CityNavAgent: Aerial Vision-and-Language Navigation with Hierarchical Semantic Planning and Global Memory

Anonymous ACL submission

Abstract

Aerial vision-and-language navigation (VLN) — requiring drones to interpret natural language instructions and navigate complex urban environments — emerges as a critical embodied AI challenge that bridges human-robot interaction, 3D spatial reasoning, and real-world deployment. Although existing ground VLN agents achieved notable results in indoor and outdoor settings, they struggle in aerial VLN due to the absence of predefined navigation graphs and the exponentially expanding action space in long-horizon exploration. In this work, we propose **CityNavAgent**, a large language model (LLM)-empowered agent that significantly reduces the navigation complexity for urban aerial VLN. Specifically, we design a hierarchical semantic planning module (HSPM) that decomposes the long-horizon task into sub-goals with different semantic levels. The agent reaches the target progressively by achieving sub-goals with different capacities of the LLM. Additionally, a global memory module storing historical trajectories into a topological graph is developed to simplify navigation for visited targets. Extensive benchmark experiments show that our method achieves state-of-the-art performance with significant improvement. Further experiments demonstrate the effectiveness of different modules of CityNavAgent for zero-shot VLN in continuous city environments.

1 Introduction

Visual-and-language navigation (VLN) is a fundamental task where an agent is required to navigate to a specified landmark or location following language instructions (Anderson et al., 2018; Gu et al., 2022; Gao et al., 2024a). With the increasing prevalence of unmanned aerial vehicles (UAVs), aerial VLN (Liu et al., 2023b) has gained significant attention. This task empowers UAVs to navigate complex, large-scale outdoor environments with language instructions, reducing the cost of human-machine interaction and offering significant advantages in applications like rescue, transportation, and urban inspections.

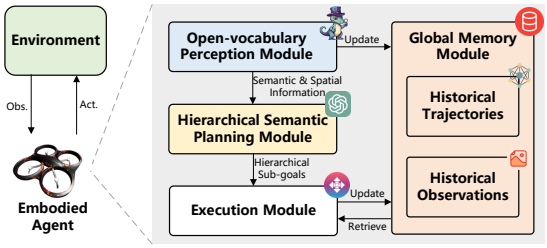

Figure 1: The overall workflow of CityNavAgent.

Most existing methods primarily address indoor VLN. One approach (Anderson et al., 2018; Kurita and Cho, 2020; Chen et al., 2024; Gao et al., 2023; Huo et al., 2023; Chen et al., 2022) formulates the task in a discrete setting, where agents teleport between nodes in pre-defined topological graphs without motion errors, limiting real-world applicability. Other methods mitigate reliance on pre-defined maps using end-to-end action prediction (Krantz et al., 2020; Raychaudhuri et al., 2021; Chen et al., 2021a) or waypoint prediction (Hong et al., 2022; An et al., 2024; Wang et al., 2024). But the former struggles with scene semantic variations, while the latter fails to adapt to large-scale outdoor scenarios. Although some methods (Schumann et al., 2024; Liu et al., 2024) extend VLN to outdoor ground navigation, they still rely on pre-defined scene graphs, which are unavailable in aerial settings. STMR (Gao et al., 2024b) introduced a zero-shot LLM-based framework for aerial VLN by constructing an online 2D semantic map, but its failure to incorporate height information leads to high navigation errors.

In this work, we focus on aerial VLN that has a more realistic and challenging setting compared to the previous VLN tasks. In this task, the agent is required to predict the next action or waypoint to approach to the target iteratively in a continuous

aerial space. The challenges are two-fold:

- **Complex scene understanding in urban environments** Urban environments exhibit considerably higher object variety than indoor scenes, incorporating extensive infrastructural elements, architectural structures, and natural landscapes. Moreover, the semantic density in urban scenes is highly dynamic. When an agent operates near ground level, the scene exhibits high semantic density, whereas at higher altitudes the semantic becomes markedly sparse. These disparities in object variety and semantic density pose substantial challenges for cross-modal grounding and instruction-related object extraction.
- **Exponential complexity in long-horizon motion planning.** The VLN task can be considered as a Partially Observable Markov Decision Process, where the agent predicts the next action or waypoint based on its current state and the environmental context. However, for the aerial VLN, the long-horizon navigation requires the agent to predict longer action sequences. Even if the number of available actions per step is limited, the total number of possible action sequences grows exponentially with the planning horizon. Specifically, if there are m actions available at each step, the number of potential action sequences over n steps is approximately m^n, which poses a great challenge to the agent's action planning.

In this work, we propose **CityNavAgent**, which consists of an **open-vocabulary perception module** and a **hierarchical semantic planning module (HSPM)** with a **global memory module** to address the above challenges. 1) To extract the complex and rich semantics in urban environments, the open-vocabulary perception module first utilizes an LLM to caption the scene and extract instruction-related objects through prompt engineering. It then integrates a vision foundation model for open-vocabulary image grounding. 2) To narrow down the possible action space during the motion planning, we design HSPM, which decomposes the navigation task into landmark-level, object-level, and motion-level planning, with progressively decreasing semantic abstraction. The planning frequency decreases from low to high levels. The landmark-level planning decomposes the navigation task into a sequence of landmarks to be traversed. The object-level planning reasons about the objects in the scene that lead to these landmarks. The motion-level planning predicts the waypoint and action sequence to reach the semantic target from higher-level planners. Additionally, CityNavAgent incorporates a global memory module to store effective waypoints and trajectories from historical tasks, enhancing long-term navigation performance.

To summarize, the main contributions of this work are as follows:

- We focus on the urban aerial VLN task, which is insufficiently explored yet valuable and highly challenging, and introduce **CityNavAgent**—an LLM-powered agent for zero-shot navigation.
- We propose an open-vocabulary perception module that enables the agent to understand the urban scene and HSPM with global memory that reduces the complexity of action planning to address the key challenges.
- We conduct extensive experiments on two aerial VLN benchmarks to demonstrate our proposed method in terms of success rate and path following. More ablation studies verify the efficacy of our designed components.

2 Related Works

Vision-and-language Navigation (VLN) VLN is first well defined by R2R (Anderson et al., 2018) which is a navigation benchmark collected in a photorealistic simulator (Chang et al., 2017) with detailed language descriptions and visual observations. Based on R2R, tons of methods (Shridhar et al., 2020; Gao et al., 2023; Huo et al., 2023; Chen et al., 2021b; Kamath et al., 2023; Li and Bansal, 2023; Li et al., 2023; Chen et al., 2022; Guhur et al., 2021; Qi et al., 2021) are proposed to enable the robots with embodied navigation capacity. Specifically, Kurita et al. (Kurita and Cho, 2020) proposed a novel generative approach that predicts the instruction distribution conditioned on the action set. However, R2R and its derivation (Ku et al., 2020; Jain et al., 2019) are defined in limited indoor scenes and discrete action spaces which the agent moves within pre-defined topological graphs. This setting yields its practical application in the continuous real-world space.

