Think before Go: Hierarchical Reasoning for Image-goal Navigation

Pengna Li[1,3,*] **Kangyi Wu**[1,*] **Shaoqing Xu**[2,3,†] **Fang Li**[2,3]
Lin Zhao[4] **Long Chen**[3] **Zhixin Yang**[2] **Nanning Zhen**[1,✉]

[1]National Key Laboratory of Human-Machine Hybrid Augmented Intelligence,
Nation Engineering Research Center for Visual Information and Applications,
Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University
[2]University of Macau, [3] Xiaomi EV
[4]School of Automation, Beijing Institute of Technology

Abstract

Image-goal navigation steers an agent to a target location specified by an image in unseen environments. Existing methods primarily handle this task by learning an end-to-end navigation policy, which compares the similarities of target and observation images and directly predicts the actions. However, when the target is distant or lies in another room, such methods fail to extract informative visual cues, leading the agent to wander around. Motivated by the human cognitive principle that deliberate, high-level reasoning guides fast, reactive execution in complex tasks, we propose Hierarchical Reasoning Navigation (HRNav), a framework that decomposes image-goal navigation into high-level planning and low-level execution. In high-level planning, a vision-language model is trained on a self-collected dataset to generate a short-horizon plan, such as whether the agent should walk through the door or down the hallway. This downgrades the difficulty of the long-horizon task, making it more amenable to the execution part. In low-level execution, an online reinforcement learning policy is utilized to decide actions conditioned on the short-horizon plan. We also devise a novel Wandering Suppression Penalty (WSP) to further reduce the wandering problem. Together, these components form a hierarchical framework for Image-goal Navigation. Extensive experiments in both simulation and real-world environments demonstrate the superiority of our method.

1 Introduction

Image-goal Navigation (Zhu et al., 2017) has emerged as a fundamental problem in embodied navigation (Zheng et al., 2024; Gan et al., 2020), which aims to enable an agent to autonomously move within an unseen environment to reach a target specified by an image. Motivated by its broad potential in last-mile delivery and household robots (Wasserman et al., 2023; Majumdar et al., 2022; Krantz et al., 2023b), this task has received growing attention from the research community.

Figure 1: **Our proposed HRNav.** We adopt a two-level hierarchy: given goal image and current observations, a *slow* planner predicts short-horizon plans, while a *fast* executor follows these short-horizon plans to produce low-level actions, enabling more efficient navigation.

Despite the rapid growth, image-goal navigation remains highly challenging due to two factors. First, the agent operates under severe partial observability: it receives no step-by-step instructions and has no access to an environment map, thus relying solely on a single egocentric RGB sensor to infer where to go. Second, real-world indoor scenes exhibit complex spatial structures, and long-horizon episodes often induce large discrepancies between the current observation and the goal image (e.g., in different rooms with little visual overlap). Together, these challenges leave the agent with little meaningful clues, causing it to drift into inefficient behaviors such as backtracking or aimless wandering. Therefore, the key to improving navigation is to mine informative cues from the limited inputs to support reliable decision-making.

To realize that, prior works have largely followed two lines of research. **On the one hand**, modular-based methods (Krantz et al., 2023a; Lei et al., 2024) explicitly decompose the problem into several isolated tasks, and introduce additional sensors

* Co-first Authors.
{sauerfisch, wukangyi747600}@stu.xjtu.edu.cn
† Project Leader.
✉ Corresponding Author.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 29336–29357

(e.g., depth and pose) to compensate for the lack of information. While effective, such pipelines typically incur extra hardware and system complexity, and their performance can be brittle due to error accumulation across modules. **On the other hand**, the end-to-end reinforcement learning (RL) paradigm (Sun et al., 2024; Li et al., 2025b) learns a navigation policy to directly map the visual representations to the action control. Although RL has proven effective across multiple navigation tasks (Huang et al., 2025; Qi et al., 2025; Li et al., 2025b), training an end-to-end policy purely with RL from scratch often fails to acquire strong spatial understanding and high-level planning capability, especially in unseen environments.

Can we realize robust informative cue extraction without introducing additional sensors? One useful perspective comes from the fast-slow view of human cognition, which characterizes behavior as the interplay between a fast system 1 for reactive execution and a slow system 2 for deliberate reasoning (Kahneman, 2011). When the goal is distant and visual evidence is weak, humans often invoke the slow system to integrate sparse observations into higher-level spatial hypotheses and to sketch a coarse plan (e.g., exit the room), which in turn constrains subsequent action execution by the fast system. This division of labor effectively mines informative cues from limited inputs by converting ambiguous, partial observations into structured, short-horizon objectives, thereby reducing the search space and mitigating aimless wandering during long-horizon navigation.

Inspired by this observation, we aim to equip image-goal navigation agent with a similar high-level planning (slow) and low-level execution (fast) hierarchical mechanism. For the fast system, we train a lightweight navigation policy to directly output low-level actions for reactive control. For the slow system, considering the strength in multi-modal understanding and reasoning, Vision-Language Models (VLMs) are well-suited for this role. However, according to our experiments, simply zero-shotting the VLM yields limited gains. We attribute this to the fact that navigation videos primarily involve viewpoint changes and geometric transitions, which are less aligned with the event-centric understanding that VLMs are typically trained for. As a result, fully unlocking the potential of VLM-based slow planning requires task-specific finetuning. Yet, such finetuning relies on large-scale annotated short-horizon planning data, which is absent in standard image-goal navigation datasets. To bridge this gap, we collect a new Hierarchical Reasoning dataset with annotated short-horizon goals for 767k trajectories. This helps the VLM-based slow system to have substantially improved planning capability.

In this paper, we propose a Hierarchical Reasoning Navigation framework (HRNav) to translate image-goal navigation to the interplay between high-level reasoning and low-level execution. Specifically, we adopt a two-stage training scheme: 1) Train the high-level reasoning with our self-collected planning dataset. 2) Freeze the slow system, incorporate its short-horizon planning ability to learn an efficient navigation policy with a novel Wandering Suppression Penalty (WSP) to further reduce the wandering problem.

We conclude our main contributions below:

- We explore the aimless wandering problem in image-goal navigation and propose HRNav to combine VLM-based short-horizon reasoning with policy-level execution to map visual observations into actionable navigation intent.
- We collect a large planning dataset with 767k trajectories to train a VLM-based slow system with strong spatial understanding and reasoning ability. We also devise a novel penalty to further reduce the wandering problem.
- HRNav outperforms all the other state-of-the-art (SOTA) methods in both simulated and real-world environments, offering a new pipeline in image-goal navigation and paving the way for future research in the field.

2 Related Work

2.1 Image-goal Navigation

The image-goal navigation task requires the agent to perceive, reason, and execute actions in an unseen environment. Early modular-based methods (Krantz et al., 2023a; Lei et al., 2024) tackle this task step by step, which decomposes navigation into different perception, mapping, and path planning modules. To localize the agent, they usually utilize the mapping module to construct a map (Yin et al., 2025; Guo et al., 2025) or graph (Kim et al., 2023; Jiao et al., 2025) using additional sensors such as GPS, depth or pose information, which limits their applicability in real-world robotic settings. An alternative line

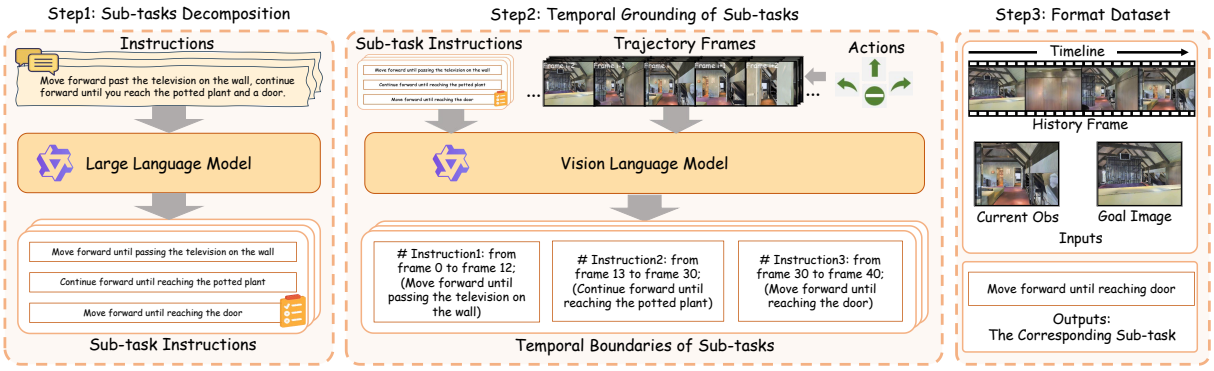

Figure 2: Overview of the hierarchical reasoning dataset construction pipeline.

of work adopts end-to-end reinforcement learning paradigms, learning a navigation policy directly to map the goal image and observations to the low-level corresponding actions (Al-Halah et al., 2022; Sun et al., 2024; Li et al., 2025b). While these methods demonstrate promising results, their training paradigms typically rely on training scenes and thus exhibit limited generalization to unseen scenes. Moreover, directly mapping the visual signal to low-level actions can be brittle in complex scenes, especially when the agent is far from the goal or located in a different room, often resulting in wandering behaviors. Recent work REGNav (Li et al., 2025b) introduces room-level relationships into the navigation process to alleviate this issue. However, it lacks executable sub-task planning, and the navigation policy is still driven by visual matching rather than structured reasoning.

