, HiMem observes navigable viewpoints and uses their enhanced visual observations as representations, adding them as new nodes (gray) to the previous map G_{t-1} to form an intermediate map G_t^{tmp}; if $t \geq t_{\text{start}}$, a pruning operation is triggered, removing N_{remove} nodes based on their pruning priority scores ($N_{\text{remove}} = 1$ in this figure), resulting in a compact map G_t.

enhanced visual observations $O^e_{k^*}$ from HiSpace. This ensures a concise, task-relevant spatial representation for the LLM. Detailed prompts are in Appendix B.

3.3 HiSpace: Spatial Reasoning Module

Humans navigate by recognizing landmarks and mentally constructing spatial configurations at destinations, enabling precise location identification. In VLN, agents often misidentify targets, such as stopping in a hallway instead of a kitchen in R2R tasks (Anderson et al., 2018), due to weak spatial reasoning. HiSpace addresses this by extracting task-relevant objects from instructions and inferring destination layouts. Using our I-O-S dataset, we fine-tuned Qwen3-4B into Qwen-Sp to generate accurate object lists and layouts, boosting navigation precision. The HiSpace architecture is shown in Figure 3. Beyond VLN, the I-O-S dataset also enhances LLMs' spatial imagination and reasoning capabilities.

3.3.1 I-O-S Dataset

The Instruction-Object-Space (I-O-S) dataset is a novel resource designed to enhance spatial reasoning in VLN by providing structured data that links natural language instructions to objects and their spatial arrangements. Comprising 28,414 samples derived from expert trajectories in indoor environments, the I-O-S dataset captures three key components: (1) Instructions, which are natural language navigation directives; (2) Objects, a list of task-relevant objects encountered along the trajectory or at the destination; and (3) Destination Spatial Layouts, describing the relative positions of objects at the destination (e.g., "The spatial layout of the room is that several chairs are placed in the center of the room, and there is a fireplace").

To construct the dataset, we extracted oracle paths from indoor environments, which provide optimal navigation trajectories. Objects along the path and at the destination were identified using the simulator's ground-truth object annotations. Spatial arrangements were generated through a two-step process: first, an LLM proposed candidate layouts based on object relationships observed in the destination scenes; second, human annotators verified and refined these layouts to ensure accuracy and consistency.

Each sample in the I-O-S dataset is formatted as a tuple (I, O, S), where I is the instruction, O is the set of objects, and S is the description of destination spatial layout. The dataset is split into 25,694 training samples and 2,720 test samples. By providing fine-grained annotations, the I-O-S dataset enables models to learn how to extract task-relevant objects from instructions and infer their spatial configurations. See Appendix D for details.

3.3.2 Spatial Reasoning Model

To enable robust spatial reasoning, we developed Qwen-Sp by fine-tuning Qwen3-4B (Yang et al., 2025) on the I-O-S dataset using Low-Rank Adaptation (LoRA) (Hu et al., 2022). Qwen-Sp employs two LoRA adapters: one to extract task-relevant objects from navigation instructions (e.g., identifying

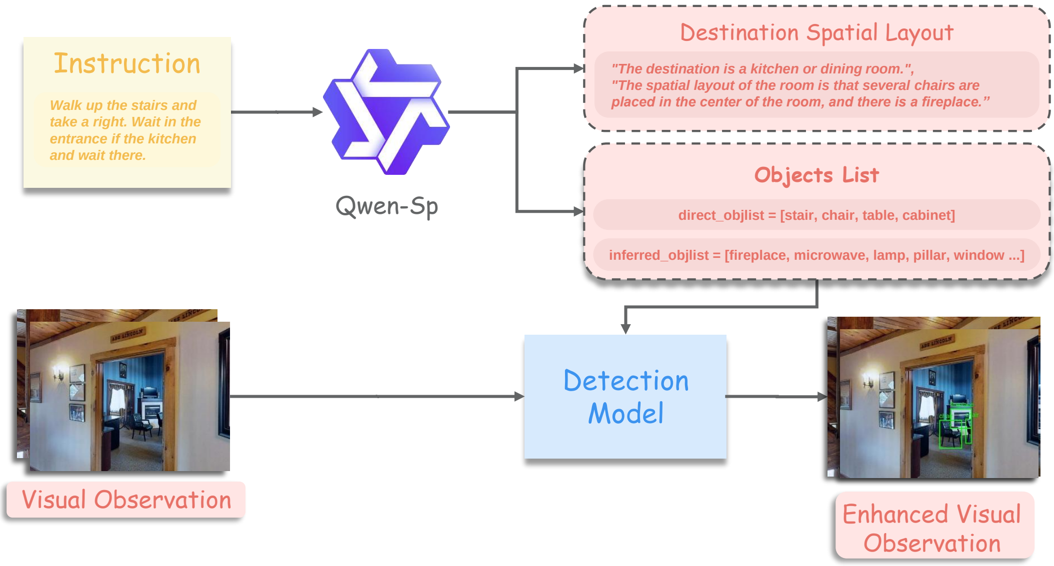


Figure 3: The HiSpace architecture, depicting the pipeline for spatial reasoning. Qwen-Sp processes instructions to extract object lists and infer destination spatial layouts, while the detection model, implemented using YOLO-World, enhances visual observations to enable the agent to better identify and navigate toward task-relevant objects.

"refrigerator" from "go to the kitchen's refrigerator") and another to infer their spatial arrangements at the destination (e.g., "the refrigerator is against the kitchen's back wall"). Fine-tuned on 25,694 I-O-S samples, Qwen-Sp achieves superior performance in object list extraction compared to leading commercial LLMs, including GPT-4o, Gemini-2.5-Flash, and Grok3 (xAI, 2025). This highlights Qwen-Sp's ability to accurately identify and prioritize task-relevant objects, which is critical for effective navigation. For zero-shot REVERIE experiments (Qi et al., 2020b), we avoid direct use of the fine-tuned model, instead leveraging its learned patterns to design prompts for commercial LLMs (e.g., GPT-4o). Qwen-Sp is to be released open-source, with training details provided in Appendix A.4.

3.3.3 Visual Input Enhancement

The Visual Input Enhancement component enables landmark-based pathfinding by enhancing visual observations with task-relevant objects from the spatial reasoning model's object list, mimicking how humans use landmarks to navigate (Skaramagkas et al., 2023). Using YOLO-World (Cheng et al., 2024), a lightweight, fast, and high-performance open-vocabulary object detection system, we annotate objects (e.g., "chair", "fireplace") in the visual observation O_{k^*} selected by HiView, guiding the agent along the instruction-specified path. The enhanced inputs $O^e_{k^*}$, integrating visual and textual object information, inform the HiDecision module's LLM, improving navigation precision. Performance impacts are reported in Section 4.

3.4 HiDecision: Decision-Making Module

LLMs exhibit reasoning and decision-making capabilities that, to a certain extent, parallel those of humans. To harness these capabilities, we introduce HiDecision, a module that employs an advanced LLM, GPT-4o, to facilitate high-level decision-making. At each time step t, HiDecision processes the following inputs: the natural language instruction (I), specifying the navigation goal; the HiMem context (C_t^{HiMem}), including the trajectory and the map; the destination spatial layout (SL^{HiSpace}) provided by HiSpace; and other prompt information (e.g., history, previous planning, and action options, denoted as P_t). The LLM integrates these inputs to output an action a_t, which is either the selection of a neighboring viewpoint or a decision to stop:

$$a_t = \text{HiDecision}(I, C_t^{\text{HiMem}}, SL^{\text{HiSpace}}, P_t). \quad (3)$$

The complete prompt structure is detailed in Appendix B.

4 Experiments

4.1 Experimental Settings

HiNav is evaluated on the R2R (Anderson et al., 2018) and REVERIE (Qi et al., 2020b) datasets in a zero-shot setting, with the I-O-S dataset used to assess Qwen-Sp's spatial inferring ability. HiNav is compared to NavGPT (Zhou et al., 2023), DiscussNav (Long et al., 2023), and MapGPT (Chen et al., 2024), using GPT-4o for a fair comparison. Qwen-Sp is also tested against GPT-4o, Gemini-2.5-Flash, and Grok3 on the I-O-S dataset. We conduct VLN experiments on the Matterport3D simulator(Chang et al., 2017). The implementation details are in Appendix A.