Krantz et al. (Krantz et al., 2020) introduced R2R-CE tasks by adapting R2R trajectories for continuous environments. End-to-end methods such as LSTM-based methods (Krantz et al., 2020; Raychaudhuri et al., 2021; Liu et al., 2023b), transformer-based methods (Irshad et al., 2022; Chen et al., 2021a; Krantz et al., 2021), and reinforcement learning-based methods (Wang et al., 2018, 2020) have been explored to improve nav-

igation policies. More recently, waypoint-based methods, such as (Hong et al., 2022; An et al., 2024; Wang et al., 2023a, 2024) have emerged by constructing online maps and waypoints to narrow down the agent's possible locations during navigation. Despite these advancements, end-to-end and waypoint prediction-based methods still face obstacles in adapting to open outdoor environments, primarily due to the differences in spatial structures and the semantic distribution of objects.

In this work, we focus on outdoor aerial VLN (Liu et al., 2023b), a more challenging task with longer navigation paths, more diverse scene semantics, and more complex action spaces. More specifically, unlike ground-level outdoor VLN (Chen et al., 2019; Schumann and Riezler, 2020; Schumann et al., 2024) that operate within discrete action spaces, aerial VLN requires agents to navigate through continuous 3D spatial coordinates, which is a more realistic setting for real-world navigation.

LLM for Embodied Navigation With the rise of LLMs (Touvron et al., 2023; Chiang et al., 2023; Achiam et al., 2023; Brown et al., 2020), many methods (Dorbala et al., 2022; Chen et al., 2023; Schumann et al., 2024; Shah et al., 2023) have leveraged their reasoning capabilities for zero-shot VLN. The main challenge for zero-shot LLM-based methods lies in constructing condensed and structured semantic maps of the environment, such as topological graphs, so that LLMs can reason over the semantic information and predict the next waypoint on these maps. Existing works either use pre-defined semantic maps (Achiam et al., 2023; Zhou et al., 2024; Chen et al., 2024) provided by simulators or predict semantic maps within indoor scenes (Wang et al., 2023a). However, indoor semantic map prediction methods face challenges related to scale and semantic shifts when applied to outdoor environments. STMR (Gao et al., 2024b) proposed an outdoor online 2D semantic map construction pipeline and achieved promising results on aerial VLN. But it fails to leverage the height information of the scene, which is also critical for navigation. In this work, we propose CityNavAgent, which comprises a hierarchical semantic planner that predicts waypoints in outdoor environments in a zero-shot manner, along with a global memory module that stores historical waypoints to enhance long-term navigation.

3 Problem Formulation

Given a language instruction $\mathcal{I}$ and the agent's egocentric observation $\mathcal{O}$, the aerial VLN agent has to determine a sequence of action to reach the target location p_d in a continuous 3D space. At each action step t, the agent follows a policy π taking current observation o_t and instruction $\mathcal{I}$ as input to predict the next action a_t and move to location p_t by its kinematic model $\mathcal{F}$, which is given by:

$$p_t = \mathcal{F}(p_{t-1}, \pi(o_t, \mathcal{I})), \tag{1}$$

Given a sequence of action, the agent reaches a final position, and the success probability P_s of reaching the target p_d is

$$P_s = P(||\mathcal{F}(\pi(p_0, \mathcal{O}, \mathcal{I})) - p_d|| < \epsilon), \tag{2}$$

where $||\cdot||$ is the Euclidean distance and ϵ is the threshold that indicates if the target is reached. Thus, the goal of VLN is to find a policy π^* that maximizes the success rate, given by:

$$\pi^* = \text{argmax}_{\pi} P_s. \tag{3}$$

4 CityNavAgent

In this section, we present the workflow of the proposed CityNavAgent for zero-shot aerial VLN in urban environments. As shown in Figure 1, CityNavAgent framework comprises three key modules. 1) The open-vocabulary perception module extracts structured semantic and spatial information from its panoramic observation via a foundation model. 2) The hierarchical semantic planning module (HSPM) leverages LLM to decompose the navigation task into hierarchical planning sub-tasks to reduce the planning complexity and predict the intermediate waypoint for the execution module. 3) The global memory module, represented as a topological graph, stores valid waypoints extracted from historical trajectories to further reduce the action space of motion planning and enhance long-range navigation stability.

4.1 Open-vocabulary Perception Module

To accurately understand complex semantics in the urban environment, we leverage the powerful open-vocabulary captioning and grounding model to extract scene semantic features. Besides, integrating the scene semantics and depth information, we construct a 3D semantic map for further planning.

Scene Semantic Perception Extracting urban scene semantics requires robust open-vocabulary object recognition. As shown in Figure 2, given a set of panoramic images $\{I^t_{R_i}\}$ at step t, we first

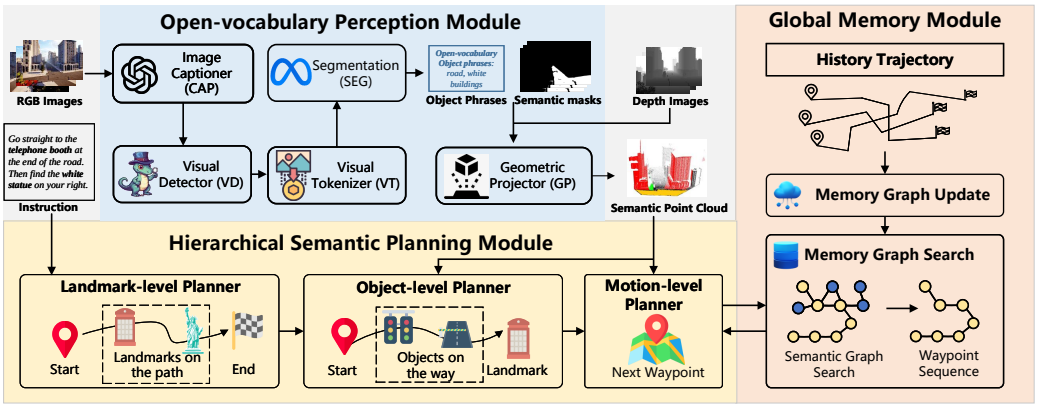

Figure 2: CityNavAgent consists of three key modules. The open-vocabulary module extracts open-vocabulary objects in the scenes and builds a semantic point cloud of the surroundings. The hierarchical semantic planning module decomposes the original instruction into sub-goals with different semantic levels and predicts the agent's next action to achieve the high-level sub-goals. The global memory module stores historical trajectories to assist motion planning toward visited targets.

use an open-vocabulary image captioner CAP$(\cdot)$ empowered by GPT-4V (Achiam et al., 2023) to generate object captions c_i^t for the image $I_{R_i}^t$. Then, we leverage a visual detector VD$(\cdot)$ named GroundingDINO (Liu et al., 2023a), to generate the bounding boxes obb_i^t for captioned objects by

$$obb_i = \text{VD}(c_i^t, I_{R_i}^t), c_i^t = \text{CAP}(I_{R_i}^t). \quad (4)$$

Finally, the bounding boxes are tokenized by a visual tokenizer VT$(\cdot)$, which is then fed into a segmentation model SEG$(\cdot)$ (Kirillov et al., 2023) to generate fine-grained semantic masks $I_{S_i}^t$ for objects as follows:

$$I_{S_i}^t = \text{SEG}(\text{VT}(obb_i), I_{R_i}^t). \quad (5)$$

Scene Spatial Perception Considering that egocentric views suffer from perspective overlap (An et al., 2023) and fail to capture 3D spatial relationships (Gao et al., 2024b), we construct a 3D point map by projecting segmentation masks of observations into metric 3D space. Leveraging the RGB-D sensor's depth map $I_{D_i}^t$ and agent's pose (R, T), a geometric projector (GP) transforms each segmented pixel $p_{ik} = (u, v) \in I_{S_i}^t$ labeled with caption c_{ik}^t into a 3D point P_{ik} via:

$$P_{ik} = R \cdot I_D(u, v) \cdot K^{-1} \cdot p + T, \quad (6)$$

where K is the intrinsic matrix of the camera, while $R \in SO(3)$ and $T \in \mathbb{R}^3$ represent the agent's instantaneous orientation and position in world coordinates. Mapping the object caption from 2D masks to 3D point cloud, a local semantic point cloud $\{(P_{ik}, c_{ik}^t) | i = 1, \ldots, n, k = 1, \ldots, m\}$ is constructed, where n is the number of panoramic images and m is the number of pixels.