2.2 Navigation with MLLM

Recent advancements in multi-modal large language model (MLLM) (Zhang et al., 2024; Liu et al., 2024a; Yang et al., 2025) have catalyzed progress across a wide range of research domains (Guo et al., 2024; Yuan et al., 2025; Fu et al., 2024; Qi et al., 2026; Liu et al., 2025, 2024b; Wu et al., 2025b,c). Their strong multi-modal understanding and reasoning capabilities provide new opportunities for embodied navigation. One line of work leverages powerful closed-source models for zero-shot navigation within modular frameworks. These methods (Yin et al., 2024; Long et al., 2024; Lyu et al., 2026) typically employ VLMs to perceive the environment and LLMs as planners to generate step-by-step decisions. For example, InstructNav (Long et al., 2024) utilizes GPT-4V (Yang et al., 2023) to construct intuition value map and GPT-4 (Achiam et al., 2023) for dynamic chain-of-navigation reasoning. SG-Nav (Yin et al., 2024) and Unigoal (Yin et al., 2025) employ LLaVA (Liu et al., 2023) to construct an online 3D scene graph to prompt LLMs. While these methods achieve strong reasoning capability, their reliance on closed-source LLMs incurs high inference cost and limits practical deployment. Another line of work focuses on finetuning the open-source models on expert trajectories collected from simulator datasets. They usually adopt Video-based LLMs (Lin et al., 2024; Li et al., 2024b; Zhang et al., 2024) to capture visual information and predict low-level action in an end-to-end manner. StreamVLN (Wei et al., 2025) introduces an efficient framework that supports streaming visual inputs with bounded memory for action generation. CompassNav (Li et al., 2025a) further explores a two-stage training paradigm by combining supervised finetuning with reinforcement learning to improve navigation performance. However, it remains challenging to finetune a VLM for image-goal navigation. Fully supervised learning will largely limit the exploration capability of the model, which is critical for the task. Instead of finetuning the VLM to predict actions, we teach it to make short-horizon plans. In this way, we leverage the VLM's understanding and reasoning for high-level planning, while retaining the action policy's capacity for efficient exploration.

3 Method

3.1 Task Setup

In a navigation episode, the agent is initially placed at a random starting position p_0 in an unseen environment and is provided only with the goal image I_g. At each time step t, the agent perceives the environment through an egocentric RGB observation I_o^t and selects a low-level action a_t from the action space (e.g., move forward, turn left, turn right, and stop) and interacts with the environment,

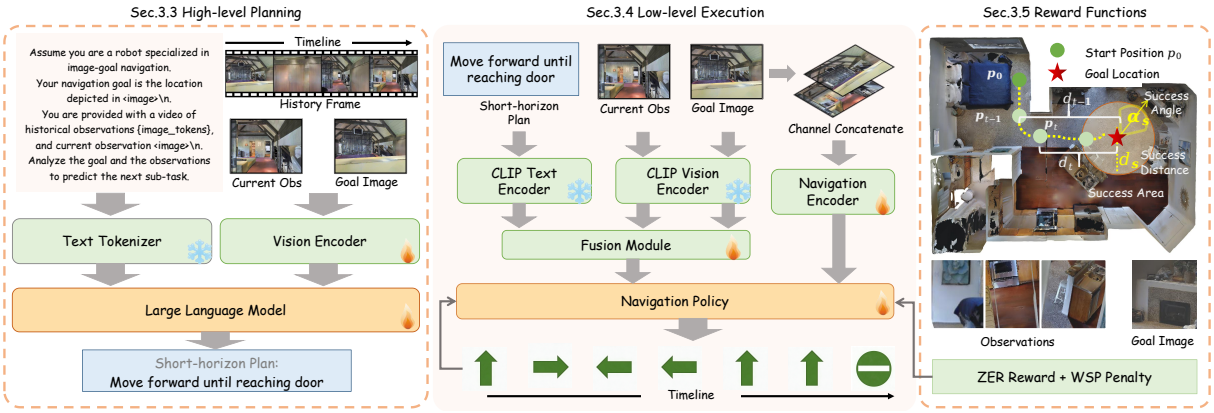

Figure 3: An overview of HRNav framework, where a high-level vision-language model predicts a short-horizon objective and a low-level policy executes actions via reinforcement learning under the reward functions.

resulting in a new state and observation. The navigation episode terminates when the agent issues the stop action or reaches the maximum time steps.

To tackle this task, we formulate image-goal navigation as a high-level planning and low-level execution problem. Specifically, a high-level planning module infers coarse-grained navigation intents based on the goal image and visual observations, while a low-level policy executes fine-grained actions conditioned on the inferred intents through reinforcement learning.

3.2 Hierarchical Reasoning Dataset

As illustrated in Fig. 2, we construct a hierarchical reasoning dataset from existing VLN trajectory data (e.g., R2R-CE (Krantz et al., 2020) and RxR-CE (Ku et al., 2020)) to support sub-task planning based on visual observations. In these datasets, each trajectory consists of a natural language instruction that provides detailed instructions and a first-person video sequence that records the agent's observation and execution in the environment.

Sub-task Decomposition. Given a full navigation instruction, we employ a large language model (Qwen3-14B (Yang et al., 2025)) to decompose it into a temporally ordered sequence of non-overlapping, atomic sub-task instructions.

Temporal Grounding of Sub-tasks. To align the decomposed sub-tasks with visual observations, we perform temporal grounding on the trajectory video. Specifically, given the trajectory video and the ordered sub-task instructions, we prompt a vision-language model (Qwen2.5-VL-32B (Bai et al., 2025)) to identify the temporal boundaries of each sub-task. We augment each video frame with overlaid textual metadata, including the timestamp and the action executed at that time step. The output is formatted as explicit frame intervals for each sub-task, which associate every video frame with its corresponding navigation sub-task.

Dataset Formatting. For each navigation trajectory, the last frame is selected as the goal image I_g. Given a current observation at time step t, we assign its supervision label as the sub-task that becomes active within a short future temporal window, encouraging the model to anticipate upcoming navigation intent. Each sample is formatted as *(history observations, current observation, goal image)* $\rightarrow$ *next sub-task instruction*. Running this pipeline produces 198K training samples from RxR (Krantz et al., 2020), 239K from xsR2R (Ku et al., 2020), and 330K from Youtube Videos (Lin et al., 2023; Cheng et al., 2024), which serve as the training corpus for high-level planning. We further apply a Triple Quality Control Mechanism (TQCM) to filter malformed, temporally inconsistent, and semantically misaligned annotations. More details are provided in Appendix C.4.

3.3 High-level Planning

The high-level planning module aims to infer the short-horizon plan based on visual observations and the goal image. We adopt VILA (Lin et al., 2024) as the backbone, which consists of three main components: a vision encoder, a projector, and a large language model. Given a sequence of historical observations, the current observation, and the goal image, the vision encoder converts the input images into visual tokens, which are downsampled and mapped into the language domain through an MLP projector. The projected visual tokens are concatenated with textual prompt tokens and fed into the LLM, which performs auto-regressive gen-

Method	Easy		Medium		Hard		Overall	
	SR↑	SPL↑	SR↑	SPL↑	SR↑	SPL↑	SR↑	SPL↑
VGM (Kwon et al., 2021)	86.1%	79.6%	81.2%	68.2%	60.9%	45.6%	76.1%	64.5%
Mem-Aug (Mezghan et al., 2022)	78.0%	63.0%	70.0%	57.0%	60.0%	48.0%	69.3%	56.0%
TSGM (Kim et al., 2023)	91.1%	**83.5%**	82.0%	68.1%	70.3%	50.0%	81.1%	67.2%
FGPrompt-EF (Sun et al., 2024)	97.1%	70.7%	94.7%	67.6%	82.3%	56.7%	90.4%	66.5%
RFSG (Feng et al., 2025)	-	-	-	-	-	-	91.0%	67.8%
NavigateDiff (Qin et al., 2025)	-	-	-	-	-	-	91.0%	64.8%
REGNav (Li et al., 2025b)	97.5%	71.4%	95.4%	69.4%	87.1%	59.4%	92.9%	67.1%
HRNav (This work)	**98.5%**	75.2%	**96.2%**	**73.7%**	**87.2%**	**66.8%**	**94.0%**	**71.2%**

Table 1: Comparison with state-of-the-art methods across three different difficulty levels on Gibson. The **best** and second performance results are highlighted.

eration to predict the current short-horizon plan.

The high-level module is trained via supervised finetuning (SFT). Besides our hierarchical reasoning dataset, we incorporate several auxiliary datasets. Following NaVILA (Cheng et al., 2024), we include trajectory summarization data constructed from navigation videos (EnvDrop (Tan et al., 2019)), real-world 3D-grounded question answering data (ScanQA (Azuma et al., 2022)), and general VQA datasets (ShareGPT-4V (Chen et al., 2024), Video-chatgpt (Maaz et al., 2024)) to enhance the model's scene understanding and multimodal reasoning ability.

3.4 Low-level Execution

Conditioned on the short-horizon objective generated by the high-level planning module, the low-level execution module produces executable navigation actions. Inspired by PSL (Sun et al., 2025), we construct two complementary representations to capture both semantic intent and navigation-specific visual cues. First, the short-horizon plan, current observation, and goal image are encoded using CLIP (Radford et al., 2021) text and vision encoders, respectively. The resulting features are fused through a multimodal fusion module to obtain a semantic representation. Second, to preserve spatial information for navigation, the current observation and the goal image are concatenated along the channel dimension and processed by a navigation encoder that focuses on geometry and layout cues. Then, we concatenate the semantic and navigation features as the fused features.

The fused features f_{fused} and previous actions a_{t-1} are jointly fed into a navigation policy network π to predict the agent's current state embedding s_t at time t, which is defined as:

$$s_t = \pi(f_{fused} \oplus a_{t-1} \mid h_{t-1}), \quad (1)$$

where h_{t-1} is the hidden layer of the recurrent layers in the policy from the previous step. An actor-critic network then utilizes s_t to predict the state value and determine the agent's next action. The low-level policy is optimized using reinforcement learning, allowing it to maintain exploration capability and adapt to unseen environments. More policy details can be found in Appendix Sec B.3.

3.5 Reward Functions

ZER Reward. Following ZER (Al-Halah et al., 2022), we use a dense distance-and-view shaping reward with a sparse success reward:

$$r_t = (d_{t-1} - d_t) + \mathbb{I}(d_t \le d_s)(\alpha_{t-1} - \alpha_t) - \gamma, \quad (2)$$

$$R_s = 5\Big[\mathbb{I}(d_t \le d_s) + \mathbb{I}(d_t \le d_s \wedge \alpha_t \le \alpha_s)\Big], \quad (3)$$

where d_t is the geodesic distance, α_t is the view-angle difference, and the view shaping is activated only within the success area ($d_t \le d_s$). We set d_s=1m and α_s=25°. However, the shaping term is purely local and does not explicitly penalize redundant motions, making the agent prone to short-range oscillations and backtracking.