Evaluation Metrics Performance is assessed using the following metrics. For VLN tasks: (1) *Success Rate (SR)*, the percentage of successful episodes; (2) *Success weighted by Path Length (SPL)*, which balances success and path efficiency; (3) *Oracle Success Rate (OSR)*, the SR with an oracle stop policy; and (4) *Navigation Error (NE)*, the average distance in meters to the target. For evaluating the spatial inference capabilities of LLMs: (5) *F1 Score*, measuring precision and recall for object list extraction; and (6) *Normalized Discounted Cumulative Gain (NDCG)*, assessing the ranking quality of extracted objects. Additionally, we introduce a novel metric, *Map Efficiency (ME)*, which reflects the HiMem module's ability to maintain task-relevant spatial information. Details of these metrics are in Appendix A.

4.2 Experimental Results

ZS-VLN Benchmark Comparison Following prior work (Zhou et al., 2023; Chen et al., 2024), we evaluate HiNav on the standard R2R subset consisting of 72 scenes and 216 samples, as shown in Table 1. HiNav achieves an SR of 50.9% and SPL of an 42.6%, outperforming MapGPT by 5.1% and 5.0%, respectively. HiView's viewpoint optimization captures critical landmarks, providing the most complete visual representation of the viewpoint. HiMem's pruning maintains compact maps, with an ME of 40.4% , enabling stable exploration in long trajectories. Notably, HiNav's higher OSR (7.3% higher than MapGPT) likely stems from HiMem's pruning, facilitating late-stage exploration without increased resource demands.

REVERIE Complex Task Evaluation In line with prior benchmarks (Chen et al., 2024), we evaluate HiNav on a randomly sampled subset of the REVERIE dataset, containing 70 scenes and 140 samples, as shown in Table 2. HiNav achieves an SR of 45.7% and an SPL of 32.8%, outperforming MapGPT by 4.3% and 4.4%, respectively. The HiMem module demonstrates significant proficiency in handling complex, long-range tasks inherent in REVERIE. However, HiNav's ME shows limited improvement, likely due to the richness of REVERIE instructions, which allow even frameworks less adept at long-range tasks to construct comprehensive maps by leveraging detailed information. Moreover, HiSpace effectively leverages the dataset's diverse object categories to enhance task-relevant object extraction, while HiView provides comprehensive observations that facilitate more effective landmark detection.

Methods	SR ↑	SPL ↑	OSR ↑	NE ↓	ME ↑
NavGPT (Zhou et al., 2023)	36.1	31.6	40.3	6.26	-
DiscussNav (Long et al., 2023)	37.5	33.3	51.0	6.30	-
MapGPT (Chen et al., 2024)	45.8	37.6	56.5	5.31	38.0
HiNav (Ours)	**50.9**	**42.6**	**63.9**	**5.02**	**40.4**

Table 1: Comparison of ZS-VLN performance on the standard R2R subset (72 scenes, 216 samples). NavGPT and MapGPT results are reproduced using GPT-4o to ensure a fair comparison.

Methods	SR ↑	SPL ↑	OSR ↑	NE ↓	ME ↑
NavGPT (Zhou et al., 2023)	28.9	23.0	32.6	7.86	-
MapGPT (Chen et al., 2024)	41.4	28.4	56.4	7.12	34.7
HiNav (Ours)	**45.7**	**32.8**	**59.3**	**7.89**	**35.6**

Table 2: Comparison of ZS-VLN performance on a randomly sampled REVERIE subset (70 scenes, 140 samples). NavGPT and MapGPT results are reproduced using GPT-4o to ensure fair comparison. To maintain the zero-shot evaluation setting, HiNav employs GPT-4o instead of Qwen-Sp for the HiSpace module in this experiment, as the I-O-S dataset includes samples derived from the REVERIE dataset.

R2R Large-Scale Evaluation To compare with prior VLN and ZS-VLN work, we evaluate HiNav on the R2R full validation unseen set (11 scenes, 783 samples), as shown in Table 3. HiNav achieves an SR of 46% and an SPL of 40%, surpassing MapGPT by 2% and 5%, respectively. The relatively moderate improvements observed here can be attributed to the limited scene diversity within the 11-scene subset, which restricts the effectiveness of HiMem's pruning and HiSpace's spatial reasoning capabilities. However, HiNav's SR outperforms three trained and pretrained methods, achieving state-of-the-art zero-shot performance.

LLM Spatial Inference Comparison We evaluate Qwen-Sp's spatial inference capability against leading LLMs (GPT-4o, Gemini-2.5-Flash, Grok3) as well as the baseline non-fine-tuned Qwen3-4B model on the I-O-S test set consisting of 2,720 sam-

Settings	Methods	SR	SPL	OSR	NE
Train	Seq2Seq (Anderson et al., 2018)	21	-	28	7.81
	Speaker (Fried et al., 2018)	35	-	45	6.62
	EnvDrop (Tan et al., 2019)	52	48	-	5.22
Pre-train	LangNav (Pan et al., 2023)	43	-	-	-
	PREVALENT (Hao et al., 2020)	58	53	-	4.71
	RecBERT (Hong et al., 2021b)	63	57	69	3.93
	HAMT (Chen et al., 2021c)	66	61	73	2.29
	DUET (Chen et al., 2022)	72	60	81	3.31
	ScaleVLN (Wang et al., 2023)	81	70	88	2.09
ZS	NavGPT (Zhou et al., 2023)	34	29	42	6.46
	DiscussNav (Long et al., 2023)	43	40	61	5.32
	MapGPT (Chen et al., 2024)	44	35	58	5.63
	HiNav (Ours)	**46**	**40**	**65**	**5.24**

Table 3: Performance comparison on the complete validation unseen set of the R2R dataset (11 scenes, 783 samples). HiNav achieves the highest SR among all zero-shot methods and surpasses three trained and pre-trained approaches.

Model	F1DO ↑	F1IO ↑	F1 ↑	NDCG ↑
GPT-4o	0.258	0.150	0.270	0.325
Grok3	0.055	0.057	0.096	0.095
Gemini-2.5-Flash	0.023	0.055	0.096	0.106
Qwen3-4B	0.236	0.039	0.138	0.198
Qwen-Sp (Ours)	**0.357**	**0.179**	**0.316**	**0.388**

Table 4: Comparative evaluation of object extraction capabilities of different LLMs on the I-O-S test set (2,720 samples). F1DO and F1IO represent the F1 scores for direct and inferred objects, respectively. Qwen-Sp outperforms other models across all metrics.

ples, as shown in Table 4. Qwen-Sp, fine-tuned on the I-O-S training set (25,694 samples), achieves an F1 score of 0.316 and an NDCG of 0.388, surpassing GPT-4o by 0.046 and 0.063, respectively. Non-fine-tuned LLMs employed a one-shot prompt for evaluation, as shown in Appendix B. The superior performance of Qwen-Sp underscores the effectiveness of targeted fine-tuning in enhancing task-relevant object identification. In contrast, GPT-4o exhibits robust spatial reasoning without specialized training. Interestingly, Grok3 and Gemini-2.5-Flash considerably underperform, even relative to the much smaller-scale Qwen3-4B, highlighting notable limitations in spatial inference among these large models and emphasizing the potential advantages of smaller LLMs. While HiSpace significantly enhances object layout inference from diverse instructions, the indoor-centric I-O-S dataset may constrain Qwen-Sp's applicability to outdoor environments, suggesting the potential value of expanding the dataset scope in future work.