4.2 Hierarchical Semantic Planning Module

4.2.1 Landmark-level Planning

Since aerial VLN tasks typically involve long-range decision-making (Liu et al., 2023b; Chen et al., 2019), directly assigning the entire navigation task to the agent can hinder accurate alignment with the instructions and task progress tracking. A more effective approach is to decompose the task into a sequence of manageable sub-goals. By addressing these sub-goals step by step, the agent can progressively reach the final destination. To achieve this, we propose a landmark-level planner driven by LLM (Achiam et al., 2023) to parse free-form instructions T and extract a sequence of landmark phases L along the path through prompt engineering. These landmarks act as sub-goals for the agent. We present a simple prompt as follows (more details in Appendix A):

> *You need to extract a landmark sequence from the given instruction. The sequence order should be consistent with their appearance order on the path.*

4.2.2 Object-level Planning

After landmark-level planning and obtaining a sequence of sub-goals, the object-level planner OP$(\cdot)$ employs the LLM to further decompose these sub-goals into more achievable steps for the agent. The key idea is to leverage the commonsense knowledge of the LLM to reason for the visible object region most pertinent to the invisible sub-goal in the current panorama. This region is referred to as the object region of interest (OROI) in this paper.

For example, if the agent only sees the buildings and a road in the current view while its sub-goal is the traffic light, by commonsense reasoning, the next OROI it should go is the road. We design a prompt that comprises the original navigation instruction T, scene object captions c^t, and current sub-goals L_i for OP$(\cdot)$ to reason for OROI c^t_{OROI}, which is given by:

$$c^t_{OROI} = \text{OP}(T, L_i, c^t), \tag{7}$$

Its template is (more details in Appendix A):

> *The navigation instruction is: .... Your next navigation subgoal is: ... Objects or areas you observed: ...*
>
> *Based on the instruction, next sub-goal, and observation, list 3 objects most pertinent to the subgoal or you will probably go next from your [Observed Objects]. Output should be in descending order of probability.*

We select the OROI with the highest possibility given by LLM to the next landmark as the next waypoint for the agent.

4.2.3 Motion-level Planning

Motion-level planning is responsible for translating the output of high-level planning modules into reachable waypoints and executable actions for the agent. Given a reasoned c^t_{OROI}, the motion-level planner first determines corresponding points $\{(P_k, c^t_k) | c^t_k == c^t_{OROI}\}$ from the semantic point cloud in §4.1 and compute the next waypoint by averaging the coordinates of selected points. Then, the planner decomposes the path to the waypoint into a sequence of executable actions for the agent.

If the agent has reached a location close to the memory graph, the motion planner will directly use the memory graph to predict the agent's future actions, which is introduced in the next section.

4.3 Global Memory Module

Since the agent sometimes revisits the target or landmarks, we designed a global memory module with a memory graph that stores historical trajectories, which helps to reduce the action space in motion planning and improves navigation robustness.

Memory Graph Construction Each historical trajectory H_i can be represented as a topological graph $G_i(N_i, E_i)$ whose nodes N_i encapsulate both the coordinates of the traversed waypoints and their panoramic observations, and edges E_i are weighted by the distance between adjacent waypoints. The memory graph M is constructed by merging all the historical trajectory graphs, given by:

$$\begin{aligned} M &= G(N, E), \\ N &= N_1 \cup \cdots \cup N_d, \\ E &= E_1 \cup \cdots \cup E_d, \end{aligned} \tag{8}$$

where d is the number of historical trajectories.

Memory Graph Update The memory graph is updated progressively by merging newly generated historical trajectory graph G_{hist}. The merging process is similar to Equation 8 merging the nodes and edges of two graphs. In addition, it will generate new edges if M and G_{hist} are connective. We calculate the distance between every pair of nodes in the two graphs. If the distance between any pair of nodes is less than a threshold H=15m, it is inferred that these nodes are adjacent and a new edge is added to the merged graph. Note that the memory graph only merges trajectories that successfully navigate to the target, ensuring the validity of the waypoints in the memory graph

Memory Graph Search for Motion-level Planning When the agent reaches a waypoint in the memory graph, the agent directly leverages the graph to determine a path and action sequence to fulfill the remaining sub-goals from the landmark-level planner. Given a sequence of remaining sub-goals $L^{(r)} = \{l_1, \ldots, l_r\}$ represented by landmark phases, our target is to find a path $V^* = \{v_1, \ldots, v_d\} \subseteq N$ with the highest possibility of traversing L_r in order. Note that each node N_i in the graph contains visual observation o_{v_i} of the surroundings. Therefore, the possibility of traversing landmark l_j by a node N_i can be formulated as $P(l_j|o_i)$. And the objective function can be formulated as:

$$V^* = \max_V \prod_{k=1, 1 \le m_1 < \cdots < m_r \le d}^{r} P(l_k | o_{v_{m_k}}). \tag{9}$$

We leverage the graph search algorithm in LM-Nav (Shah et al., 2023) to solve this problem, which is more detailed in Alg. 1 in Appendix A.2.1. Once V^* is obtained, the agent decomposes it into an executable action sequence to reach the target.

To conclude, with the two specially designed perception and planning modules, along with the memory module, the aforementioned key challenges of the aerial VLN are addressed one by one.

Table 1: Overall performance comparisons on AirVLN-S.

Method	Validataion Seen					Validation Unseen				
	SR↑	SPL↑	OSR↑	SDTW↑	NE↓	SR↑	SPL↑	OSR↑	SDTW↑	NE↓
RS	0.0	0.0	0.0	0.0	109.6	0.0	0.0	0.0	0.0	149.7
AC	0.9	-	5.7	0.3	213.8	0.2	-	1.1	0.3	237.6
LingUNet	0.6	-	6.9	0.2	383.8	0.4	-	3.6	0.9	368.4
Seq2seq	4.8	-	19.8	1.6	146.0	2.3	-	11.7	0.7	218.9
CMA	3.0	-	23.2	0.6	121.0	3.2	-	16.0	1.1	172.1
NavGPT	0.0	0.0	0.0	0.0	163.5	0.0	0.0	0.0	0.0	82.1
MapGPT	2.1	1.5	4.7	0.8	124.9	0.0	0.0	0.0	0.0	107.0
VELMA	0.0	0.0	0.0	0.0	150.5	0.0	0.0	0.0	0.0	117.4
LM-Nav	12.5	9.4	28.5	4.6	81.1	10.4	9.3	33.9	4.7	60.3
STMR	12.6	-	**31.6**	-	96.3	10.8	-	23.0	-	119.5
CityNavAgent	**13.9**	**10.2**	30.2	**5.1**	**80.8**	**11.7**	**9.9**	**35.2**	**5.0**	**60.2**

5 Experiments

5.1 Experimental Setup

Datasets We evaluate CityNavAgent on a novel aerial VLN benchmark named AirVLN-S provided by Liu et al.(Liu et al., 2023b). The benchmark is collected in Unreal Engine 4 to mimic real-world urban environments. It contains 25 different city-level scenes including downtown cities, factories, parks, and villages, with more than 870 different kinds of urban objects. It also consists of 3,916 flying paths collected by experienced UAV pilots.