Wandering Suppression Penalty. Built upon the above reward, we further introduce a *Wandering Suppression Penalty* (WSP) that explicitly discourages unnecessary detours and short-term revisits. Concretely, WSP consists of a path-length penalty and a revisit penalty:

$$r_t^{\text{wsp}} = -(\ell_t - \ell_{t-1}) - \Delta c_t, \quad (4)$$

where ℓ_t is the cumulative traveled path length, $(\ell_t - \ell_{t-1})$ is the step displacement, c_t is a monotonic oscillation counter that increases when the agent exhibits short-term backtracking. The overall step reward is:

$$\tilde{r}_t = r_t + R_s + \lambda_w r_t^{\text{wsp}}, \quad (5)$$

Method	MP3D	
	SR↑	SPL↑
Mem-Aug	6.9%	3.9%
ZER	14.6%	10.8%
FGPrompt-EF	75.7%	48.8%
REGNav	78.0%	50.2%
HRNav	**81.4%** (+3.4%)	**56.3%** (+6.1%)

Table 2: Cross-domain evaluation on MP3D.

Method	HM3D	
	SR↑	SPL↑
Mem-Aug	3.5%	1.9%
ZER	9.6%	6.3%
FGPrompt-EF	75.2%	42.1%
RFSG	73.4%	42.7%
REGNav	75.2%	44.0%
HRNav	**80.0%** (+4.8%)	**49.3%** (+5.4%)

Table 3: Cross-domain evaluation on HM3D.

where λ_w is the weight hyperparameter. The ablation on λ_w can be found in Appendix Sec. D.2. More reward details can be found in Appendix Sec. B.5 and Sec. B.6.

4 Experiments

4.1 Simulation Experiments

Datasets and Evaluation Metrics. The high-level planning module is trained via supervised finetuning using the datasets described in Sec. 3.3, including our hierarchical reasoning dataset as well as auxiliary navigation and visual reasoning datasets. For the low-level execution module, all of the experiments are conducted on the Habitat simulator (Savva et al., 2019; Szot et al., 2021). We train it on the Gibson dataset (Xia et al., 2018) using the dataset split provided by (Mezghan et al., 2022). Gibson contains diverse indoor environments and consists of 9000 training episodes from 72 scenes and 4200 testing episodes from 14 scenes. To evaluate cross-domain generalization, we test the trained agent on the Matterport3D (MP3D) (Chang et al., 2017) and Habitat-Matterport3D (HM3D) (Ramakrishnan et al., 2021) datasets.

We adopt standard navigation metrics including Success Rate (SR) and Success weighted by Path Length (SPL) (Anderson et al., 2018). SPL balances the efficiency and success rate by calculating the weighted sum of the ratio of the shortest path length to the predicted path length. The maximum number of steps per episode is set to 500.

Implementation Details We adopt a two-stage training scheme: first train the high-level module with supervised finetuning (SFT) for one epoch on the curated corpus. After SFT, the high-level module is frozen and only used for inference; then train the low-level policy with DD-PPO (Wijmans et al., 2019). We keep the same *fast–slow* schedule: the low-level policy runs frequently to output actions, while the high-level module is invoked sparsely to update sub-task instructions (every 15 steps). In simulation, the agent height is 1.5m with a radius of 0.1m, using a single RGB sensor with a 90° FOV at 128×128 resolution. The action space includes MOVE_FORWARD (0.25m), TURN_LEFT/RIGHT (30°), and STOP. We set the slack reward γ to 0.01. All experiments are conducted on 8 NVIDIA H20 GPUs; high-level finetuning takes 64 hours and low-level training takes 30 hours (20M steps).

Efficiency Analysis. HRNav achieves 41.16 ms average latency per step and 24.29 FPS over 5,000 navigation steps on one NVIDIA H20 GPU with the default 15-step planning interval. Although each slow-planner call takes 374.12 ms, its cost is amortized by sparse invocation, while the low-level executor runs at 14.22 ms per step. Detailed results are given in Appendix D.3.

In-domain Evaluation on Gibson. Table 1 reports the performance comparison on the Gibson dataset. (We report the results averaged over 3 random seeds. The variances are less than 1e-3.) HRNav achieves the best overall results, reaching an SR of 94.0% and an SPL of 71.2%. Compared with all prior methods, HRNav improves the overall SPL by **+4.1%** and the SR by **+1.1%**. Notably, the gains are more pronounced on challenging scenarios: HRNav improves SPL by **+4.3%** on the Medium set and **+7.4%** on the Hard set, demonstrating more efficient long-horizon navigation and reduced wandering behavior.

Cross Domain Evaluation on MP3D and HM3D. Tables 2 and 3 report cross-domain evaluation results on MP3D and HM3D. All methods are trained on the Gibson dataset and directly tested on these datasets without any finetuning, evaluating their generalization ability to unseen environments.

HRNav achieves the best performance on both benchmarks. On MP3D, HRNav reaches an SR

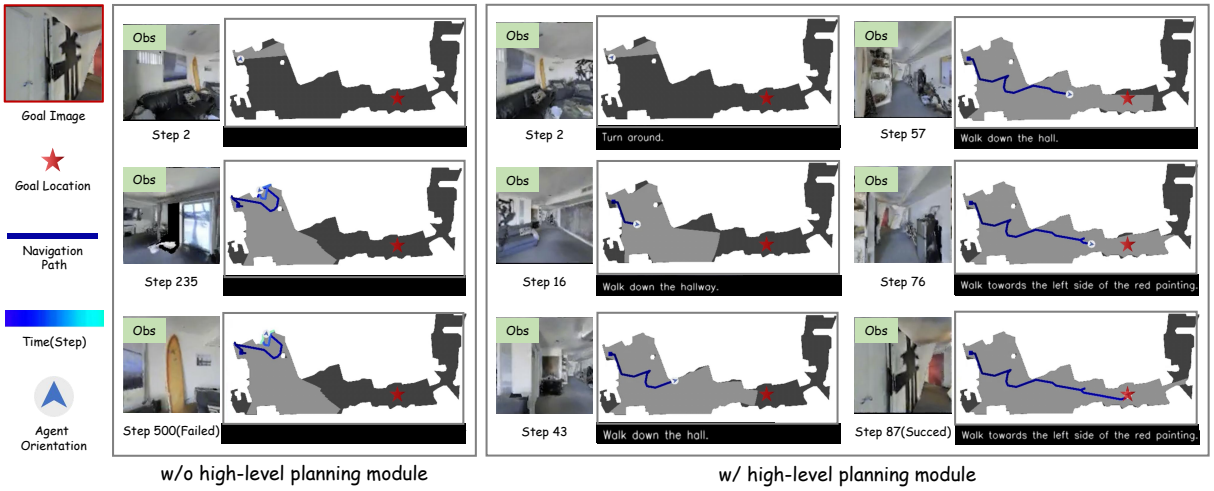

Figure 4: Visualization of navigation trajectories with and without high-level planning. For the model with the high-level planning module (right), each top-view map is annotated with the short-horizon objective, which is predicted online and remains consistent with the subsequent navigation path. High-level planning leads to more efficient and successful navigation.

of 81.4% and an SPL of 56.3%, outperforming the strongest prior method REGNav by **+3.4% SR** and **+6.1% SPL**. On the more challenging HM3D benchmark, HRNav further improves SR to 80.0% and SPL to 49.3%, exceeding REGNav by **+4.8% SR** and **+5.4% SPL**. These consistent gains demonstrate that hierarchical short-horizon planning significantly improves cross-domain robustness, enabling more efficient navigation trajectories and reducing domain-specific overfitting.

Visualization. Fig. 4 presents a representative visualization comparing navigation behaviors with and without the high-level planning module. Without high-level planning, the agent exhibits frequent backtracking and wandering, eventually failing to reach the goal within the maximum step budget. In contrast, with short-horizon objectives, the agent follows a more direct and coherent trajectory, successfully reaching the goal with fewer redundant explorations. This example qualitatively demonstrates how high-level planning provides effective guidance for long-horizon navigation and improves trajectory efficiency. Additional qualitative results are provided in the appendix.

Ablation Studies. Table 4 summarizes the ablation results on model components. Removing the high-level planning module (w/o HL) leads to a clear performance drop in both SR and SPL, indicating that a hierarchical short-horizon objective is essential for effective navigation. Replacing the high-level module with Qwen2 also degrades performance compared to the full model, suggesting that the proposed high-level design provides more suitable objectives for this task. For the low-level policy, removing either semantic features (Semantic-only) or navigation-specific visual features (Nav-only) causes substantial degradation, with the semantic-only variant failing dramatically. This demonstrates that both semantic intent and structural navigation cues are necessary. In addition, excluding previous actions reduces performance, highlighting the importance of action history for stable control.

Method	**SR↑**	**SPL↑**
HRNav	**93.36%**	**66.21%**
Ablation A: High-level planning		
w/o HL	92.52%	63.53%
rp HL with Qwen2	81.62%	55.29%
Ablation B: Low-level policy		
Semantic-only	49.4%	32.6%
Nav-only	81.6%	61.1%
w/o prev actions	87.7%	62.3%

Table 4: Ablation study on different components on Gibson test set. All the experiments are conducted without the new reward.