Methods	SR ↑	SPL ↑	OSR ↑	NE ↓
NavGPT (Zhou et al., 2023)	36.1	31.6	40.3	6.26
NavGPT+HiSpace	38.9	34.1	43.1	5.96
MapGPT (Chen et al., 2024)	45.8	37.6	56.5	5.31
MapGPT+HiSpace	48.1	39.6	58.3	5.11
HiNav w/o HiView	49.5	41.4	62.5	5.17
HiNav w/o HiMem	48.1	40.1	58.8	5.32
HiNav w/o HiSpace	47.7	39.6	61.1	5.37
HiNav	**50.9**	**42.6**	**63.9**	**5.02**

Table 5: Ablation study on the R2R subset (72 scenes, 216 samples). This table presents the performance of HiNav with individual modules removed, and demonstrates the performance improvement achieved by integrating the HiSpace module into other frameworks. For NavGPT, which converts visual inputs to text using a grounding model, only HiSpace's Destination Spatial Layout was incorporated.

4.3 Ablation Study

As shown in Table 5, we conducted ablation experiments on the R2R subset to evaluate HiNav's modules and integrated our pluggable HiSpace module into other zero-shot VLN frameworks. For NavGPT, which processes visual inputs into text via a grounding model, only HiSpace's Destination Spatial Layout was applied. Results confirm the effectiveness of each HiNav module. Notably, removing HiMem significantly reduces OSR, as its proficiency in long-range exploration enhances success under oracle stopping conditions, aligning with its design. Incorporating HiSpace into other frameworks via simple prompt modifications yields substantial improvements, demonstrating its effectiveness and seamless transferability across map-based and non-map-based VLN frameworks.

5 Conclusion

This paper introduces HiNav, a novel zero-shot vision-and-language navigation (ZS-VLN) framework that enhances navigation by emulating human cognitive processes through its modular architecture. HiNav demonstrates state-of-the-art performance on the R2R and REVERIE datasets, and its HiSpace module offers versatile plug-and-play integration to augment existing VLN frameworks. To further advance spatial understanding in large models, we introduced the Instruction-Object-Space (I-O-S) dataset. Leveraging this resource, we fine-tuned Qwen3-4B to develop Qwen-Sp, a model that demonstrably surpasses leading commercial LLMs like GPT-4o in critical instruction analysis and object extraction tasks.

Limitations

Despite HiNav's strong performance in ZS-VLN tasks, the sub-optimality of single-pass LLM decisions persists, a challenge clearly evidenced by the performance gap compared to idealized iterative correction (detailed in Appendix A.1). This underscores the significant potential to enhance LLM performance in ZS-VLN tasks by stabilizing and optimizing their decision outputs. Furthermore, while our I-O-S dataset aims to bolster LLM spatial reasoning, evaluating its impact on inferred spatial layouts is methodologically challenging. Unlike object list extraction, which uses direct metrics, spatial layout assessment currently relies on indirect validation through VLN task performance, highlighting the need for dedicated metrics for a more direct and streamlined evaluation of this capability.

References

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. 2018. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3674–3683.

Alan D. Baddeley and Graham Hitch. 1974. Working memory. In *The Psychology of Learning and Motivation*, volume 8, pages 47–89. Elsevier.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901.

Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Nießner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. 2017. Matterport3d: Learning from rgb-d data in indoor environments. In *Proceedings of the IEEE International Conference on 3D Vision (3DV)*.

Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. 2019. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12538–12547.

Jiaqi Chen, Bingqian Lin, Ran Xu, Zhenhua Chai, Xiaodan Liang, and Kwan-Yee K. Wong. 2024. 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*.

Kevin Chen, Junshen K. Chen, Jo Chuang, Marynel Vázquez, and Silvio Savarese. 2021a. Topological planning with transformers for vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11276–11286.

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

Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. 2021c. History aware multimodal transformer for vision-and-language navigation. *arXiv preprint arXiv:2104.01814*.

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

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Uniter: Universal image-text representation learning. In *European Conference on Computer Vision*, pages 104–120. Springer.

Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. 2024. Yolo-world: Real-time open-vocabulary object detection. *arXiv preprint arXiv:2401.17270*.

Zhiwei Deng, Karthik Narasimhan, and Olga Russakovsky. 2020. Evolving graphical planner: Contextual global planning for vision-and-language navigation. *Advances in Neural Information Processing Systems*, 33:20660–20672.

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

Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. 2018. Speaker-follower models for vision-and-language navigation. *Advances in Neural Information Processing Systems*, 31.

Jorge Fuentes-Pacheco, Josue Ruiz-Ascencio, and Juan Manuel Rendon-Mancha. 2015. Visual simultaneous localization and mapping: A survey. *Artificial Intelligence Review*, 43:55–81.

Chen Gao, Jinyu Chen, Si Liu, Luting Wang, Qiong Zhang, and Qi Wu. 2021. Room-and-object aware knowledge reasoning for remote embodied referring expression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3064–3073.

Google DeepMind. 2025. Start building with gemini 2.5 flash. `https://developers.googleblog.com/en/start-building-with-gemini-25-flash/`. Accessed: 2025-05-19.

Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. 2020. Towards learning a generic agent for vision-and-language navigation via pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13137–13146.

Keji He, Ya Jing, Yan Huang, Zhihe Lu, Dong An, and Liang Wang. 2024. Memory-adaptive vision-and-language navigation. *Pattern Recognition*, 153:110511.

Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. 2021a. VLN BERT: A recurrent vision-and-language BERT for navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1643–1653.

Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. 2021b. Vln bert: A recurrent vision-and-language bert for navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1643–1653.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Haoshuo Huang, Vihan Jain, Harsh Mehta, Alexander Ku, Gabriel Magalhaes, Jason Baldridge, and Eugene Ie. 2019. Transferable representation learning in vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7403–7412. IEEE.

Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. 2023. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*.

Liyiming Ke, Xiujun Li, Yonatan Bisk, Ari Holtzman, Zhe Gan, Jingjing Liu, Jianfeng Gao, Yejin Choi, and Siddhartha Srinivasa. 2019. Tactical rewind: Self-correction via backtracking in vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6741–6749. IEEE.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*.

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

Heng Li, Minghan Li, Zhi-Qi Cheng, Yifei Dong, Yuxuan Zhou, Jun-Yan He, Qi Dai, Teruko Mitamura, and Alexander G. Hauptmann. 2024. Human-aware vision-and-language navigation: Bridging simulation to reality with dynamic human interactions. *arXiv preprint arXiv:2406.19236*. Spotlight at NeurIPS 2024 Datasets and Benchmarks Track.

Xiujun Li, Xi Yin, Chun Yuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 121–137. Springer.

Bingqian Lin, Yunshuang Nie, Ziming Wei, Jiaqi Chen, Shikui Ma, Jianhua Han, Hang Xu, Xiaojun Chang, and Xiaodan Liang. 2024. Navcot: Boosting llm-based vision-and-language navigation via learning disentangled reasoning. *arXiv preprint arXiv:2403.07376*.

Yuxing Long, Xiaoqi Li, Wenzhe Cai, and Hao Dong. 2023. Discuss before moving: Visual language navigation via multi-expert discussions. *arXiv preprint arXiv:2309.11382*.

Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, and Zsolt Kira. 2019. The regretful agent: Heuristic-aided navigation through progress estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6732–6740.

Gaël Malleret, Paul Salin, Stéphanie Mazza, and Gaëlle Plancher. 2024. Working memory forgetting: Bridging gaps between human and animal studies. *Neuroscience & Biobehavioral Reviews*, 163:105742.

OpenAI. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

OpenAI. 2024. Gpt-4o system card. `https://arxiv.org/abs/2410.21276`. Accessed: 2025-05-19.

Bowen Pan, Rameswar Panda, SouYoung Jin, Rogerio Feris, Aude Oliva, Phillip Isola, and Yoon Kim. 2023. Langnav: Language as a perceptual representation for navigation. *arXiv preprint arXiv:2310.07889*.

Yuankai Qi, Zizheng Pan, Shengping Zhang, Anton van den Hengel, and Qi Wu. 2020a. Object-and-action aware model for visual language navigation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 303–317. Springer.