While AirVLN provides a valuable benchmark, it suffers from ambiguous landmark references in its relatively coarse-grained instructions. This lack of explicit spatial grounding (e.g., "going straight to the buildings") makes it challenging to systematically assess the agent's performance at following each part of the instruction. To address this limitation, we follow the similar idea of (Hong et al., 2020) to enrich the original instruction with sub-instructions and their corresponding paths. We collect 101 fine-grained instruction-path pairs in 10 scenes from AirVLN to construct an instruction-enriched aerial VLN benchmark, named AirVLN-Enriched. The details are in Appendix A.5.

Metrics Following the same metrics used in AirVLN, we report and compare Success Rate (SR), Oracle Success Rate (OSR), Navigation Error (NE), SR weighted by Normalized Dynamic Warping (SDTD) and SR weighted by Path Length (SPL) of tested methods. The task is successfully completed if NE is within 20 meters.

Implmentation Details We take the training samples as the historical tasks and initialize the memory graph by their trajectories. The memory graph remains accessible to the agent throughout the evaluation. In each test case, the agent is spawned at a random location in the scene. It first follows the instruction to explore the environment, and upon reaching the memory graph, it leverages the graph to complete the rest of the navigation path. It has six low-level actions: *Forward*, *Turn Left*, *Turn Right*, *Ascend*, *Descend*, *Stop*. The number of total action steps is counted based on low-level actions. If the agent requires n low-level actions to reach the next waypoint, the action count increases by n. The agent stops when it either triggers the stop action or exceeds the maximum action steps.

Baselines We choose three mainstream types of continuous VLN baselines.

- **Statistical-based Methods.** We use random sample (RS) that agents uniformly select an action from the action space at each step and action sample (AC) that agents sample actions according to the action distribution of the dataset as our baselines.
- **Learning-based Methods.** We choose classic learning-based methdods Seq2Seq (Anderson et al., 2018), CMA (Krantz et al., 2020), and LingUNet (Misra et al., 2018) as our baselines.
- **Zero-shot LLM-based methods.** We use SOTA outdoor VLN methods VELMA (Schumann et al., 2024), LM-Nav (Shah et al., 2023), and STMR (Gao et al., 2024b) as baselines. To validate the effectiveness of indoor VLN methods, we also evaluate SOTA indoor VLN methods: NavGPT (Zhou et al., 2024) and MapGPT (Chen et al., 2024).

5.2 Overall Performance

In Table 1 and Table 2, we report the overall performance of CityNavAgent and baselines on the

Table 2: Overall performance comparisons AirVLN-E.

Methods	SR↑	SPL↑	NE↓
RS	0.0	0.0	129.6
AC	0.0	0.0	290.4
Seq2seq	0.0	0.0	398.5
CMA	0.0	0.0	278.3
NavGPT	0.0	0.0	127.2
MapGPT	3.3	1.5	133.7
VELMA	0.0	0.0	138.0
LM-Nav	23.6	19.2	119.4
CityNavAgent	**28.3**	**23.5**	**95.1**

two aerial VLN benchmark. From these results, we have the following observations:

- **CityNavAgent significantly outperforms previous SOTAs**. 1) Statistical-based methods have the worst performance indicating aerial VLN requires stronger planning capacity rather than random guess. 2) Learning-based methods that predict agent's action directly also have relatively poor performance with SR less than 5%, which can be explained by the complex action space for long-range navigations. 3) Indoor LLM-based methods suffer significantly performance drop while outdoor LLM-based methods have better performance. 4) Compared to these baselines, CityNavAgent outperforms the best of them by 1.3%, 0.8%, 0.5%, and 16.1% in SR, SPL, SDTW and NE for validation seen dataset and by 0.9%, 0.6%, 1.3%, and 0.2% in SR, SPL, SDTW and NE for validation unseen dataset. It demonstrates that the semantic hierarchical planning and memory graph-based motion planning improve the agent's long-range navigation capacity.
- **CityNavAgent has better instruction-following performance.** We can observe that our proposed CityNavAgent achieves the highest SPL and STDW, which outperforms the best of baselines by 0.8% and 0.5%, respectively. We explain the cause of this result as: 1) the HSPM decomposes the original long-distance navigation task into shorter sub-navigation tasks, reducing the planning difficulty. 2) memory graph-based motion planning further guides the agent to traverse the decomposed landmark sequence.
- **Enriched Instructions Promote Navigation Performance.** The performance of MapGPT, LM-Nav, and CityNavAgent on AirVLN-E is better than on AirVLN-S. The highest improvements in SR and SPL are 14.4% and 13.3%, respectively, indicating that the enriched instructions provide clearer landmarks, helping the agent follow the path to the target more effectively.

Table 3: Effectiveness of different modules in CityNavAgent. MG and SM represent the memory graph and semantic map, respectively.

Modules	SR↑	SPL↑	NE↓
w/o MG	11.7	9.1	206.1
w/o SM	23.6	19.2	119.4
w/ GPT-3.5	23.3	16.1	98.9
w/ GPT-4V (ours)	28.3	23.5	95.1

Table 4: Comparison of waypoint predictor under different scenarios.

	Inputs	$\lvert\triangle\rvert$	$d_{rel}\downarrow$	$d_C\downarrow$	$d_H\downarrow$
Outdoor	RGBD	1.66	0.88	6.46	5.16
	RGB	1.59	0.88	**6.40**	**5.15**
	Depth	1.60	0.88	6.48	5.26
Indoor	RGBD	1.40	-	1.05	2.01
	RGB	1.38	-	1.08	2.03
	Depth	1.39	-	**1.04**	**2.01**

5.3 Ablation Study

Effect of semantic map-based exploration. To evaluate the effectiveness of semantic map-based waypoint prediction, we substitute this module with a random walk strategy. As shown in Table 3, the agent without the semantic map (second row) suffers a 4.7% and 4.3% drop in SR and SPL, and a 25.6% increase on NE with CityNavAgent (last row). This result reveals that the semantic map extracts structured environmental information, facilitating the LLM in commonsense reasoning so that the agent navigates to the region or objects that are more relevant to the navigation task. Consequently, the accuracy and efficiency of the navigation is improved.