Table 5 evaluates the impact of the proposed reward components. Without the new reward, the agent achieves an SPL of 66.21%. Introducing either the revisit penalty or the path-length regularization alone improves SPL, while combining both yields the best performance, improving SPL to **71.15%** and SR to **94.00%**. These results show

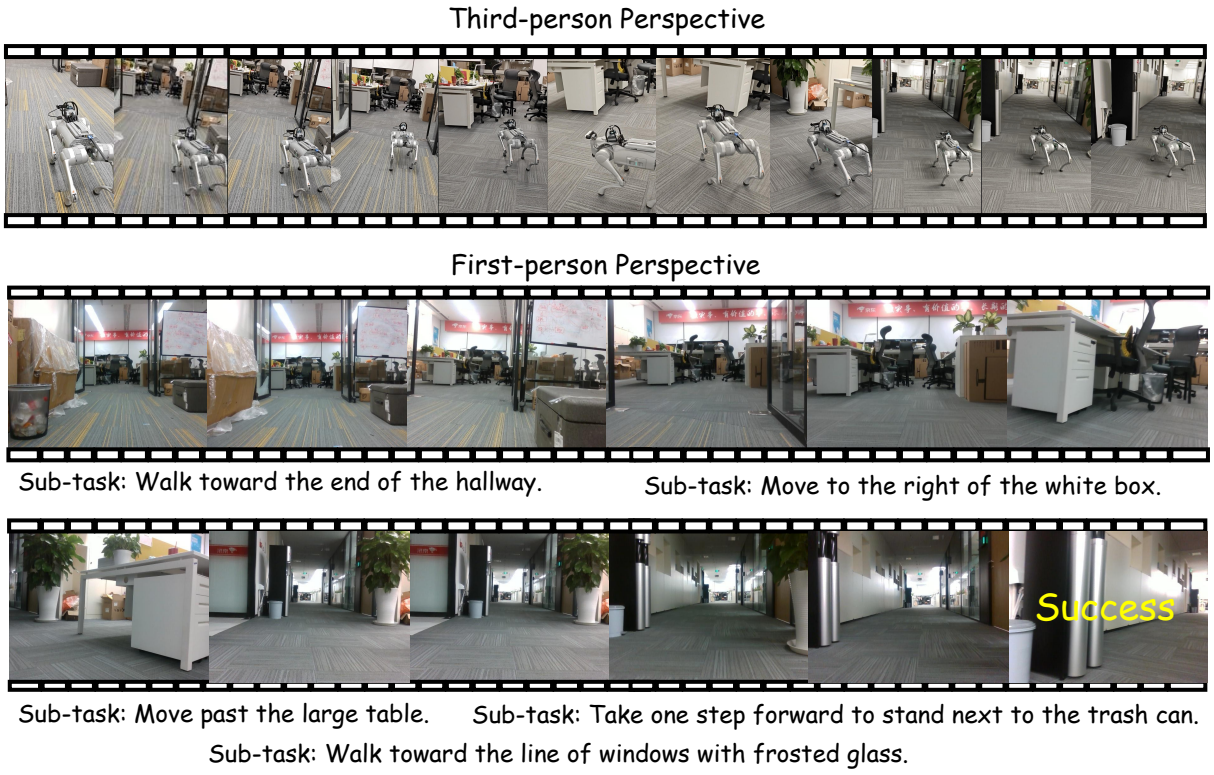

Figure 5: Real-world qualitative visualization of HRNav.

Setting	SR↑	SPL↑
w/o new reward	93.36%	66.21%
only revisit	91.62%	70.62%
only path_len	91.43%	68.13%
revisit+path_len	**94.00%**	**71.15%**

Table 5: Ablation study on the proposed reward design on Gibson test set.

Model	B4↑	M↑	C↑	R↑
Qwen2-VL	1.2	9.1	0.13	22.8
Qwen3-VL	3.9	14.4	0.23	21.8
HRNav-Slow	**31.5**	**38.3**	**2.74**	**45.6**

Table 6: Evaluation of high-level planning on a test subset of the self-collected Hierarchical Reasoning dataset.

that the two reward components are complementary and jointly encourage more efficient and less redundant navigation trajectories.

Effectiveness of High-Level Planning Module. To more intuitively assess whether our slow system can perform high-level planning accurately, we randomly sampled 100 trajectories from our collected dataset for evaluation. We feed the history and observation images into different models in the same manner and use the same prompt to elicit short-horizon planning. We compare with Qwen2-VL (Bai et al., 2025) and Qwen3-VL (Yang et al., 2025), which are two widely used open-source VLMs. Four metrics (Wu et al., 2025a) are calculated: BLEU-4 (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), ROUGE-L (Lin, 2004), and CIDEr (Vedantam et al., 2015). The results are presented in TABLE 6. It's clear that after training, our model can generate short-horizon goals more precisely. This also helps explain why HRNav ultimately achieves stronger navigation performance.

4.2 Real-World Experiments

Settings. Our real-world experiments were conducted on a Unitree Go2 robot equipped with an Intel RealSense D435i camera on the front. During navigation, the system continuously streams RGB observations from the D435i to a remote server powered by NVIDIA H20 GPUs. Given a goal image, HRNav processes the observations to generate the next sub-task and predicts actions. The predicted action is then transmitted back to the Go2, which executes it by calling the robot motion API.

Qualitative results. Fig. 5 visualizes a successful real-world navigation episode executed by our HRNav. The goal image depicts a water dispenser

near the trash can. The robot is initially placed in the room adjacent to the goal. As shown in the figure, the slow system guides the robot to exit the room by walking down the hallway, and then sets different short-horizon goals to move the robot closer to the target location. When the object in the goal image comes into view, the slow system accurately recognizes it and directs the robot to approach the trash can and stop. This demonstrates that by building a hierarchical fast–slow system, HRNav can reduce a difficult task into manageable steps and reach the destination progressively.

5 Conclusion

We propose **HRNav**, a hierarchical framework for image-goal navigation. Inspired by human cognition, we combine a slow planning module with a fast execution policy for efficient control. To train the slow system, a hierarchical reasoning dataset is constructed by decomposing task instructions into temporally grounded sub-tasks. We add a Wandering Suppression Penalty, which greatly reduces the wandering problem. Experiments on Gibson and cross-domain MP3D/HM3D datasets demonstrate promising results, while ablations validate the effectiveness of key components.

Limitations

Although HRNav demonstrates promising perfomance, sim-to-real transfer remains challenging. First, there is a noticeable gap between the camera settings in simulation and those on the physical robots, leading to mismatched viewpoints and visual distributions. This discrepancy can distort the perceived appearance of rooms, landmarks, and obstacles, thereby degrading sub-task reasoning and action selection in real-world deployment. Second, the robot's body embodiment is not fully modeled during simulation training. In practice, a quadrupedal robot may get stuck or experience unintended contacts when moving close to obstacles (e.g., its hind legs may scrape against furniture), which are not captured by the simulator.

In future work, we will try to incorporate more accurate sensor and embodiment modeling (e.g., camera configuration and full-body collision geometry) to narrow the sim-to-real gap and improve robustness on real robots.

Acknowledgement

This work was supported by Fundamental and Interdisciplinary Disciplines Breakthrough Plan of the Ministry of Education of China under Grant JYB2025XDXM504.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Ziad Al-Halah, Santhosh Kumar Ramakrishnan, and Kristen Grauman. 2022. Zero experience required: Plug & play modular transfer learning for semantic visual navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17031–17041.

Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, and 1 others. 2018. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*.

Daichi Azuma, Taiki Miyanishi, Shuhei Kurita, and Motoaki Kawanabe. 2022. Scanqa: 3d question answering for spatial scene understanding. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 19129–19139.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, and 1 others. 2025. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

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. *arXiv preprint arXiv:1709.06158*.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Conghui He, Jiaqi Wang, Feng Zhao, and Dahua Lin. 2024. Sharegpt4v: Improving large multi-modal models with better captions. In *European Conference on Computer Vision*, pages 370–387. Springer.

An-Chieh Cheng, Yandong Ji, Zhaojing Yang, Zaitian Gongye, Xueyan Zou, Jan Kautz, Erdem Bıyık, Hongxu Yin, Sifei Liu, and Xiaolong Wang. 2024. Navila: Legged robot vision-language-action model for navigation. *arXiv preprint arXiv:2412.04453*.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Zhicheng Feng, Xieyuanli Chen, Chenghao Shi, Lun Luo, Zhichao Chen, Yun-Hui Liu, and Huimin Lu. 2025. Image-goal navigation using refined feature guidance and scene graph enhancement. *arXiv preprint arXiv:2503.10986*.

Jingwen Fu, Xiaoyi Zhang, Yuwang Wang, Wenjun Zeng, and Nanning Zheng. 2024. Understanding mobile gui: From pixel-words to screen-sentences. *Neurocomputing*, 601:128200.

Chuang Gan, Yiwei Zhang, Jiajun Wu, Boqing Gong, and Joshua B Tenenbaum. 2020. Look, listen, and act: Towards audio-visual embodied navigation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9701–9707. IEEE.

Wenxuan Guo, Xiuwei Xu, Hang Yin, Ziwei Wang, Jianjiang Feng, Jie Zhou, and Jiwen Lu. 2025. Iglnav: Incremental 3d gaussian localization for image-goal navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6808–6817.

Xuechen Guo, Wenhao Chai, Shi-Yan Li, and Gaoang Wang. 2024. Llava-ultra: Large chinese language and vision assistant for ultrasound. In *Proceedings of the 32nd ACM international conference on multimedia*, pages 8845–8854.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Ting Huang, Dongjian Li, Rui Yang, Zeyu Zhang, Zida Yang, and Hao Tang. 2025. Mobilevla-r1: Reinforcing vision-language-action for mobile robots. *arXiv preprint arXiv:2511.17889*.

Jianhao Jiao, Jinhao He, Changkun Liu, Sebastian Aegidius, Xiangcheng Hu, Tristan Braud, and Dimitrios Kanoulas. 2025. Litevloc: Map-lite visual localization for image goal navigation. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5244–5251. IEEE.

Daniel Kahneman. 2011. Fast and slow thinking. *Allen Lane and Penguin Books, New York*.

Nuri Kim, Obin Kwon, Hwiyeon Yoo, Yunho Choi, Jeongho Park, and Songhwai Oh. 2023. Topological semantic graph memory for image-goal navigation. In *Conference on Robot Learning*, pages 393–402. PMLR.

Diederik P Kingma. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Jacob Krantz, Shurjo Banerjee, Wang Zhu, Jason Corso, Peter Anderson, Stefan Lee, and Jesse Thomason. 2023a. Iterative vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14921–14930.

Jacob Krantz, Theophile Gervet, Karmesh Yadav, Austin Wang, Chris Paxton, Roozbeh Mottaghi, Dhruv Batra, Jitendra Malik, Stefan Lee, and Devendra Singh Chaplot. 2023b. Navigating to objects specified by images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10916–10925.

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.

Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. 2020. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. *arXiv preprint arXiv:2010.07954*.

Obin Kwon, Nuri Kim, Yunho Choi, Hwiyeon Yoo, Jeongho Park, and Songhwai Oh. 2021. Visual graph memory with unsupervised representation for visual navigation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 15890–15899.

Xiaohan Lei, Min Wang, Wengang Zhou, Li Li, and Houqiang Li. 2024. Instance-aware exploration-verification-exploitation for instance imagegoal navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16329–16339.

LinFeng Li, Jian Zhao, Yuan Xie, Xin Tan, and Xuelong Li. 2025a. Compassnav: Steering from path imitation to decision understanding in navigation. *arXiv preprint arXiv:2510.10154*.

Pengna Li, Kangyi Wu, Jingwen Fu, and Sanping Zhou. 2025b. Regnav: Room expert guided image-goal navigation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 4860–4868.