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

Vasileios Skaramagkas, Giorgos Giannakakis, Emmanouil Ktistakis, Dimitris Manousos, Ioannis Karatzanis, Nikolaos S. Tachos, Evanthia Tripoliti, Kostas Marias, Dimitrios I. Fotiadis, and Manolis Tsiknakis. 2023. Review of eye tracking metrics involved in emotional and cognitive processes. *IEEE Reviews in Biomedical Engineering*, 16:260–277.

Hao Tan, Licheng Yu, and Mohit Bansal. 2019. Learning to navigate unseen environments: Back translation with environmental dropout. pages 2610–2621, Minneapolis, Minnesota.

Sebastian Thrun. 1998. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, 99(1):21–71.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. 2019. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6630–6639. IEEE.

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

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

xAI. 2025. Grok 3 beta — the age of reasoning agents. Accessed: 2025-05-19.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. In *arXiv preprint arXiv:2505.09388*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Gengze Zhou, Yicong Hong, and Qi Wu. 2023. Navgpt: Explicit reasoning in vision-and-language navigation with large language models. *arXiv preprint arXiv:2305.16986*.

Appendices

A Experiment Details

A.1 Analysis of HiNav Potential with Multi Round Experiment

While HiNav demonstrates strong zero-shot performance (Table 1), we further conducted an exploratory analysis to estimate its potential upper bound. This iterative refinement process, with round-by-round results detailed in Table 6, showed that by cumulatively refining performance—in each round, we re-evaluated samples that had failed in the previous round's cumulative results, and then updated the overall results with these new outcomes—HiNav's SR could be significantly enhanced. This process was halted after five rounds because the improvement in Oracle Success Rate (OSR) became marginal (increasing from 76.4% in the fourth round to 77.8% in the fifth round). This saturation suggested that the remaining failures were largely due to episodes where the target was fundamentally unreachable by the agent, rather than sub-optimal local decisions. Through this iterative refinement, HiNav's SR was progressively improved from its initial 50.9% to a remarkable 73.6%, with a corresponding increase in SPL from 42.6% to 59.3%.

This significant gap primarily highlights the current variability and sub-optimality in the LLM's decision-making process for VLN tasks. While strategies such as reducing LLM temperature can enhance output consistency, they often bias the agent towards overly conservative actions (e.g., premature STOP decisions), which can be detrimental, particularly in long-horizon navigation tasks. Nevertheless, achieving a 73.6% SR through the described iterative refinement—a figure that surpasses many fully supervised or pretrained methods—underscores the immense, albeit not fully realized, potential of LLMs in this domain. This strongly suggests that future efforts focused on optimizing the LLM's output for navigational decision-making are crucial for substantially advancing the

Iteration Round	SR ↑	SPL ↑	OSR ↑	NE ↓
1 (Base HiNav)	50.9	42.6	63.9	5.02
2	62.5	51.3	70.8	4.46
3	67.1	54.3	74.5	4.31
4	69.9	56.2	76.4	4.05
5	73.6	59.3	77.8	3.67

Table 6: HiNav performance progression on the R2R subset (72 scenes, 216 samples) with 5 iterative rounds. Each round addresses failures from the preceding one.

performance of LLM-based ZS-VLN systems like HiNav.

A.2 Himem Details

The HiMem module's dynamic map pruning, described in Section 3.2, uses optimized parameters to maintain a compact topological map. The parameters include $t_{\text{start}} = 15$, $\theta_{\text{keep}} = 3$, $\theta_{\text{age}} = 10$, $N_{\text{remove}} = 1$, $\lambda_t = 1.0$, $\lambda_d = 2.0$, $\lambda_f = 5.0$, and $\lambda_{\text{dist}} = 0.5$. These values, tuned empirically via grid search on the R2R subset (Anderson et al., 2018), balance map compactness and navigation efficiency. Validated on R2R and REVERIE (Qi et al., 2020b), these settings achieved ME scores of 40.4% and 35.6%, respectively (Section 4).

A.3 Metric Details

This section provides definitions for all evaluation metrics used in the experiments to ensure clarity and reproducibility. Standard VLN metrics (NE, SR, OSR, SPL) follow established definitions (Anderson et al., 2018), while F1 and NDCG are tailored to the I-O-S dataset's object extraction task. The novel Map Efficiency (ME) metric is detailed to highlight its role in evaluating topological map quality.

Navigation Error (NE) NE measures the average Euclidean distance (in meters) between the agent's final position and the target location at the end of an episode.

Success Rate (SR) SR is the percentage of episodes where the agent stops within 3 meters of the target location.

Oracle Success Rate (OSR) OSR is the percentage of episodes where the agent passes within 3 meters of the target at any point during navigation, assuming an oracle stop policy.

Success weighted by Path Length (SPL) SPL balances navigation success and path efficiency, computed as the ratio of the shortest path length to the actual path length, weighted by success.

F1 Score The F1 score measures the precision and recall of extracted object lists in the I-O-S dataset. It is computed separately for direct objects (F1DO, objects explicitly mentioned in instructions) and inferred objects (F1IO, objects implicitly relevant based on context). The overall F1 score is calculated for the entire object list, which combines both direct and inferred objects into a single set.

Normalized Discounted Cumulative Gain (NDCG) NDCG assesses the ranking quality of extracted objects by comparing the predicted object list to the ground-truth list. It accounts for the relevance order of objects, with higher scores indicating better alignment with the ground-truth ranking. For each sample, NDCG is calculated as:

$$\text{NDCG} = \frac{\text{DCG}}{\text{IDCG}}, \tag{4}$$

where DCG is the discounted cumulative gain based on predicted object ranks, and IDCG is the ideal DCG based on the ground-truth ranks.

Map Efficiency (ME) The Map Efficiency (ME) metric evaluates the quality of topological maps in VLN tasks. It is defined as:

$$\text{ME} = \frac{|T_t \cap T_{\text{expert}}|}{|T_{\text{expert}}|} \cdot \frac{1}{1 + \alpha \cdot \frac{|V_t|}{|T_{\text{expert}}|}}, \tag{5}$$

where T_t represents the agent's trajectory, T_{expert} denotes the expert path node set, V_t is the agent's map node set, and $\alpha = 0.25$. The first term quantifies the proportion of expert nodes covered by the trajectory, while the second term penalizes overly large maps. A higher ME score indicates a more compact and efficient map. HiNav's HiMem pruning strategy yields superior ME scores, demonstrating enhanced map efficiency. The penalty factor $\alpha = 0.25$ was optimized through a grid search over the range [0.1, 1.0] using the R2R subset.

A.4 Qwen-Sp Fine-tuning Details

This study employs the pretrained Qwen3-4B, a causal language model with approximately 4 billion parameters, as the base model. To adapt it for instruction-following and scene understanding tasks in vision-language navigation (VLN), we utilize Low-Rank Adaptation (LoRA), a parameter-efficient fine-tuning (PEFT) technique. Specifically, we train two independent LoRA adapters on the I-O-S dataset, comprising 25,694 training samples: the Object Adapter, which predicts task-relevant object pairs from instructions, and the Spatial Relation Adapter, which infers spatial relationships and overall layouts among objects.

To ensure consistency and comparability, both adapters share the same LoRA configuration. The rank of the low-rank matrices is set to $r = 16$, the scaling factor to $\alpha = 32$, and the dropout rate for LoRA layers to 0.05. The LoRA adapters are applied to key layers of the Qwen3 model, specifically the projection layers of the multi-head self-attention (MHSA) mechanism—namely, the query (q_{proj}), key (k_{proj}), value (v_{proj}), and output (o_{proj}) projections—as well as the linear layers of the feed-forward network (FFN), comprising the gate ($gate_{\text{proj}}$), up (up_{proj}), and down ($down_{\text{proj}}$) projections in the SwiGLU-based FFN. These layers are selected due to their critical role in instruction understanding, as MHSA layers effectively model complex dependencies within text sequences, and FFN layers enable nonlinear transformations and high-level feature abstraction. This targeted application of LoRA facilitates efficient learning of task-specific patterns while minimizing computational and storage requirements.