Effect of memory graph-based exploitation. In this case, we omit the global memory module during the navigation and replace the graph search algorithm with the random walk strategy. Presented in the first row in Table 3, the lack of memory graph results in a 16.6%, 14.4% decrease in SR, SPL, and a 116.7% increase in NE over CityNavAgent. Moreover, the memory graph has a more noticeable impact on the agent's navigation performance compared with the semantic map. This indicates that the memory graph effectively prevents the agent from falling into dead ends or engaging in blind exploration in long-distance outdoor navigation sce-

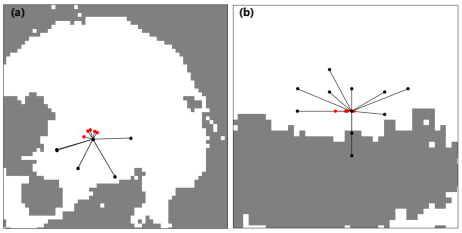

Figure 3: The qualitative result of CWP. From left to right are the top-down view and the front-facing view. ***Red*** and ***black*** circles denote predicted waypoints and reference waypoints on real trajectories.

narios, thereby ensuring the stability of navigation performance.

Effect of different LLMs. We also evaluate the effectiveness of different LLMs for commonsense reasoning in object-level planning (§4.2). Although CityNavAgent with GPT-3.5 demonstrates a competitive performance, replacing GPT-3.5 with GPT-4V which has enhanced reasoning capability results in further performance improvement, *e.g.*, 5.0% and 7.4% increases in SR and SPL, respectively. We attribute this improvement to the fact that GPT-4V has a lower hallucination rate and stronger reasoning ability. Thus, it generates more contextually appropriate responses based on the semantic map and navigation instruction to facilitate the agent in exploring areas most relevant to the target.

5.4 Effectiveness of Indoor Waypoint Prediction

To evaluate the effectiveness of the waypoint prediction method CWP (Hong et al., 2022) in previous continuous VLN methods (An et al., 2024; Koh et al., 2021; Wang et al., 2023a,b; Krantz and Lee, 2022; Wang et al., 2024), we compare the predicted waypoint with target waypoints in outdoor environments. We apply waypoint metrics (Hong et al., 2022) to assess the quality of predicted waypoints. $|\triangle|$ measures the difference in the number of target waypoints and predicted waypoints. d_{rel} measures the ratio of average waypoint distance. d_C and d_H are the Chamfer distance and the Hausdorff distance, respectively. As depicted in Table 4, CWP achieves the best performance in indoor environments with 1.04 d_C and 2.01 d_H while in outdoor environments with 6.4 d_C and 5.15 d_H. This result indicates that although the predicted indoor waypoints by CWP are close to the indoor target waypoints, predicted outdoor waypoints are far from outdoor target waypoints, which is illustrated intuitively in Figure 3(a). We attribute this

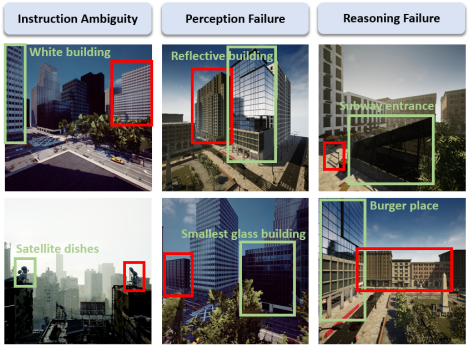

Figure 4: The qualitative result of failure cases. The ***green*** captions and bounding boxes are the referred landmarks in instructions. The ***red*** bounding boxes are the misreferred landmarks due to three failure reasons.

to the scale difference between indoor and outdoor environments. Besides, the dimensional difference is another negative factor for CWP. Depicted in Figure 3(b), CWP only predicts waypoints in 2D space and cannot be applied to open urban environments.

5.5 Case Analysis

In this section, we analyze the failure cases across different datasets and categorize them into three types: 1) Instruction Ambiguity: The navigation instruction lacks clear landmarks, or there are many similar landmarks within the same scene, making it difficult for the agent to accurately refer to the landmark mentioned in the instruction. 2) Perception Failure: Despite the strong object recognition and detection capabilities of our open-vocabulary perception module, outdoor scenes still present many edge cases, leading to incorrect identification of landmarks referenced in the instruction. 3) Reasoning Module: During the hierarchical planning process, the object-level planner may encounter reasoning errors when attempting to infer the location of OROI. This can happen when there is insufficient semantic connection between the objects in the scene and the referenced landmarks, resulting in incorrect OROI reasoning. The visualization results of these failure cases are shown in Figure 4. More qualitative results are in Appendix A.6.3.

6 Conclusion

In this paper, we approach the problem of zero-shot vision-language navigation by proposing an embodied aerial agent, CityNavAgent, which leverages the pre-trained knowledge in large foundation models and historical experience to deal with long-term navigation in urban spaces. The experimental results illustrate the efficacy and robustness of our method from different perspectives.

Limitations

One limitation of our work is that the whole system has not been deployed on a real drone. Though our methods achieve promising results in simulated outdoor environments, low-level motion control problems such as self-pose estimation, control latency, and control errors are not considered in our work. The second is that the agent lacks a backtracking mechanism. CityNavAgent so far only relies on the sub-goal decomposition to track the navigation path.

References

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

Dong An, Yuankai Qi, Yangguang Li, Yan Huang, Liang Wang, Tieniu Tan, and Jing Shao. 2023. Bevbert: Multimodal map pre-training for language-guided navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2737–2748.

Dong An, Hanqing Wang, Wenguan Wang, Zun Wang, Yan Huang, Keji He, and Liang Wang. 2024. Etpnav: Evolving topological planning for vision-language navigation in continuous environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

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.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 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. *arXiv preprint arXiv:1709.06158*.

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 for unified vision-and-language navigation. *arXiv preprint arXiv:2401.07314*.

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*, pages 11276–11286.

Peihao Chen, Xinyu Sun, Hongyan Zhi, Runhao Zeng, Thomas H Li, Gaowen Liu, Mingkui Tan, and Chuang Gan. 2023. A^2 nav: Action-aware zero-shot robot navigation by exploiting vision-and-language ability of foundation models. *arXiv preprint arXiv:2308.07997*.

Shizhe Chen, Pierre-Louis Guhur, Cordelia Schmid, and Ivan Laptev. 2021b. History aware multimodal transformer for vision-and-language navigation. *Advances in neural information processing systems*, 34:5834–5847.

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*, pages 16537–16547.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6.

Edsger W Dijkstra. 2022. A note on two problems in connexion with graphs. In *Edsger Wybe Dijkstra: His Life, Work, and Legacy*, pages 287–290.

Vishnu Sashank Dorbala, Gunnar Sigurdsson, Robinson Piramuthu, Jesse Thomason, and Gaurav S Sukhatme. 2022. Clip-nav: Using clip for zero-shot vision-and-language navigation. *arXiv preprint arXiv:2211.16649*.

Chen Gao, Xingyu Peng, Mi Yan, He Wang, Lirong Yang, Haibing Ren, Hongsheng Li, and Si Liu. 2023. Adaptive zone-aware hierarchical planner for vision-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14911–14920.

Peng Gao, Peng Wang, Feng Gao, Fei Wang, and Ruyue Yuan. 2024a. Vision-language navigation with embodied intelligence: A survey. *arXiv preprint arXiv:2402.14304*.