Pengna Li, Kangyi Wu, Wenli Huang, Sanping Zhou, and Jinjun Wang. 2024a. Camera-aware label refinement for unsupervised person re-identification. *arXiv preprint arXiv:2403.16450*.

Yanwei Li, Chengyao Wang, and Jiaya Jia. 2024b. Llama-vid: An image is worth 2 tokens in large language models. In *European Conference on Computer Vision*, pages 323–340. Springer.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Ji Lin, Hongxu Yin, Wei Ping, Pavlo Molchanov, Mohammad Shoeybi, and Song Han. 2024. Vila: On pre-training for visual language models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 26689–26699.

Kunyang Lin, Peihao Chen, Diwei Huang, Thomas H Li, Mingkui Tan, and Chuang Gan. 2023. Learning vision-and-language navigation from youtube videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8317–8326.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, and 1 others. 2024a. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *European conference on computer vision*, pages 38–55. Springer.

Yuhan Liu, Jingwen Fu, Yang Wu, Kangyi Wu, Pengna Li, Jiayi Wu, Sanping Zhou, and Jingmin Xin. 2025. Mind the gap: Aligning vision foundation models to image feature matching. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20313–20323.

Yuhan Liu, Qianxin Huang, Siqi Hui, Jingwen Fu, Sanping Zhou, Kangyi Wu, Pengna Li, and Jinjun Wang. 2024b. Semantic-aware representation learning for homography estimation. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 2506–2514.

Yuxing Long, Wenzhe Cai, Hongcheng Wang, Guanqi Zhan, and Hao Dong. 2024. Instructnav: Zero-shot system for generic instruction navigation in unexplored environment. *arXiv preprint arXiv:2406.04882*.

Kailin Lyu, Kangyi Wu, Pengna Li, Xiuyu Hu, Qingyi Si, Cui Miao, Ning Yang, Zihang Wang, Long Xiao, Lianyu Hu, and 1 others. 2026. Himemvln: Enhancing reliability of open-source zero-shot vision-and-language navigation with hierarchical memory system. *arXiv preprint arXiv:2603.14807*.

Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Khan. 2024. Video-chatgpt: Towards detailed video understanding via large vision and language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12585–12602.

Arjun Majumdar, Gunjan Aggarwal, Bhavika Devnani, Judy Hoffman, and Dhruv Batra. 2022. Zson: Zero-shot object-goal navigation using multimodal goal embeddings. *Advances in Neural Information Processing Systems*, 35:32340–32352.

Lina Mezghan, Sainbayar Sukhbaatar, Thibaut Lavril, Oleksandr Maksymets, Dhruv Batra, Piotr Bojanowski, and Karteek Alahari. 2022. Memory-augmented reinforcement learning for image-goal navigation. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3316–3323. IEEE.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Yukun Qi, Pei Fu, Hang Li, Yuhan Liu, Chao Jiang, Bin Qin, Zhenbo Luo, and Jian Luan. 2026. Patchcue: Enhancing vision-language model reasoning with patch-based visual cues. *arXiv preprint arXiv:2603.05869*.

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

Yiran Qin, Ao Sun, Yuze Hong, Benyou Wang, and Ruimao Zhang. 2025. Navigatediff: Visual predictors are zero-shot navigation assistants. *arXiv preprint arXiv:2502.13894*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, and 1 others. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Santhosh K Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alex Clegg, John Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, and 1 others. 2021. Habitat-matterport 3d dataset (hm3d): 1000 large-scale 3d environments for embodied ai. *arXiv preprint arXiv:2109.08238*.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, and 1 others. 2019. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9339–9347.

Xinyu Sun, Peihao Chen, Jugang Fan, Jian Chen, Thomas Li, and Mingkui Tan. 2024. Fgprompt: fine-grained goal prompting for image-goal navigation. *Advances in Neural Information Processing Systems*, 36.

Xinyu Sun, Lizhao Liu, Hongyan Zhi, Ronghe Qiu, and Junwei Liang. 2025. Prioritized semantic learning for zero-shot instance navigation. In *European Conference on Computer Vision*, pages 161–178. Springer.

Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, and 1 others. 2021. Habitat 2.0: Training home assistants to rearrange their habitat. *Advances in neural information processing systems*, 34:251–266.

Hao Tan, Licheng Yu, and Mohit Bansal. 2019. Learning to navigate unseen environments: Back translation with environmental dropout. *arXiv preprint arXiv:1904.04195*.

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Justin Wasserman, Karmesh Yadav, Girish Chowdhary, Abhinav Gupta, and Unnat Jain. 2023. Last-mile embodied visual navigation. In *Conference on Robot Learning*, pages 666–678. PMLR.

Meng Wei, Chenyang Wan, Xiqian Yu, Tai Wang, Yuqiang Yang, Xiaohan Mao, Chenming Zhu, Wenzhe Cai, Hanqing Wang, Yilun Chen, and 1 others. 2025. Streamvln: Streaming vision-and-language navigation via slowfast context modeling. *arXiv preprint arXiv:2507.05240*.

Erik Wijmans, Abhishek Kadian, Ari Morcos, Stefan Lee, Irfan Essa, Devi Parikh, Manolis Savva, and Dhruv Batra. 2019. Dd-ppo: Learning near-perfect pointgoal navigators from 2.5 billion frames. *arXiv preprint arXiv:1911.00357*.

Kangyi Wu, Pengna Li, Jingwen Fu, Yizhe Li, Yang Wu, Yuhan Liu, Jinjun Wang, and Sanping Zhou. 2025a. Event-equalized dense video captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8417–8427.

Kangyi Wu, Pengna Li, Jingwen Fu, Yang Wu, Yuhan Liu, Sanping Zhou, and Jinjun Wang. 2025b. Cemnet: Cross-emotion memory network for emotional talking face generation. *Preprint*, arXiv:2508.12368.

Yang Wu, Ye Deng, Pengna Li, Wenli Huang, Kangyi Wu, Xiaomeng Xin, and Jinjun Wang. 2025c. Att-cr: Adaptive triangular transformer for cloud removal. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*.

Yang Wu, Ye Deng, Sanping Zhou, Yuhan Liu, Wenli Huang, and Jinjun Wang. 2024. Cr-former: Single-image cloud removal with focused taylor attention. *IEEE Transactions on Geoscience and Remote Sensing*, 62:1–14.

Fei Xia, Amir R. Zamir, Zhi-Yang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. 2018. Gibson env: real-world perception for embodied agents. In *Computer Vision and Pattern Recognition (CVPR), 2018 IEEE Conference on*. IEEE.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Zhengyuan Yang, Linjie Li, Kevin Lin, Jianfeng Wang, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. 2023. The dawn of lmms: Preliminary explorations with gpt-4v (ision). *arXiv preprint arXiv:2309.17421*.

Hang Yin, Xiuwei Xu, Zhenyu Wu, Jie Zhou, and Jiwen Lu. 2024. Sg-nav: Online 3d scene graph prompting for llm-based zero-shot object navigation. *Advances in neural information processing systems*, 37:5285–5307.

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

Yuqian Yuan, Hang Zhang, Wentong Li, Zesen Cheng, Boqiang Zhang, Long Li, Xin Li, Deli Zhao, Wenqiao Zhang, Yueting Zhuang, and 1 others. 2025. Videorefer suite: Advancing spatial-temporal object understanding with video llm. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 18970–18980.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. 2023. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 11975–11986.

Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. 2024. Video instruction tuning with synthetic data. *arXiv preprint arXiv:2410.02713*.

Duo Zheng, Shijia Huang, Lin Zhao, Yiwu Zhong, and Liwei Wang. 2024. Towards learning a generalist model for embodied navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13624–13634.

Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. 2017. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3357–3364. IEEE.

Appendix

This supplementary material provides additional details on the proposed method and experimental results that could not be included in the main manuscript due to page limitations.

Specifically, the structure of this appendix is organized as follows:

Section A discusses the potential risk and data consent in the checklist.

Section B provides more method details, including the high-level/low-level modules, navigation policy, prompts for dataset construction, and the wandering suppression penalty design.

Section C summarizes experimental settings, including evaluation metrics, datasets, implementation details and data quality control details.

Section D reports additional results omitted from the main paper, including fine-grained cross-domain evaluation, reward-weight ablations, efficiency analysis and more qualitative visualizations.

A Checklist

A.1 Potential risk

Our work targets image-goal navigation in indoor environments and is primarily evaluated in simulation, so the overall risk profile is limited. The main practical risk is that behaviors learned in simulation may not fully transfer to real robots due to differences in sensing and embodiment, which could lead to occasional navigation inefficiency or minor unintended contacts in cluttered spaces. In addition, the high-level vision-language reasoning module may inherit typical robustness limitations of large models, resulting in degraded performance under rare visual conditions. These issues are not specific to our method and can be mitigated with cautious deployment (e.g., safe-speed operation, collision avoidance layers) and improved sim-to-real modeling in future work.

A.2 Data Contains Personally Identifying Info Or Offensive Content

Our experiments primarily use standard simulation datasets (Gibson, MP3D, HM3D), which contain reconstructed indoor environments and do not include personal identifiers such as names, faces, voices, or other uniquely identifying information. For our real-world demonstrations, we collected only egocentric RGB observations of indoor scenes in a controlled setting.

A.3 Data Consent

We use only publicly released datasets that are obtained through the official application/registration process and downloaded under the corresponding licenses/terms of use. We did not collect or curate new data from identifiable individuals. Consent for any original data collection is governed by the dataset providers, and our usage follows the permitted research use terms specified by the official data access agreements.

A.4 Information About Use Of AI Assistants

We used an AI assistant to support scientific writing and language polishing (e.g., improving clarity, grammar, and formatting). All technical content, experimental design, results, and claims were produced and verified by the authors, who also reviewed and edited the final manuscript to ensure correctness and originality.

B More Method Details

B.1 High-level reasoning module

We follow the model design of NaVILA (Cheng et al., 2024): a vision encoder (SigLIP (Zhai et al., 2023)) extracts visual tokens from the history frames, current observation, and goal image; an MLP projector maps the visual tokens into the language embedding space; and an LLM (LLaMA3 (Dubey et al., 2024)) policy head performs auto-regressive decoding to predict the next sub-task instruction.