The fine-tuning process employs the AdamW optimizer in PyTorch, with an initial learning rate of 1×10^{-4}, a cosine decay schedule, and a weight decay of 0.1 for L2 regularization. The maximum gradient norm is clipped at 1.0, and training is performed using bfloat16 (bf16) mixed precision without gradient accumulation. The Object Adapter is trained for 1 epoch with a per-device batch size of 64, while the Spatial Relation Adapter is trained for 8 epochs with a per-device batch size of 48.

B Prompt Structures

ZS-VLN Prompts This section describes the prompt structures employed by HiNav to guide the LLM in ZS-VLN tasks. The overall prompt architecture is depicted in Figure 4. The task description prompt is elaborated in Figure 5, while the single-round prompt input to the HiDecision module is shown in Figure 6. For experiments on the REVERIE dataset, only the instruction component is modified to:

> “‘Instruction’ serves as global guidance that you should follow. Your task is to locate the specified or hidden target object, stop, and disregard any actions related to the target object mentioned in the ‘Instruction’. You should not overly focus on color details of landmarks or the target object described in the ‘Instruction’, as these color descriptions may be inaccurate.”

Query to LLM
Task Description (System-level prompt, provided to LLM only at the start):
[Task Background] [Input Definitions] [Output Requirements]
Prompts of HiDecision (Provided to LLM in every round):
[Instruction]
[History]
[Trajectory] [Map] [Supplementary Info] (from HiMem)
[Previous Planning]
[Possible Destination Info] (from HiSpace)
[Action Options]

LLM Response
[Thought]
[New Planning]
[Action]

Figure 4: Complete prompt structure. The top section specifies the static system-level task description prompt provided to the LLM at the outset. The middle section elaborates on the dynamic prompts supplied during each navigation round. The bottom section presents the LLM’s response.

All other components remain consistent with those used for the R2R dataset. Our prompt design draws on insights from prior work (Chen et al., 2024) while incorporating adaptations tailored to HiNav’s framework.

Spatial Inferring Prompts To evaluate the spatial imagination and reasoning capabilities of leading commercial large language models (LLMs), we utilize the prompt shown in Figure 7 to evaluate their ability to extract objects from instructions. For experiments on the REVERIE dataset, to ensure a zero-shot setting, we refrain from using our Qwen-Sp model. Instead, we employ GPT-4o for extracting the object list and predicting the destination spatial layout. The corresponding prompts are presented in Figures 7 and 8, respectively.

C Qualitative Analysis of HiNav

C.1 Successful Case Study

This successful case demonstrates that the HiNav framework effectively tackles complex instruction-guided navigation problems for challenging pathfinding tasks. It achieves this by integrating several key capabilities: HiMem’s dynamic pruning, HiSpace’s spatial relationship

Task Description

Task Background

You are an embodied robot that navigates in the real world. You need to explore between some places marked with IDs and ultimately find the destination to stop. At each step, a series of images corresponding to the places you have explored and have observed will be provided to you. **Target detection boxes in images may highlight objects relevant to the optimal navigation path, so take them into account as needed.**

Input Definitions

'Instruction' is a global, step-by-step detailed guidance, but you might have already executed some of the commands. You need to carefully discern the commands that have not been executed yet.
'History' represents the places you have explored in previous steps along with their corresponding images. It may include the correct landmarks mentioned in the 'Instruction' as well as some past erroneous explorations. Due to map optimization, images for older places in history may be placeholders, and these places might not be in the current 'Map'. Rely on textual context then.
'Trajectory' represents the ID info of the places you have explored. You start navigating from Place 0.
'Map' refers to the connectivity between the places you have explored and other places you have observed. This map is dynamically updated; some previously seen places/connections might be optimized (pruned) for clarity.
'Supplementary Info' records some places and their corresponding images you have ever seen but have not yet visited. These places are only considered when there is a navigation error, and you decide to backtrack for further exploration.
'Previous Planning' records previous long-term multi-step planning info that you can refer to now.
'Possible Destination Info' describes a possible spatial layout of objects for the target destination. This is intended as a reference and may not be completely accurate.
'Action options' are some actions that you can take at this step.

Output Requirements

For each provided image of the places, you should combine the 'Instruction' and carefully examine the relevant information, such as scene descriptions, landmarks, and objects. You need to align 'Instruction' with 'History' (including corresponding images) to estimate your instruction execution progress and refer to 'Map' for path planning. Check the Place IDs in the 'History' and 'Trajectory', avoiding repeated exploration that leads to getting stuck in a loop, unless it is necessary to backtrack to a specific place. If you can already see the destination, estimate the distance between you and it. If the distance is far, continue moving and try to stop within 1 meter of the destination.
Your answer should be JSON format and must include three fields: 'Thought', 'New Planning', and 'Action'. **You need to combine 'Instruction', 'Trajectory', 'Map', 'Supplementary Info', your past 'History', 'Previous Planning', 'Possible Destination Info', 'Action Options' and the provided images to think about what to do next and why, and complete your thinking into 'Thought'**. Based on your 'Map', 'Previous Planning' and current 'Thought', you also need to update your new multi-step path planning to 'New Planning'. At the end of your output, you must provide a single capital letter in the 'Action options' that corresponds to the action you have decided to take, and place only the letter into 'Action', such as "Action: A".

Figure 5: Task description prompts for the R2R dataset. The bolded sections in the figure highlight the prompt representations of each HiNav module. For the REVERIE dataset, only the instruction section was modified, while the other sections remained unchanged.

information, and HiView's visual enhancement.

Consider the instruction: "Walk down the stairs all the way and past the Christmas tree. Make a right turn and walk past the blue chair into the room with the white sink." This instruction delineates transitions across three key locations: the stairs, the room with the blue chair, and the room with the white sink. HiSpace provides HiNav with potential destination information, including a detailed linguistic description of the final destination extracted from the instruction and the spatial relationships of objects surrounding it. Meanwhile, HiView enhances the visual scene images by highlighting object pairs inferred by Qwen-SP, marking existing objects in green (as shown in Figures 9, 10, and 11). These enhanced images are then fed into HiNav.

After step 15, HiNav activates HiMem's dynamic pruning mechanism. In each subsequent step, it calculates a "pruning priority score" for trajectory points meeting specific criteria and removes the point with the highest score. As illustrated in the figures, at step 15, the point with the highest pruning priority score is "forgotten" (removed from the map) based on the calculated scores. This process eliminates trajectory points irrelevant to the current decision, thereby increasing the likelihood of progressing toward the final destination. In every subsequent decision step, HiNav continues this dynamic pruning, retaining only those trajectory points likely to lead to the final destination. This iterative process, akin to human pathfinding, enables robust and successful navigation in complex environments.

C.2 Failed Case Study

This case illustrates a failure in instruction-following navigation, with the instruction: "Turn right to exit the room. Turn right when you reach the end of the hallway. Walk toward the couches and stop there by the couches." The instruction outlines clear sequential sub-goals: (1) exit the current room, (2) turn right at the end of the hallway, and (3) walk toward the couches and stop nearby.

The agent successfully completed the first sub-goal in Step 0 (moving from Place 0 to Place 3), correctly turning right to exit the room. However, the failure occurred during the execution of the second sub-goal, which required the agent to "turn right when you reach the end of the hallway." After reaching Place 3 in Step 0, the agent proceeded to "go forward to Place 6" in Step 1. Critically, upon arriving at Place 6, the agent failed to execute the required second "right turn" as specified. Instead, in Step 2 (moving from Place 6 to Place 8),

Query to LLM

Instruction

Turn left and to towards the kitchen, turn left at the kitchen and go until you are right before the yellow couch by the picture on your right.