Yunpeng Gao, Zhigang Wang, Linglin Jing, Dong Wang, Xuelong Li, and Bin Zhao. 2024b. Aerial vision-and-language navigation via semantic-topo-metric representation guided llm reasoning. *arXiv preprint arXiv:2410.08500*.

Jing Gu, Eliana Stefani, Qi Wu, Jesse Thomason, and Xin Eric Wang. 2022. Vision-and-language navigation: A survey of tasks, methods, and future directions. *arXiv preprint arXiv:2203.12667*.

Pierre-Louis Guhur, Makarand Tapaswi, Shizhe Chen, Ivan Laptev, and Cordelia Schmid. 2021. Airbert: In-domain pretraining for vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1634–1643.

Yicong Hong, Cristian Rodriguez-Opazo, Qi Wu, and Stephen Gould. 2020. Sub-instruction aware vision-and-language navigation. *arXiv preprint arXiv:2004.02707*.

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

Jingyang Huo, Qiang Sun, Boyan Jiang, Haitao Lin, and Yanwei Fu. 2023. Geovln: Learning geometry-enhanced visual representation with slot attention for vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23212–23221.

Muhammad Zubair Irshad, Niluthpol Chowdhury Mithun, Zachary Seymour, Han-Pang Chiu, Supun Samarasekera, and Rakesh Kumar. 2022. Semantically-aware spatio-temporal reasoning agent for vision-and-language navigation in continuous environments. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 4065–4071. IEEE.

Vihan Jain, Gabriel Magalhaes, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. 2019. Stay on the path: Instruction fidelity in vision-and-language navigation. *arXiv preprint arXiv:1905.12255*.

Aishwarya Kamath, Peter Anderson, Su Wang, Jing Yu Koh, Alexander Ku, Austin Waters, Yinfei Yang, Jason Baldridge, and Zarana Parekh. 2023. A new path: Scaling vision-and-language navigation with synthetic instructions and imitation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10813–10823.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. 2023. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026.

Jing Yu Koh, Honglak Lee, Yinfei Yang, Jason Baldridge, and Peter Anderson. 2021. Pathdreamer: A world model for indoor navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14738–14748.

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

Jacob Krantz and Stefan Lee. 2022. Sim-2-sim transfer for vision-and-language navigation in continuous environments. In *European Conference on Computer Vision*, pages 588–603. Springer.

Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. 2020. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII 16*, 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*.

Shuhei Kurita and Kyunghyun Cho. 2020. Generative language-grounded policy in vision-and-language navigation with bayes' rule. *arXiv preprint arXiv:2009.07783*.

Jialu Li and Mohit Bansal. 2023. Improving vision-and-language navigation by generating future-view image semantics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10803–10812.

Xiangyang Li, Zihan Wang, Jiahao Yang, Yaowei Wang, and Shuqiang Jiang. 2023. Kerm: Knowledge enhanced reasoning for vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2583–2592.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. 2023a. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*.

Shubo Liu, Hongsheng Zhang, Yuankai Qi, Peng Wang, Yanning Zhang, and Qi Wu. 2023b. Aerialvln: Vision-and-language navigation for uavs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15384–15394.

Yang Liu, Xinshuai Song, Kaixuan Jiang, Weixing Chen, Jingzhou Luo, Guanbin Li, and Liang Lin. 2024. Multimodal embodied interactive agent for cafe scene. *arXiv preprint arXiv:2402.00290*.

Dipendra Misra, Andrew Bennett, Valts Blukis, Eyvind Niklasson, Max Shatkhin, and Yoav Artzi. 2018. Mapping instructions to actions in 3d environments with visual goal prediction. *arXiv preprint arXiv:1809.00786*.

Yuankai Qi, Zizheng Pan, Yicong Hong, Ming-Hsuan Yang, Anton Van Den Hengel, and Qi Wu. 2021. The road to know-where: An object-and-room informed sequential bert for indoor vision-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1655–1664.

Sonia Raychaudhuri, Saim Wani, Shivansh Patel, Unnat Jain, and Angel X Chang. 2021. Language-aligned waypoint (law) supervision for vision-and-language navigation in continuous environments. *arXiv preprint arXiv:2109.15207*.

Raphael Schumann and Stefan Riezler. 2020. Generating landmark navigation instructions from maps as a graph-to-text problem. *arXiv preprint arXiv:2012.15329*.

Raphael Schumann, Wanrong Zhu, Weixi Feng, Tsu-Jui Fu, Stefan Riezler, and William Yang Wang. 2024. Velma: Verbalization embodiment of llm agents for vision and language navigation in street view. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18924–18933.

Dhruv Shah, Błażej Osiński, Sergey Levine, et al. 2023. Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action. In *Conference on robot learning*, pages 492–504. PMLR.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2020. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10740–10749.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hanqing Wang, Wei Liang, Luc Van Gool, and Wenguan Wang. 2023a. Dreamwalker: Mental planning for continuous vision-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10873–10883.

Hu Wang, Qi Wu, and Chunhua Shen. 2020. Soft expert reward learning for vision-and-language navigation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IX 16*, pages 126–141. Springer.

Xin Wang, Wenhan Xiong, Hongmin Wang, and William Yang Wang. 2018. Look before you leap: Bridging model-free and model-based reinforcement learning for planned-ahead vision-and-language navigation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 37–53.

Zihan Wang, Xiangyang Li, Jiahao Yang, Yeqi Liu, Junjie Hu, Ming Jiang, and Shuqiang Jiang. 2024. Lookahead exploration with neural radiance representation for continuous vision-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13753–13762.

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

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

A Appendix

A.1 Prompt Engineering

In this part, we discuss the prompt used in this work. <> denotes the content needs to be filled in the prompting process.

LLM Prompt for High-level Planner The complete prompt is:

Task Planning Prompt

You are a navigation aircraft, and now you need to navigate to a specified location according to a natural language instruction. You need to extract a landmark sequence from the instruction. The sequence order should be consistent with their appearance on the path. Your output should be in JSON format and must contain two fields: "Landmark sequence" and "Thought." "Landmark sequence" is your thinking result comprised of landmark phrases in the instruction. "Thought" is your thinking process. The instruction is < "First, you need to find a stop sign. Then take left and right and continue until you reach a square with a tree. Continue first straight, then right, until you find a white truck. The final destination is a white building." >

The example output is:

"Landmark sequence": [
"stop sign",
"square with a tree",
"white truck",
"white building",
]
"Thought": "The instruction outlines a series of landmarks that need to be followed in order. The first landmark is a 'stop sign,' which I need to locate first. After finding the stop sign, I will navigate to a 'square with a tree' by taking left and right turns. Then, I will continue straight and turn right until I encounter a 'white truck.' Finally, my destination is a 'white building.' I have listed these landmarks in the order they appear in the navigation process."

The Prompt for Low-level Planner The complete prompt for the LLM commonsense reasoning in the low-level planner is:

Commonsense Reasoning Prompt

Based on the instruction, next navigation subgoal, and observation, list 3 objects you will probably go next from your OBSERVED OBJECTS in descending order of probability. *You are a drone and your task is navigating to the described target location!*

Navigation instruction: Start from the 'open' logo, fly forward and pass the yellow 'restaurant' logo in the front road. Then fly forward to the left, pass the circle grassland, and turn right. Finally, stop in front of the glass door with the trash bin and bench.