B.2 Low-level prediction module

For the semantic sub-task channel, we encode the predicted sub-task instruction, the current observation, and the goal image using CLIP text/vision encoders (ResNet50 backbone) (He et al., 2016; Radford et al., 2021; Li et al., 2024a). For the navigation-specific channel, we concatenate the current observation and the goal image along the channel dimension and extract navigation features. The navigation encoder is initialized with a ResNet50 (He et al., 2016) pretrained in Imagenet (Deng et al., 2009). We fuse the semantic features and navigation features via a 2-layer MLP fusion module before feeding them to the navigation policy.

B.3 Navigation policy

The navigation policy is composed of a 2-layer GRU with a 128-dimensional embedding size. We

```
"""
You are a Visual-Language Navigator. Your task is to break down the given instruction into multiple sub-
instructions. Each sub-instruction should contain a simple and specific action or a target to reach.
Here is the instruction provided by the user:
<instruction>
{}
</instruction>

When breaking down the instruction, please follow these rules:
- Each sub-instruction should have a single, clear action or a clear target.
- Sub-instructions should be non-overlapping and as simple as possible.
Please put your breakdown results below:
"""
```

Figure 6: Sub-task decomposition prompt. We prompt the LLM to break down a long navigation instruction into a sequence of atomic sub-instructions.

```
"""
The first-person video is captured by a robot completing a series of instructions in chronological order.
The instructions are {sub_instruction_prompt}.
The robot finishes them one-by-one.
We put the frame number at the top left of the video.
First, you need to describe how the robot moves in the video. You should specify the frame number in this
part.
Finally, You need to do temporal grounding for each of the instruction.

The output format is :
<Description>
</Description>
<Temporal Grounding>
# Instruction1: from frame ? to frame ?;
# Instruction2: from frame ? to frame ?;
........
</Temporal Grounding>
"""
```

Figure 7: Temporal grounding prompt and annotated video clip. **Top:** The prompt instructs the model to assign a temporal interval to each sub-instruction using a structured output format. **Bottom:** an example input video clip used for grounding. To facilitate temporal reasoning, we overlay the *frame number* and the *executed action* at the top-left corner of each frame.

train the navigation policy using reinforcement learning. To enhance the generalization ability of the agent, we follow previous reinforcement learning methods (Mezghan et al., 2022; Al-Halah et al., 2022; Wu et al., 2024) to apply data augmentation to the visual observation and goal images. We adopt two kinds of augmentations: (1) random cropping, which enlarges the input image and then selects a random portion of the original size; (2) colour jitter, which randomly adjusts the brightness, contrast, saturation, and hue of the image. During navigation, the agent obtains the current observation from the sensors at each step. We sequentially apply both data augmentation methods to these images, which can enhance the visual diversity in the training data.

After the data augmentation, the visual observation and goal images are fed into the semantic and navigation channel to obtain fused features. The agent receives the fused features f_{fused} and passes them into the navigation policy π. The policy utilizes the fused features along with a representation of previous actions to predict the agent's current state embedding s_t at time t, and we formulate this as follows:

$$s_t = \pi(f_{fused} \oplus a_{t-1} \mid h_{t-1}), \tag{6}$$

where a_{t-1} denotes the representation of previous actions and h_{t-1} is the hidden layer of the recurrent layers in the policy from the previous step. An actor-critic network then utilizes s_t to predict the state value and determine the agent's next action.

B.4 Prompts for Dataset Construction

For training the high-level reasoning module, we construct a hierarchical reasoning dataset from existing VLN trajectory data. As described in Sec. 3.2, we prompt Qwen3 (Yang et al., 2025) to decomposes a long-horizon instruction into a sequence of non-overlapping, atomic sub-instructions, where each sub-instruction contains a single clear action or a specific target to reach. Then we prompt visual language model Qwen2-VL (Bai et al., 2025) to performs temporal grounding by analyzing the changes of first-person observations over time and selecting which sub-instruction is currently being carried out. For readability, we optionally visualize these prompts and the corresponding input/output format.

Sub-task decomposition prompt As illustrated in Fig. 6, we prompt the LLM to break down the given navigation instruction into multiple sub-instructions with clear and minimal objectives, ensuring each sub-instruction is simple and does not overlap with others.

Temporal grounding prompt. The prompt is shown in Fig. 7. Given the ordered sub-instructions and a first-person video clip from time 0 to t (each video frame is augmented with the frame number and the action executed at that time step.), we prompt the VLM to infer which sub-instruction the agent is executing based on *temporal changes* in the observed frames, rather than static scene recognition. This grounding step produces an alignment between sub-instructions and temporal segments.

B.5 ZER Reward

ZER Reward. Following ZER (Al-Halah et al., 2022), we use a dense distance-and-view shaping reward together with a sparse success reward. After taking action a_t, the step reward is

$$r_t = (d_{t-1} - d_t) + \mathbb{I}(d_t \le d_s)(\alpha_{t-1} - \alpha_t) - \gamma \quad (7)$$

where d_t is the geodesic distance to the goal, α_t is the minimum angular difference (in radians) between the agent view and the goal views and γ is the slack reward for efficiency. The view term is activated only within the success area ($d_t \le d_s$, where d_s is the success distance). The sparse success reward is

$$R_s = 5\Big[\mathbb{I}(d_t \le d_s) + \mathbb{I}(d_t \le d_s \wedge \alpha_t \le \alpha_s)\Big], \quad (8)$$

with $d_s = 1$m and $\alpha_s = 25°$. The agent receives a sparse reward when it reaches the success area and stops within an view angle α_s from the goal angle.

Although the ZER reward provides dense shaping, its distance term $(d_{t-1} - d_t)$ is local and does not explicitly penalize redundant motions: when the agent explores ambiguous corridors or makes wrong turns, it can easily fall into short-range oscillations (e.g., left-right rotations or backtracking), where the net progress in geodesic distance becomes small and noisy.

B.6 Wandering Suppression Penalty Details

Path-length penalty. The Path-length penalty penalize unnecessarily long trajectories by incorporating the cumulative traveled length into the potential. Let $\mathbf{p}_t \in \mathbb{R}^3$ be the agent position at time step t. We define the step displacement as the Euclidean distance between two consecutive positions:

$$\Delta\ell_t = \|\mathbf{p}_t - \mathbf{p}_{t-1}\|_2, \quad (9)$$

and accumulate it over time to obtain the traveled path length:

$$\ell_t = \sum_{\tau=1}^{t} \Delta\ell_\tau, \qquad \ell_0 = 0. \quad (10)$$

This term is monotonic increasing and thus yields a consistent penalty.

Revisit penalty. In this part, we provides more design details about the revisit penalty. At each time step t, let $\mathbf{p}_t \in \mathbb{R}^3$ denote the agent position. We discretize $\mathbf{p}_t$ into a voxel index (grid key) κ_t with resolution s by

$$\kappa_t = \big(\lfloor p_x/s \rfloor,\ \lfloor p_y/s \rfloor,\ \lfloor p_z/s \rfloor\big), \quad (11)$$

where s corresponds to `revisit_radius` in our implementation and it is set to 0.25. We maintain a visited set $\mathcal{V}$ that stores voxel keys observed so far in the episode. The revisit cost is then computed as

$$c_t = \lambda_{\text{rv}}\, \mathbb{I}(\kappa_t \in \mathcal{V}), \qquad \mathcal{V} \leftarrow \mathcal{V} \cup \{\kappa_t\}, \quad (12)$$

where λ_{rv} corresponds to `revisit_weight`, which is set to 0.02. Intuitively, whenever the agent returns to a previously visited voxel, it receives an additional penalty, discouraging redundant backtracking and local dithering.

Finally, the path length and revisit cost are integrated into the shaping function as

$$\Phi_t = \lambda_w\big(\ell_t + c_t\big) + d_t + \alpha_t, \quad (13)$$

where ℓ_t is the cumulative traveled length, c_t is the reivisit cost, d_t is the geodesic distance to the goal, and α_t is the view-angle distance to the closest goal view. λ_w is the hyperparameter weight. The step reward is obtained by the potential difference, i.e., $r_t \leftarrow r_t + (\Phi_{t-1} - \Phi_t)$, so that our new design reward consistently decrease the reward and thus suppress wandering.

Using Wandering Suppression Penalty (WSP). While WSP provides consistent negative feedback for redundant motions, directly enabling it from scratch may harm early-stage exploration. In the initial phase of PPO training, the policy is largely uninformative and relies on stochastic exploration to discover trajectories that reduce d_t and eventually enter the success area. If WSP is applied too early, the additional penalties on traveled length and revisits can dominate the learning signal, making "doing nothing" (or minimal motion) a locally attractive behavior since it avoids accumulating ℓ_t and c_t. This may lead to overly conservative policies and slow down (or even stall) learning.

To mitigate this issue, we adopt a two-stage training schedule. We first pretrain the policy using the original ZER reward (Eqs. (7)–(8)) until the navigation behavior becomes reasonably stable (i.e., the agent consistently makes progress toward the goal). Then, we activate WSP and continue training with the combined reward. In this second stage, the policy already has a meaningful notion of goal-directed movement, and WSP mainly acts as a regularizer that suppresses dithering and detours, thereby reducing wandering and improving trajectory efficiency.

In practice, we find that WSP takes effect quickly once activated. After the initial warm-up with the ZER reward, enabling WSP and continuing training for about 10M environment steps is sufficient to noticeably suppress wandering behaviors.

C Experimental Details

C.1 Evaluation metric details

We evaluate image-goal navigation performance with the success rate (SR) and success weighted by path length (SPL) (Anderson et al., 2018).

SR (Success Rate): SR evaluates the success rate of the agent in successfully navigating to the target location. The maximum success distance is 1m from the real target location. SR is defined by:

$$SR = \frac{1}{N_e} \sum_{i=1}^{N_e} S_i, \tag{14}$$

$$S_i = \begin{cases} 1, \text{ if the } i\text{th episode succeeds,} \\ 0, \text{ if the } i\text{th episode fails,} \end{cases} \tag{15}$$

where N_e represents the total number of test episodes and S_i indicates whether the ith episode is successful.

SPL (Success weighted by Path Length): SPL considers the navigation path length along with the success rate:

$$SPL = \frac{1}{N_e} \sum_{i=1}^{N_e} S_i \frac{l_i}{\max(p_i, l_i)}, \tag{16}$$

where l_i is the shortest path distance from the agent's starting position to the goal location in the ith episode and p_i is the predicted navigation path distance executed by the agent.