History

step 0: turn left to Place 2 which is corresponding to Image 2, step 1: turn left to Place 3 which is corresponding to Image 3, step 2: go forward to Place 6 which is corresponding to Image 6, step 3: go forward to Place 8 which is corresponding to Image 8

Trajectory

Place 0 2 3 6 8

Map

Place 0 is connected with Places 1, 2
Place 2 is connected with Places 0, 3, 4, 5
Place 3 is connected with Places 1, 6, 7, 2
Place 6 is connected with Places 1, 8, 9, 7, 3
Place 8 is connected with Places 10, 11, 9, 6

Supplementary Info

Place 1, which is corresponding to Image 1
Place 4, which is corresponding to Image 4
Place 5, which is corresponding to Image 5
Place 7, which is corresponding to Image 7

Previous Planning

1. Go forward to Place 8.
2. Stop before the yellow couch by the picture.

Possible Destination Info (For reference only, may not be completely accurate)

The destination is the living room, located near a couch. The spatial layout of the destination is centered around a couch with a pillow, with a picture nearby and a bench also in the vicinity.

Action Options (step 4)

['A. stop', 'B. go forward to Place 10 which is corresponding to Image 10', 'C. turn right to Place 11 which is corresponding to Image 11', 'D. turn right to Place 9 which is corresponding to Image 9', 'E. turn around to Place 6 which is corresponding to Image 6']

LLM Response

Thought

I am currently at Place 8, which corresponds to Image 8. The instruction is to stop right before the yellow couch by the picture on the right. Image 8 shows a living room with a sofa and a picture, but no yellow couch is visible. **The Possible Destination Info suggests the destination is near a couch with a picture nearby. I should explore further to find the yellow couch and picture.**

New Planning

1. Turn right to Place 11 to check for the yellow couch and picture. 2. If not found, explore Place 9."

Action: "C"

Figure 6: Prompt input to the HiDecision module for a specific example in a single navigation round. The bolded sections highlight the contributions of HiNav's modules, with the Map and Supplementary Info components dynamically updated based on pruning. The LLM Response's thought section demonstrates the influence of Possible Destination Info, showing how the LLM incorporates the provided destination description into its decision-making process.

it performed a "left turn to Place 8," as depicted in Figure 12. Relying on the "Possible Destination Info," which provided clues about the couches (Place 8 contains a sofa chair and side table), the agent incorrectly identified Place 8 as the destination and executed the stop action. Consequently, it failed to complete the full navigation path and all intermediate steps, as shown in Figure 13.

The agent appears to have over-prioritized reaching a location matching the final destination's description, stopping prematurely after mistakenly assuming it had completed all necessary intermediate navigation steps.

D I-O-S Dataset Details

The Instruction-Object-Space (I-O-S) dataset is designed to enhance spatial reasoning in vision-and-language navigation (VLN) by integrating natural language instructions, task-relevant objects, and their spatial configurations. The dataset contains 28,414 samples, with 25,694 allocated for training and 2,720 for testing. These samples are derived from oracle paths in the REVERIE dataset (Qi et al., 2020b) and manually crafted trajectories developed for this study. Each sample consists of a natural language instruction I, a list of relevant objects O, and the spatial arrangement at the destination S. The average instruction length is 23.05 words. Object lists are categorized into direct and inferred objects, with averages of 3.99 and 9.98 objects, respectively. Descriptions of the spatial layout at the destination have an average length of 39.12 words. An example from the I-O-S dataset is provided below:

- **Instruction**: Exit the kitchen area through the doorway slightly to your left. Walk across the dining table area. Turn right and pass the blue chair or sofa near the Christmas tree. Stop there.
- **Direct Objects**: ["chair", "sofa chair", "table", "Christmas tree"]
- **Inferred Objects**: ["mirror", "lamp", "plant", "picture", "painting", "decoration", "fan", "light", "island", "heater", "trash can", "stool", "cabinet", "coffee table"]
- **Destination Spatial Layout**: "The destination is the foyer or entryway, located near a blue chair and a Christmas tree. The spatial layout includes

a blue chair or sofa positioned beside a Christmas tree, with a coffee table and a lamp also present in the area."

To construct the I-O-S dataset, we processed data from the REVERIE dataset and our manually annotated trajectories as follows. For samples derived from REVERIE, we directly used its pre-generated bounding box files, which contains object IDs, names, visible view indices, and bounding boxes in $[x, y, w, h]$ format. For our custom instructions and trajectories, we adopted the REVERIE methodology (Qi et al., 2020b) to generate bounding boxes by (1) using Matterport3D's 3D object annotations (center point, axis directions, radii) to define object vertices, (2) projecting these vertices onto 2D image planes using viewpoint camera poses to form $[x, y, w, h]$ bounding boxes, (3) filtering occluded objects by comparing depth overlaps with closer objects, and (4) including only objects within 3 meters of the viewpoint. These annotations are stored in JSON files matching REVERIE's format for consistency.

After obtaining the objects observed along the expert trajectories (i.e., the bounding box files for each navigation point), we processed them as follows to derive the corresponding object lists and destination spatial layouts. We used the LLM Gemini-2.5-Flash to analyze the instructions, bounding box data from navigation points along the path, and the destination, generating the object list for each sample and the spatial layout of objects at the destination. These outputs were then manually inspected and refined.

Objective

•Add two new fields to each JSON entry: direct_obj and potential_obj, based on the instruction field.
  •direct_obj: A list of all objects explicitly mentioned in instruction (e.g., ["sink", "table"] in "clean the sink and table").
  •potential_obj: A list of objects reasonably inferred based on the task or room type, describing the environment or related to the task.
•Retain original fields (instruction`) unchanged in the output.

Processing Steps

Identify Direct Object (direct_obj)

•Extract all objects explicitly named in instruction as targets of the main task or verbs.
  •Focus on nouns syntactically tied to task-related verbs (e.g., "sink" and "table" in "clean the sink and table").
  •Use syntactic parsing (e.g., dependency parsing) to identify direct objects of verbs when possible.
  •Include all objects explicitly mentioned as task targets, even if tied to different verbs (e.g., "clean the sink and organize the table" → ["sink", "table"]).
•Sort direct_obj using:
  1.Frequency (40%): Objects mentioned multiple times rank higher.
  2.Task Relevance (40%): Objects tied to the primary task or verb rank higher (e.g., "sink" in "clean the sink and check the table").
  3.Order of Mention (20%): Earlier-mentioned objects rank higher if frequency and relevance are equal.
•If no objects are mentioned (e.g., "go to the spa") or only pronouns/vague terms are used (e.g., "clean it"), set direct_obj to [].
•Match nouns exactly as in instruction (e.g., "sink", not "basin").
•Do not infer objects for direct_obj; they must be explicitly stated.

Identify Potential Objects (potential_obj)

•Include objects that are:
  •Inferred based on the task or room type, up to a maximum of 3 inferred objects (5 for vague instructions), describing the environment or context (e.g., "bed", "tiles" in "Go to the spa with one bed, brown tiles").
  •Guidelines for inference:
  •Select inferred objects most relevant to the task (e.g., "sponge" for cleaning) or room type (e.g., "towel" in a spa).
  •Avoid speculative inferences (e.g., do not infer "chandelier" in a spa unless mentioned).
  •Use singular nouns for inferred objects unless context suggests plural.
  •If more than 3 (or 5 for vague instructions) inferred objects are possible, prioritize by typicality (e.g., "sponge" over "toaster" for cleaning in a kitchen).
•Sort potential_obj by:
  1.Explicit Mention (40%): Explicitly mentioned objects rank higher.
  2.Frequency (30%): Objects mentioned multiple times rank higher.
  3.Task Relevance (30%): Objects closer to the task or central to the environment rank higher (e.g., "sponge" for cleaning over "lamp").
•If no objects are inferred, set potential_obj to [].