Your next navigation subgoal: yellow 'restaurant' logo

Objects or areas you observed: building with stairs, road, street lamp, street lamp, 'open' logo, building with stairs

Based on the instruction, next navigation subgoal, and observation, list 3 objects you will probably go next from your OBSERVED OBJECTS in descending order of probability.

A.2 Global Memory Module

A.2.1 Memory Graph Search Algorithm

The graph search problem is formulated as given a memory graph $G(N, E)$ and a sequence of landmark phrases $L = (\ell_1, \ell_2, \ldots, \ell_n)$ extracted from the language instructions, the goal is to determine a sequence of waypoints $W = (w_0, w_1, \ldots, w_m)$ that maximizes $P(r_L = 1|W, L)$, where $r_L = 1$ indicates that the sequence of the landmarks is traversed successfully. A scoring function $Q(i, w_k)$ is defined to represent the max probability of a path ending in w_k that visited the landmarks $(\ell_1, \ldots, \ell_i)$ and $P(r_L = 1|W, L) = Q(n, W)$. Then, a graph search method integrated with the Dijkstra algorithm (Dijkstra, 2022) is designed for calculating W^*.

Algorithm 1 Graph Search (Shah et al., 2023)

Input: Landmarks $(\ell_1, \ell_2, \ldots, \ell_n)$.
Input: Graph $G(N, E)$.
Input: Starting node S.
$\forall i = 0, \ldots, n, \forall w \in N, Q[i, w] \leftarrow -\infty$
$Q[0, S] \leftarrow 0$
DIJKSTRA_ALGORITHM$(G, Q[0, *])$

for $i \in 1, 2, \ldots, n$ **do**
  $\forall w \in W, Q[i, w] \leftarrow Q[i-1, w] +$ LLM(w, ℓ_i)
  DIJKSTRA_ALGORITHM$(G, Q[i, *])$
end for
$destination \leftarrow \arg\max(Q[n, *])$
return BACKTRACK$(destination, Q[n, *])$

A.2.2 Visualization of Memory Graph

The memory graph in different scenes is shown in Figure 6.

A.3 Point Clould Construction

As shown in Fig.2, with the camera intrinsic matrix K and agent's pose (p, α), pixels p_o of the depth image of observation view can be projected to a 3D point cloud in the world coordinate system as $P_w^{\alpha'} = R_{\alpha'} \cdot Z \cdot K^{-1} \cdot p_o + p$, where $R_{\alpha'}$ is the rotation matrix of observation view and Z denotes depth values of pixels. The final point cloud $\mathcal{M}_s$ is given by $\mathcal{M}_s = P_w^{\alpha-90^\circ} \odot P_w^{\alpha-45^\circ} \odot P_w^{\alpha} \odot P_w^{\alpha+45^\circ} \odot P_w^{\alpha+90^\circ}$. Note that RGB-D images in each observation view are well-aligned, meaning that the extracted semantic masks of RGB images have their counterparts in depth images as well

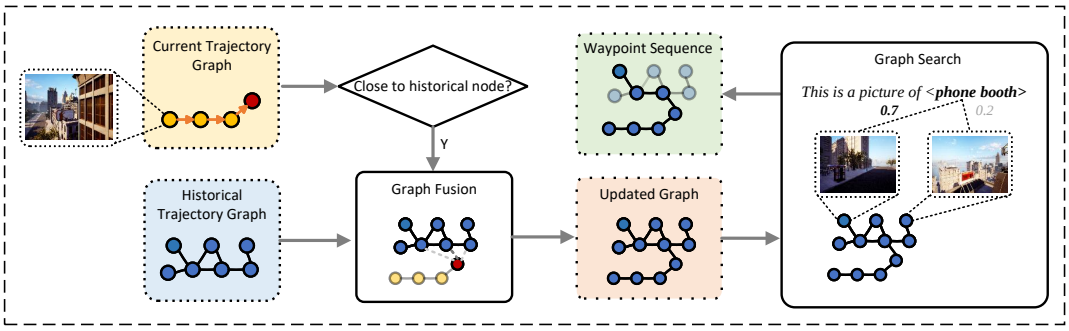

Figure 5: The illustration of the memory module. The agent stores its visual observation in the current trajectory graph. Once it reaches the node in the historical trajectory graph, the agent fuses these two graphs and searches for a path with the highest probability to the target. The probability is measured by a similarity score between the landmark phase in the instruction and visual observation stored in the node.

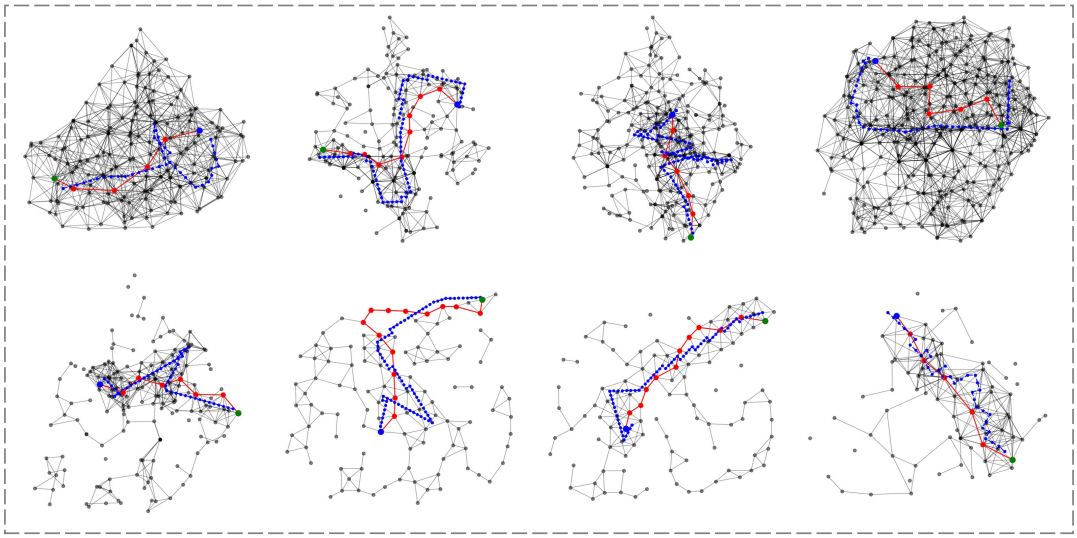

Figure 6: The memory graphs in different scenes. Each node in graphs stores the node's location and the agent's observation. The long distance between different historical trajectories results in the disconnection of memory graphs. *Blue* dot lines and *red* dot lines are ground truth and planned trajectories, respectively. The planned trajectories in the first row fail to follow the ground-truth trajectories while the last row have better instruction following the performance.

as in the point cloud. To this end, a local map containing both semantic and spatial information is constructed.

A.4 Details on Perception Module

For all experiments, we employ GPT-4V (Achiam et al., 2023) for object reasoning and landmark phase extraction. During the image grounding, a target is considered successfully detected if the bounding box's confidence score exceeds the threshold $\theta = 0.4$. The agent is equipped with an aligned RGB-D camera with 512x512 resolution and 90° field of view (FOV), capturing panoramic observations by rotating itself. The panoramic view directions are set at $p - 90^\circ$, $p - 45^\circ$, p°, $p + 45^\circ$, and $p + 90^\circ$ where p represents the agent's heading direction. The agent's low-level action space is (*"move forward"*, *"turn left"*, *"turn right"*, *"go up"*, *"go down"*, and *"stop"*). The moving step is 5 meters and each rotation turns the agent by 15°. The agent will receive its GPS location at each step.