C.2 Dataset details

We trained REGNav in the Gibson dataset (Xia et al., 2018) and inference in the Gibson, Matterport3D (MP3D) (Chang et al., 2017) and HM3D (Ramakrishnan et al., 2021) datasets. Specifically, Gibson is composed of full indoor environments, which have multi-rooms. The training set in Gibson comprises 9000 episodes sampled from 72 Gibson training scenes. These episodes are evenly distributed across three difficulty levels based on the goal location's geodesic distance from the starting location: easy (1.5–3m), medium (3–5m), and hard (5–10m). The test set contains 4200 episodes sampled from 14 unseen scenes and covers the same three difficulty levels. The test scenes are separate from the training scenes to evaluate the agent's ability to generalize to previously unseen environments. Compared with Gibson, MP3D comprises more complex and expansive scenes and HM3D provides more diverse scenes. For evaluation on MP3D and HM3D, we adopt the same test splits as ZER (Al-Halah et al., 2022). The test set of MP3D includes 1000 episodes of 100 scenes per difficulty level while HM3D contains 1000 episodes of 18 scenes per level.

C.3 Inplementation Details

For the high-level module training stage, we use a learning rate of 1e4 with cosine decay and a warm-up ratio of 0.03. The low-level modules are trained

Data	# Samples	Format Compliance	Temporal Consistency	Semantic Grounding
Before TQCM	1328K	93.2%	87.2%	84.2%
After TQCM	767K	99.9%	99.6%	98.6%

Table 7: Quality evaluation of the generated Hierarchical Reasoning Dataset. TQCM substantially improves format compliance, temporal consistency, and semantic grounding accuracy.

end-to-end using the Adam optimizer (Kingma, 2014) with DD-PPO (Wijmans et al., 2019), using 64 forward steps, an entropy coefficient of 0.01, and a clipping value of 0.2. The slack reward γ is set to -0.01. In the simulation environments, the height of agent is set to 1.5m and the radius is 0.1m. The agent has a single RGB sensor with a 90° FOV and 128×128 resolution. The action space consists of MOVE_FORWARD by 0.25m, TURN_LEFT, TURN_RIGHT by 30° and STOP. Our HRNav is trained on a server with 8 NAVIDIA H20 GPUs. The high-level module finetuning requires 64 hours, and the low-level module requires 30 hours. During inference, we run the low-level policy for 15 steps, and then invoke the high-level module once. We will release both our training code and data upon paper publication.

C.4 Dataset Quality Control

Since the Hierarchical Reasoning Dataset is automatically constructed with LLMs and VLMs, we further conduct a quality-control evaluation to assess the reliability of the generated temporal grounding and sub-task annotations. We consider three complementary aspects: format compliance, temporal consistency, and semantic grounding accuracy.

Format Compliance. We first evaluate whether the generated temporal grounding annotations follow the required structured format. Each annotation line is expected to specify the sub-task index and its corresponding temporal interval, e.g., *# Instruction1: from frame 0 to frame 8;* or *# Instruction1: from frame 0 onwards;*. An annotation is considered format-compliant if all its lines can be successfully parsed into a valid tuple consisting of the instruction index, start frame, and end frame or onwards marker. In addition, the instruction indices are required to be continuous and start from 1, without missing or duplicated indices. This metric reflects whether the generated annotations can be reliably used for downstream dataset construction.

Temporal Consistency. We then check whether the grounded sub-tasks are temporally consistent with the navigation trajectory. After sorting the parsed intervals by their instruction indices, we require the start frames of consecutive sub-tasks to be strictly increasing. We also require that adjacent intervals do not overlap under the end-exclusive interval convention, i.e., an interval from frame s to frame e is interpreted as $[s, e)$. For annotations with an *onwards* marker, the interval is extended to the last available frame of the trajectory. An annotation is considered temporally consistent only if all sub-task intervals satisfy the above ordering and non-overlap constraints. This metric evaluates whether the decomposed sub-tasks form a valid temporal progression along the trajectory.

Semantic Grounding Accuracy. Finally, we evaluate whether each grounded sub-task is semantically supported by the visual observations within its assigned temporal interval. For each sub-task interval, we uniformly sample several frames from the interval and provide them, together with the corresponding sub-task instruction, to a VLM judge. The judge is prompted to determine whether the instruction is visually supported by the sampled frames and to return a structured JSON output containing a binary consistency decision, evidence frames, confidence score, and a short reason. We report semantic grounding accuracy as the percentage of judged sub-task intervals that are classified as semantically consistent. This metric measures whether the temporal interval not only satisfies structural constraints but also visually corresponds to the intended navigation sub-task.

Analysis. As shown in Table 7, the raw automatically generated annotations already achieve reasonable quality, but still contain non-negligible noise. Before quality control, 93.2% of the annotations satisfy the required output format, while 87.2% pass the temporal consistency check, indicating that some generated sub-task intervals contain malformed structures, missing or duplicated indices, or overlapping temporal boundaries. The

Method	MP3D							
	Easy		Medium		Hard		Overall	
	SPL↑	SR↑	SPL↑	SR↑	SPL↑	SR↑	SPL↑	SR↑
FGPrompt-MF	56.4%	88.1%	44.6%	77.8%	32.0%	60.1%	44.3%	75.3%
FGPrompt-EF	61.6%	88.9%	50.6%	78.7%	34.1%	59.5%	48.8%	75.7%
REGNav	<u>61.7%</u>	<u>90.0%</u>	<u>52.0%</u>	<u>81.4%</u>	<u>36.9%</u>	<u>62.7%</u>	<u>50.2%</u>	<u>78.0%</u>
HRNav	**64.6%**	**90.9%**	**58.3%**	**84.2%**	**45.9%**	**69.1%**	**56.3%**	**81.1%**

Table 8: **Fine-grained cross-domain evaluation on MP3D.** We compare HRNav against FGPrompt (Sun et al., 2024) and REGNav(Li et al., 2025b) across three difficulty levels on MP3D (Chang et al., 2017). Each difficulty level contains 1000 episodes sampled from 100 unseen scenes. All methods are trained on Gibson (Xia et al., 2018) and directly transferred to MP3D for zero-shot inference. The **best** and <u>second-best</u> results are highlighted.

semantic grounding accuracy is 84.2%, suggesting that a portion of the automatically grounded intervals do not fully match the visual evidence in the corresponding frames.

After applying the proposed Triple Quality Control Mechanism (TQCM), the annotation quality is significantly improved. The filtered dataset achieves 99.9% format compliance and 99.6% temporal consistency, showing that the remaining annotations are structurally valid and temporally well ordered. The semantic grounding accuracy also increases to 98.6%, indicating that most retained sub-task annotations are visually supported by their grounded frame intervals. Although this filtering process reduces the dataset size from 1328K to 767K samples, it removes noisy annotations and yields a cleaner training corpus for high-level planning. This quality-control process improves the reliability and reproducibility of the constructed dataset, which is important for training the VLM-based slow planner.

D Additional Results

We provide additional experimental results that are omitted from the main paper due to space constraints, including: (i) fine-grained evaluation on MP3D and HM3D (per difficulty level), (ii) ablation on the weights of the wandering suppression penalty, and (iii) more qualitative visualizations.

D.1 Cross-domain evaluation on different difficulty levels.

To examine the cross-domain generalization of image-goal navigation, we report fine-grained results on MP3D and HM3D by stratifying episodes into three difficulty levels (Easy/Medium/Hard), where the *Hard* split contains the longest trajectories and requires traversing multiple rooms. As shown in Table 8 and Table 9, all methods are trained on Gibson and directly transferred to unseen scenes for zero-shot evaluation. We compare HRNav against the FGPrompt baseline (Sun et al., 2024) and REGNav (Li et al., 2025b).

Analysis. HRNav consistently outperforms prior methods across all difficulty levels on both datasets, with the most pronounced gains on **Hard** trajectories. On MP3D, HRNav achieves **45.9% SPL** and **69.1% SR** on the Hard split, surpassing the strongest baseline (REGNav) by **+9.0 SPL** and **+6.4 SR**, respectively. On HM3D, HRNav further improves to **41.3% SPL** and **66.5% SR** on Hard, yielding **+6.4 SPL** and **+3.7 SR** over the best competing method. These results indicate that the advantage of HRNav becomes larger as the navigation horizon increases: when the goal is far from the start and direct visual matching becomes less reliable, HRNav's hierarchical reasoning and wandering suppression penalty better reduce wandering and help the agent maintain consistent progress toward the goal, leading to higher success and more efficient paths in the most challenging settings.

D.2 Ablation study on weights of wandering suppression penalty.

We ablate the weight λ_w of the wandering suppression penalty (WSP) to study its influence on navigation performance. Specifically, we train the same model on Gibson while only varying $\lambda_w \in \{1.0, 0.8, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1\}$, and report SR and SPL on the Gibson validation set under the standard evaluation protocol. All other training configurations (network architecture, RL hy-

Method	HM3D							
	Easy		Medium		Hard		Overall	
	SPL↑	SR↑	SPL↑	SR↑	SPL↑	SR↑	SPL↑	SR↑
FGPrompt-MF	44.0%	83.2%	40.8%	75.6%	31.8%	62.6%	38.9%	73.8%
FGPrompt-EF	47.3%	85.0%	44.2%	77.8%	<u>34.9%</u>	<u>62.8%</u>	42.1%	75.2%
REGNav	<u>50.6%</u>	<u>85.4%</u>	<u>47.0%</u>	<u>78.9%</u>	34.3%	61.3%	<u>44.0%</u>	<u>75.2%</u>
HRNav	**54.3%**	**89.6%**	**52.3%**	**83.8%**	**41.3%**	**66.5%**	**49.3%**	**80.0%**

Table 9: **Fine-grained cross-domain evaluation on HM3D.** We compare HRNav against FGPrompt (Sun et al., 2024) and REGNav (Li et al., 2025b) across three difficulty levels on HM3D (Szot et al., 2021). Each difficulty level contains 1000 episodes sampled from 18 unseen scenes. All methods are trained on Gibson (Xia et al., 2018) and directly transferred to HM3D for zero-shot evaluation. The **best** and <u>second-best</u> results are highlighted.