Handle Special Cases

•Vague Instructions:
  •An instruction is vague if it lacks specific object nouns (e.g., "clean the room") or uses generic verbs without clear targets (e.g., "fix something").
  •Set direct_obj to [] and infer up to 5 typical objects for potential_obj based on room type (e.g., ["table", "chair", "lamp"] for a generic room).
•Compound Objects:
  •Include all explicitly mentioned task targets in direct_obj (e.g., "clean the sink and table" → ["sink", "table"]), sorted as above.
•Non-Physical Objects:
  •Exclude abstract entities (e.g., "mess" in "clean the mess") from both direct_obj and potential_obj, setting to [].
  •Allow inferred physical objects relevant to the task (e.g., "sponge" for cleaning).
•Multi-Room Instructions:
  •Infer potential_obj based on the room where the task occurs (e.g., spa for "go from kitchen to spa and clean the sink"). If unclear, use the last-mentioned room.
•Empty/Malformed Instructions:
  •If instruction is empty, null, or malformed (e.g., ""), set direct_obj to [] and potential_obj to [].

Input Validation

•Validate input JSON before processing:
  •If instruction is missing, set direct_obj to [] and potential_obj to [].
  •If other required fields (path, heading, scan, path_id, instr_id) are missing, retain them as null or their default type (e.g., empty list for path) and proceed.

Output Format

•Generate a JSON dictionary with fields in order: instruction, direct_obj, potential_obj.
•Ensure:
  •direct_obj is a list of strings, sorted by frequency, task relevance, order of mention, and alphabetical tiebreaker.
  •potential_obj is a list of strings, sorted by explicit mention, frequency, task relevance, and alphabetical tiebreaker.
  •JSON is well-formed, with 2-space indentation, no trailing commas, and consistent double quotes.
  •Original fields (instruction) are unchanged.
•Validate output:
  •All required fields are present in the specified order.
  •direct_obj and potential_obj are lists of strings.
  •JSON syntax is correct with no trailing commas or missing brackets.

Notes

•Exact Matching for direct_obj: Match nouns exactly as in instruction.
•Inference for potential_obj: Limited to 5 inferred objects to ensure relevance.
•Language Consistency: Use singular/plural as in instruction for mentioned objects; inferred objects use singular unless context suggests plural.

Examples

Example 1
Input:
{"instruction": "Go to the spa with one bed, brown tiles on the walls, a visible white radiator, and clean out the sink and bed"}
Output:
{"instruction": "Go to the spa with one bed, brown tiles on the walls, a visible white radiator, and clean out the sink and bed",
"direct_obj": ["sink", "bed", "tiles", "radiator",],
"potential_obj": ["towel", "tub"]}

Task

•Process the JSON input provided by me and return a complete JSON output adhering to the above requirements.
•Ensure the output is correctly formatted, readable, and both direct_obj and potential_obj are sorted by specified criteria.
•Wait for the my JSON input to process. Do not process sample inputs unless explicitly provided.

Figure 7: Prompt designed for object list extraction in spatial inference experiments, applied to non-fine-tuned LLMs (GPT-4o, Grok3, Gemini-2.5-Flash, Qwen3-4B). This structured prompt directs the LLM to accurately identify and enumerate task-relevant objects from navigation instructions. The same prompt is used for GPT-4o in REVERIE dataset experiments.

You are tasked with processing a JSON input containing navigation instructions and generating a new JSON output. The input JSON has the following structure:

```
{
"path": [string, ...],
"objId": number,
"heading": number,
"scan": string,
"path_id": string,
"instr_id": string,
"instruction": string
}
```

Your goal is to create a valid JSON output that meets these requirements:

1.Retained Fields:
- •Copy path, heading, scan, path_id, instr_id, and instruction from the input, unchanged.
- •Exclude objId.

2.Final Destination Spatial Relations:
- •Include exactly two strings:
  - •First: Starts with "The destination is ..." and describes the final location based on the instruction (e.g., "the laundry room on the first level"). If the destination is unclear, use a generic description (e.g., "the specified location").
  - •Second: Starts with "The spatial layout of the destination is ..." and infers a simple, typical layout for the destination type (e.g., for a laundry room, a shelf above a washing machine). Base inferences on common knowledge, avoiding overly specific assumptions.

3.Output Constraints:
- •Include only the specified fields (path, heading, scan, path_id, instr_id, instruction, final_destination_spatial_relations).
- •Ensure the output is a valid JSON object.
- •Assume the input JSON is valid.

Example Input:

```
{
"path": [
"faa7088781e647d09df1d5b470609aa3",
"7d708a5b80ee45979870bae83b2bdd44",
"c29e99f090194613b5b11af906c47dab",
"aa4cfd0126dd4c6a9c533ca9cb4a033d",
"5bc41a6e3b7748149e0e8592c5b4d142",
"3f432ddd169d4433979e004d1237d029",
"3e51eeaac8404b31ad8a950bb2bb953d"
],
"objId": 156,
"heading": 0.32,
"scan": "cV4RVeZvu5T",
"path_id": "7172_156",
"instr_id": "7172_156_0",
"instruction": "Go to the laundry room on the first level and remove the leopard trinket from the shelf"
}
```

Example Output:

```
{
"path": [
"faa7088781e647d09df1d5b470609aa3",
"7d708a5b80ee45979870bae83b2bdd44",
"c29e99f090194613b5b11af906c47dab",
"aa4cfd0126dd4c6a9c533ca9cb4a033d",
"5bc41a6e3b7748149e0e8592c5b4d142",
"3f432ddd169d4433979e004d1237d029",
"3e51eeaac8404b31ad8a950bb2bb953d"
],
"heading": 0.32,
"scan": "cV4RVeZvu5T",
"path_id": "7172_156",
"instr_id": "7172_156_0",
"instruction": "Go to the laundry room on the first level and remove the leopard trinket from the shelf",
"final_destination_spatial_relations": [
"The destination is the laundry room on the first level.",
"The spatial layout of the destination is a shelf above a washing machine with the leopard trinket on it."
]
}
```

Task:
- •Wait for the my JSON input to process.
- •Generate a complete JSON output adhering to the requirements.
- •Ensure the final_destination_spatial_relations field reflects the instruction's destination and a simple, typical layout

Figure 8: Prompt employed for destination spatial layout inference using GPT-4o in REVERIE experiments. This prompt directs the LLM to generate accurate spatial arrangements of objects at the target location.

Figure 9: Step 15 in a Successful Case Study of HiNav. HiSpace, based on the instructions, accurately and precisely describes information related to the final destination and the relationships of surrounding objects. By highlighting scene objects (e.g., pictures, lamps), HiView guides HiNav, increasing the likelihood of reaching the desired destination. The field 'Pruning Scores Detail' illustrates the complete calculation process and the resulting pruned nodes from HiMem's dynamic pruning. As the current navigation point is already located in a downstairs room, HiNav removes the upstairs navigation point place 5.