A.5 Fine-grained AirVLN Dataset

We follow the similar idea of (Hong et al., 2020) to provide detailed descriptions of visual landmarks along the path and the agent's actions, making instructions more specific. Fig. 7 illustrated the sample of a fine-grained sample. The following are the comparison between a fine-grained instruction and an original AirVLN instruction:

Figure 7: The fine-grained AirVLN instruction and trajectory. The instructions and trajectories are well-aligned.

***Instruction in AirVLN**: "turning left and going straight to the buildings and slight right turn. coming near the lake and turning right and going up to the building. coming down to the building and again going up and going straight. going top of the building and turning left and coming down to the building. roaming around a tower and searching each floor of apartment."*
***Fine-grained instruction**:Start from the blue billboard with 'leartes bank', fly forward, pass the 'GAS' sign, and turn left. Continue flying forward, passing the yellow billboard and the 'Americar' billboard ahead. Continue flying forward and left until you reach the blue billboard with 'leartes bank' and stop."*

Dataset statistics The fine-grained samples are collected in 10 scenes from AirVLN. In each scene, 10 samples are collected. We further divide the fine-grained samples into three difficulty levels based on their trajectory length: easy tasks traverse two landmarks, normal tasks pass through three to four landmarks, and hard tasks involve navigating past five or more landmarks. The ratio of these three types of tasks is 1:3:1. The detailed statics are shown in Tab. 6.

A.6 More Experiment Results

A.6.1 Result on AirVLN-E of different difficulties

A.6.2 More Results of Reasoning Process

We illustrate the qualitative navigation process of CityNavAgent to further illustrate how the commonsense reasoning and memory graph work. As depicted in Figure 8, the agent is spawned at a random location with a navigation instruction. The agent has to explore the ordered landmarks in the instruction based on its visual observation. Thanks to its reasoning capabilities, the agent infers objects in its FOV that are semantically related to landmarks, even when those landmarks are not visually observed. In the given example, the agent tries to find the obelisk in the park which is currently invisible. Hinted by the instruction that the obelisk is in the park, CityNavAgent reasons that the trees probably appear in the park and decides to explore the areas near the trees while LM-Nav is gradually lost due to its lack of exploration ability (first three columns of Figure 8). Once the agent reaches a place visited before, CityNavAgent leverages the memory graph to search for a path to the target while VELMA only relies on LLM for action planning and is trapped in an unfamiliar place (see the last columns of Figure 8).

A.6.3 Failure Case Distribution

Table 5: Overall performance comparisons AirVLN-E.

Method	Easy			Normal			Hard			Mean		
	SR↑	SPL↑	NE↓	SR↑	SPL↑	NE↓	SR↑	SPL↑	NE↓	SR↑	SPL↑	NE↓
RS	0.0	0.0	85.6	0.0	0.0	127.9	0.0	0.0	164.9	0.0	0.0	129.6
AC	0.0	0.0	242.2	0.0	0.0	315.6	0.0	0.0	263.2	0.0	0.0	290.4
Seq2seq	0.0	0.0	201.1	0.0	0.0	359.4	0.0	0.0	713.3	0.0	0.0	398.5
CMA	0.0	0.0	152.1	0.0	0.0	317.6	0.0	0.0	286.8	0.0	0.0	278.3
NavGPT	0.0	0.0	79.8	0.0	0.0	126.1	0.0	0.0	177.6	0.0	0.0	127.2
MapGPT	0.0	0.0	97.5	5.6	2.9	135.2	0.0	0.0	165.5	3.3	1.5	133.7
VELMA	0.0	0.0	76.5	0.0	0.0	141.7	0.0	0.0	192.4	0.0	0.0	138.0
LM-Nav	15.4	13.7	123.1	22.2	18.1	124.3	33.3	**28.1**	**114.2**	23.6	19.2	119.4
CityNavAgent	**25.0**	**21.3**	**74.7**	**27.8**	**23.3**	**93.4**	**33.3**	26.3	121.5	**28.3**	**23.5**	**95.1**

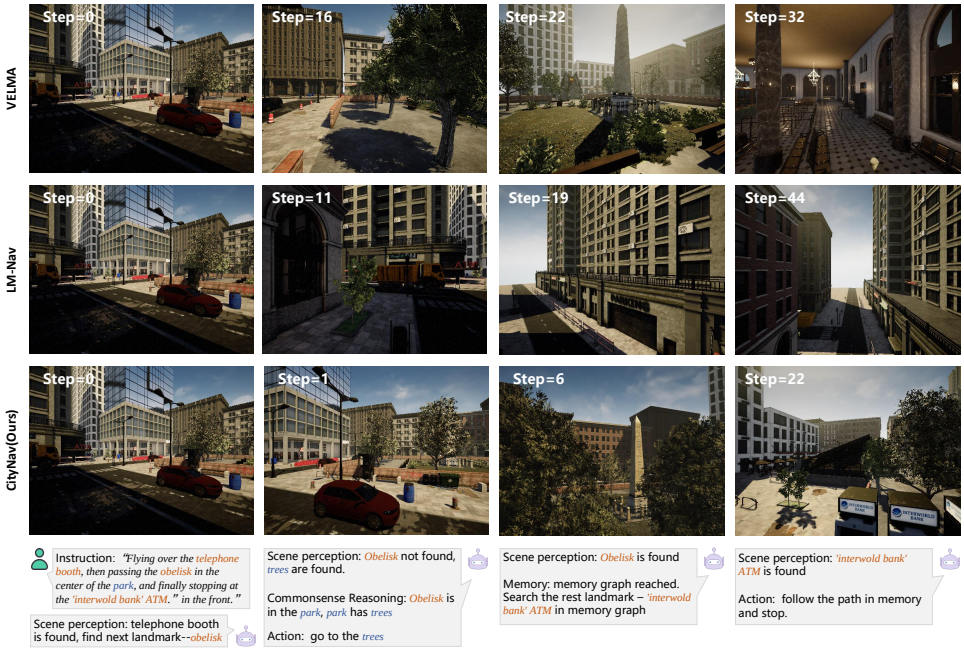

Figure 8: Qualitative result of the navigation process. The first three rows are the first-person view of VELMA, LM-Nav and CityNavAgent during the navigation. The last row is the reasoning process of CityNavAgent. *Orange* and *blue* represents the navigable landmarks in the instruction and common objects that semantically relevant to the landmarks, respectively.

Table 6: Comparision of fine-grained dataset with AirVLN.

Dataset	Routes	Vocab	Instr. Len.	# of Landmark	Traj. Len.	Traj. Len. (easy)	Traj. Len. (normal)	Traj. Len. (Hard)
AirVLN	3,916	2.8k	82	-	321.3	-	-	-
Refined	101	0.4k	39	4.1	156.1	104.3	147.9	235.8

Table 7: The distribution of failure cases.

Datasets	Instruction Ambiguity	Perception Failure	Reasoning Failure
AirVLN-S	45.4	34.1	20.5
AirVLN-E	5.8	76.8	17.4