Setting	SR↑	SPL↑
1.0	71.83%	65.86%
0.8	81.02%	70.75%
0.6	85.05%	71.91%
0.5	89.40%	**73.30%**
0.4	88.57%	72.59%
0.3	90.57%	71.83%
0.2	**94.00%**	71.20%
0.1	91.88%	72.68%

Table 10: Ablation study on the weight λ_w of wandering suppression penalty.

perparameters, and data augmentations) are kept identical to ensure a controlled comparison.

Analysis. As shown in Table 10, λ_w exhibits a non-monotonic effect on performance. A large penalty (e.g., $\lambda_w = 1.0$) overly suppresses revisits/backtracking, which can prevent necessary corrective behaviors and results in a clear drop (71.83% SR / 65.86% SPL). As λ_w decreases, both SR and SPL improve substantially, indicating that moderate suppression effectively reduces aimless wandering while preserving flexibility. Notably, $\lambda_w = 0.2$ achieves the **highest SR** of **94.00%** while maintaining competitive SPL (71.20%), reflecting the best overall robustness in goal-reaching. Although $\lambda_w = 0.5$ yields the highest SPL (73.30%), it is accompanied by a lower SR (89.40%), suggesting that a stronger penalty may occasionally hinder recovery from suboptimal states. Therefore, we set $\lambda_w = 0.2$ as the default in all experiments to prioritize consistent success in unseen environments while still keeping path efficiency at a strong level.

D.3 Efficiency Analysis

To assess the practical deployment cost of HRNav, we evaluate its inference efficiency under the same setting as our real-world experiments. Specifically, we measure the average latency per step over 5,000 navigation steps on a server with one NVIDIA H20 GPU while varying the planning interval k of the high-level planner. Here, the *slow latency* denotes the runtime of one VLM-based planning call, the *fast latency* denotes the runtime of one low-level policy step, and the *average latency per step* amortizes the sparse slow-planner calls over the entire navigation process.

Analysis. Table 11 shows that HRNav maintains practical real-time performance despite introducing a VLM-based slow planner. With our default setting of k=15, the system runs at 24.29 FPS with an average latency of 41.16 ms per step, which is sufficient for real-time robotic control. Although each slow-planner call takes around 370–400 ms, the planner is only invoked sparsely, so its overhead is effectively amortized across subsequent execution steps. In contrast, the low-level executor remains consistently lightweight, requiring only 14–16 ms per step across all settings.

The table also reveals a clear efficiency–performance trade-off. Using a smaller planning interval (e.g., k=5) provides slightly stronger navigation performance, but substantially increases average latency because the slow planner is called more frequently. Increasing the interval to 30 or 60 further improves throughput, but leads to a mild drop in SR and SPL due to less frequent planning updates. We therefore choose k=15 as the default setting, since it provides the best balance between navigation performance and computational

Step Interval (k)	SR (%)	SPL (%)	Avg Latency (ms)	FPS	Slow-latency (ms)	Fast-latency (ms)
5	94.7	71.5	96.81	10.33	395.22	15.92
10	94.4	71.1	55.08	18.13	383.28	14.45
15 (Ours)	94.0	71.2	41.16	24.29	374.12	14.22
30	93.4	70.6	29.75	33.61	389.47	15.08
60	92.2	69.8	24.26	41.22	408.27	15.41

Table 11: Efficiency analysis under different planning intervals. The high-level planner is invoked once every k steps. The reported average latency per step amortizes the sparse VLM calls over the full navigation trajectory.

efficiency.

D.4 More visualization

Qualitative visualizations. We provide additional qualitative results in both simulation and real-world environments to better illustrate the behavior of HRNav. Fig. 8 shows a representative simulated episode in a top-down view, where we overlay the agent trajectory to reveal its global navigation pattern. Fig. 9 presents a real-world episode with third-person and egocentric views along the timeline, highlighting how the robot progresses under the sub-task plan.

Analysis. Across both settings, HRNav exhibits more goal-directed trajectories with fewer redundant backtracks, indicating effective wandering suppression and stable long-horizon execution. In simulation, the top-down view clearly shows that the agent follows a smoother and more direct route toward the goal, rather than oscillating between nearby locations. In the real world, HRNav maintains consistent progress by executing actions conditioned on the current sub-task, and switches subtasks smoothly as the scene evolves. These visualizations support our quantitative results, suggesting that hierarchical reasoning with subtask-conditioned control improves robustness, particularly in challenging long-horizon navigation.

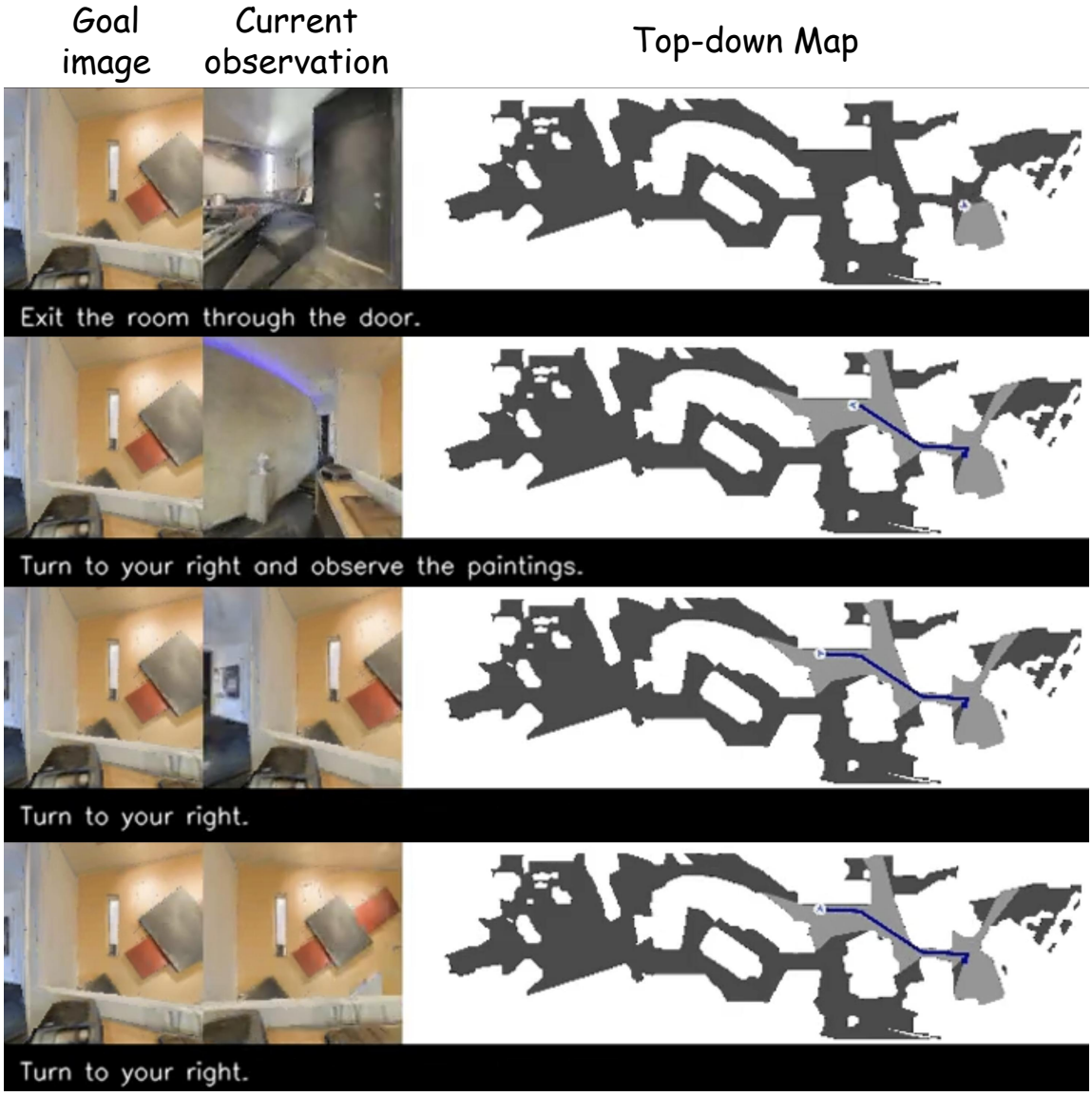

Figure 8: **Simulation qualitative result (top-down view).** Visualization of an example episode in simulation. The agent trajectory is shown in a global top-down map, illustrating the navigation progress toward the goal and the overall path efficiency.

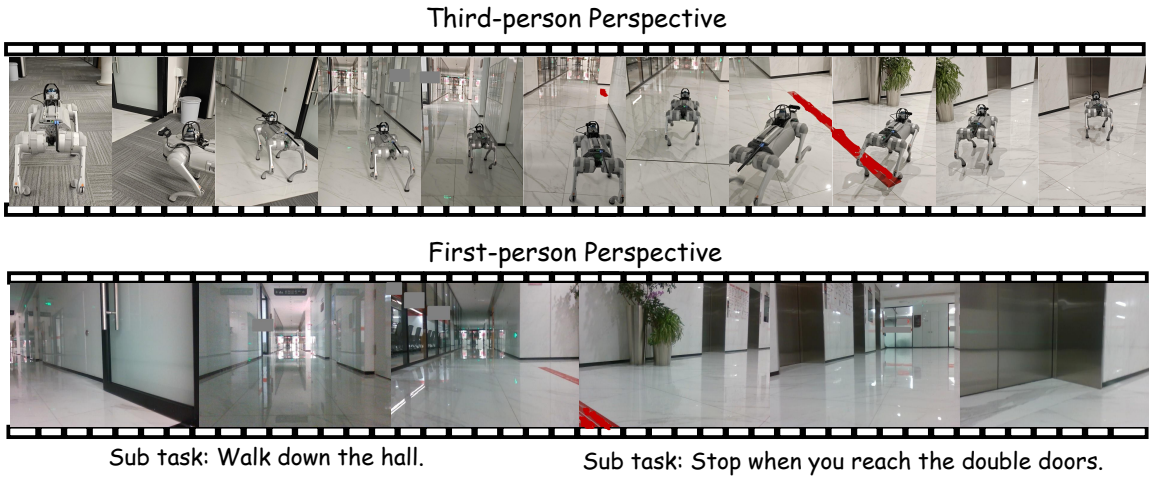

Figure 9: **Real-world qualitative result.** Visualization of an example episode in a real environment. We show third-person recordings and the robot's egocentric observations along the timeline, together with the predicted sub-task sequence, to illustrate subtask-conditioned execution and successful goal reaching.