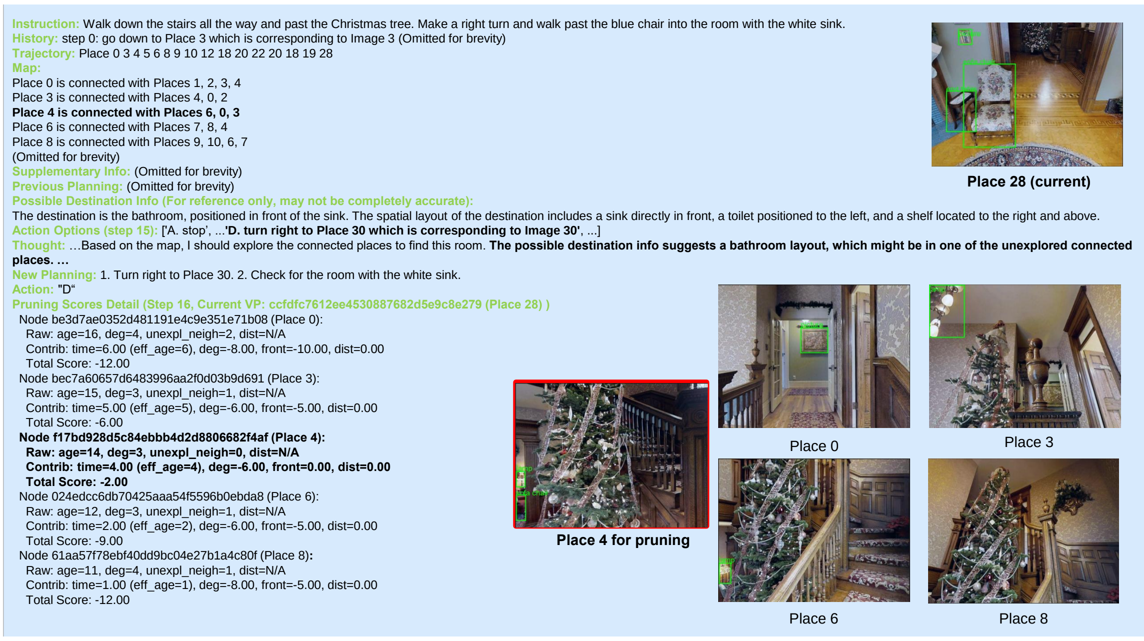


Figure 10: Step 16 follows by Step 15 in this successful case study. The navigation point removed during the pruning in step 15 (Place 5) has already been removed from the set of active nodes considered in this step's memory/map. HiMem now initiates a new round of dynamic pruning. From the five navigation points currently under consideration, Place 4 is identified and subsequently removed.

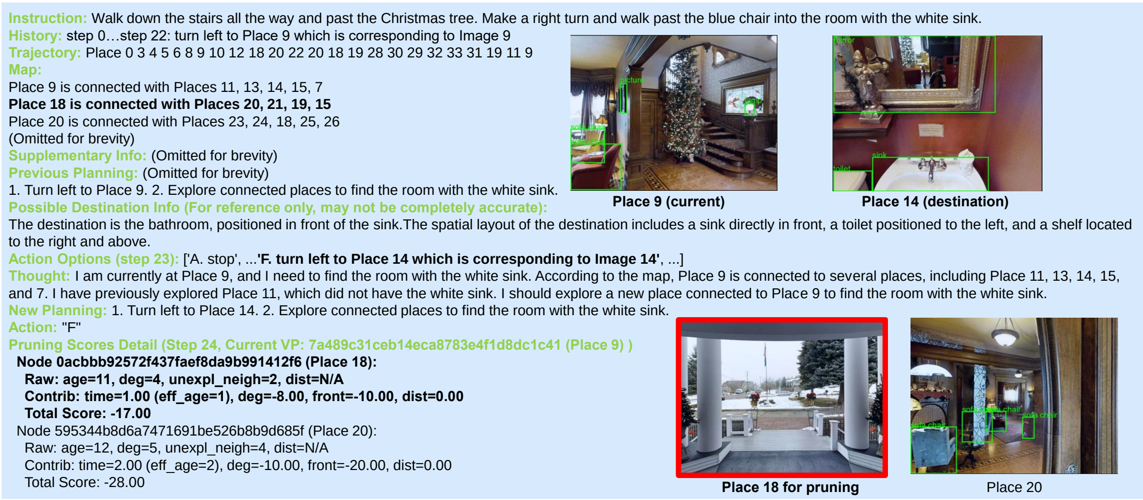


Figure 11: The final step in the successful case study of HiNav. HiMem continued its dynamic pruning process after step 15 to the final step. Leveraging the precise destination information and object visual enhancements inferred by HiSpace and HiView respectively, HiNav successfully selected the crucial Action in this step: F. turn left to Place 14 which is corresponding to Image 14. Ultimately, the agent successfully reached the intended destination after a total of 25 steps.

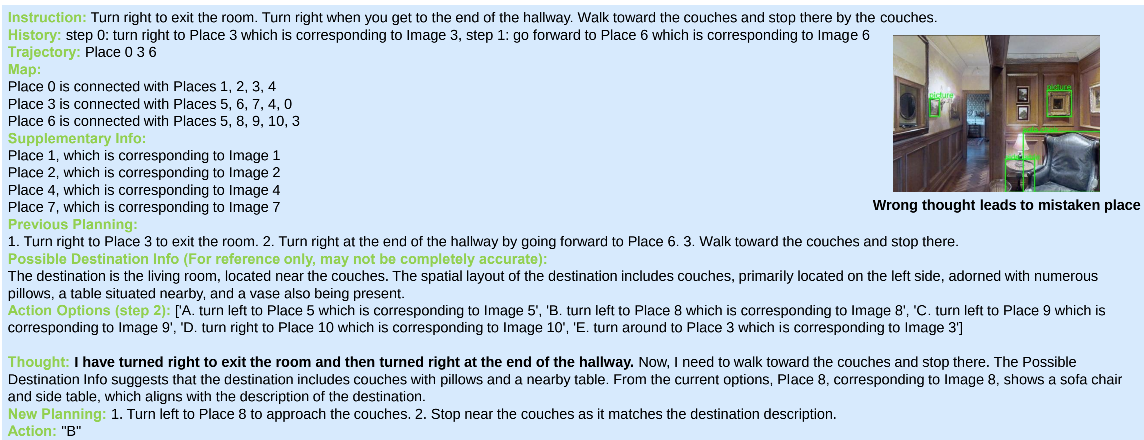
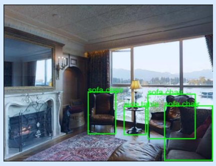
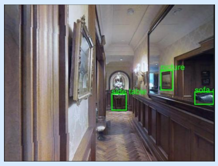


Figure 12: The key step 2 of a failure case for HiNav. Following the instruction to navigate to the room with couches, after executing the initial steps, HiNav, at step 2, selected the action to proceed to Place 8 based on the possible destination information (describing the living room and couches). However, this choice did not successfully guide the agent to ultimately reach the specific couch destination specified in the instruction.

Instruction: Turn right to exit the room. Turn right when you get to the end of the hallway. Walk toward the couches and stop there by the couches.
History: step 0: turn right to Place 3 which is corresponding to Image 3, step 1: go forward to Place 6 which is corresponding to Image 6, step 2: turn left to Place 8 which is corresponding to Image 8
Trajectory: Place 0 3 6 8
Map:
Place 0 is connected with Places 1, 2, 3, 4
Place 3 is connected with Places 5, 6, 7, 4, 0
Place 6 is connected with Places 5, 8, 9, 10, 3
Place 8 is connected with Places 11, 12, 13, 6, 5
Supplementary Info:
Place 1, which is corresponding to Image 1
Place 2, which is corresponding to Image 2
Place 4, which is corresponding to Image 4
Place 7, which is corresponding to Image 7
Place 9, which is corresponding to Image 9
Place 10, which is corresponding to Image 10



Intended Destination



Actual Destination (Place 8)

Previous Planning:
1. Turn left to Place 8 to approach the couches. 2. Stop near the couches as it matches the destination description.
Possible Destination Info (For reference only, may not be completely accurate):
The destination is the living room, located near the couches. The spatial layout of the destination includes couches, primarily located on the left side, adorned with numerous pillows, a table situated nearby, and a vase also being present.
Action Options (step 3): [**'A. stop'**, 'B. turn left to Place 11 which is corresponding to Image 11', 'C. go forward to Place 12 which is corresponding to Image 12', 'D. turn right to Place 13 which is corresponding to Image 13', 'E. turn around to Place 6 which is corresponding to Image 6', 'F. turn around to Place 5 which is corresponding to Image 5']

Thought: **I have reached Place 8**, which matches the description of the destination with couches and a side table. The instruction was to stop by the couches, and the previous planning also indicated that Place 8 is the destination. Therefore, I should stop here.
New Planning: Stop at Place 8 as it matches the destination description with couches and a side table.
Action: "A"

Figure 13: The final step of the failure case for HiNav. Following the instruction to navigate to the room with couches, after executing the first two steps to reach Place 8, HiNav, at step 3, mistakenly determined that the current location (Place 8) was the final destination based on its internal planning and possible destination information, and chose to stop. As the agent failed to reach the actual couch destination specified in the instruction, this navigation attempt ultimately failed.