Embodied Zero-shot Vision-and-Language Navigation in City Environment with LLM-empowered Agent

Weichen Zhang[1,2] **Chen Gao**[1] **Shiquan Yu**[3] **Baining Zhao**[1,2] **Han Li**[1]
Qian Zhang[1] **Susu Xu**[4] **Jinqiang Cui**[2] **Xinlei Chen**[1,2*] **Yong Li**[1]
[1]Tsinghua University, [2]Pengcheng Laboratory, [3]University of Oxford, [4]Johns Hopkins University
{zhangwc23,zbn22,h-li23, zhangq22}@mails.tsinghua.edu.cn
{chgao96, liyong07}@tsinghua.edu.cn, chen.xinlei@sz.tsinghua.edu.cn
shiquan.yu@lmh.ox.ac.uk, sxu83@jhu.edu

Abstract

A fundamental objective within the field of AI research is the development of intelligent agents capable of engaging in natural language interactions with humans, comprehending their surroundings, and executing tasks in real-world settings. One of the most representative tasks, named vision-and-language navigation (VLN), requires the agent to navigate to a specific location following human language instruction. Although existing zero-shot agents for VLN achieved notable results in indoor settings, they mostly rely on navigation graphs (nav-graphs for short) to select the next potential waypoint, which is generally impractical in 3D continuous spaces in urban environments. In this work, we propose a novel city-level VLN agent, CityNav, that introduces semantic map-based spatial reasoning for next-waypoint prediction and a topological memory graph to guarantee navigation stability in 3D urban environments in the absence of nav-graphs. Specifically, CityNav first builds a 3D local semantic map and relies on LLM agents' common knowledge to reason for the object in the map that is mostly relevant to the navigation instruction or the target; the agent then obtains navigable locations within the space and the record them in a topological memory graph; last, for the navigation target already stored in the memory graph, we can directly search for the shortest path to the target in the graph which can largely reduce the cost in exploration. Extensive benchmark experiments show that our method improves significantly over baselines ($\sim$5% in SR) and achieves state-of-the-art performance. Further experiments demonstrate the effectiveness of LLM-based commonsense reasoning and memory graph exploitation for zero-shot VLN in city environments.

1 Introduction

Visual-and-language navigation is a fundamental task wherein the agent navigates to a landmark or a spot in urban environments following language instructions [1, 2, 3]. Such a task is easy for humans but challenging for AI agents, especially unmanned aerial vehicles and agents navigating within open and dynamic scenarios such as city environments. The task requires the agent to have strong capacities in both vision and language understanding, along with the ability to execute complicated actions in interactive environments. As a task that measures the intelligence level of human-like perception and decision-making in embodied environments [4, 5], VLN has attracted much attention in embodied artificial intelligence. However, most of the existing works on VLN [6, 7, 8, 9, 10] are deployed on discrete indoor settings [11, 12, 1, 13] in which the environment is limited to a small scale. That is, agents only need to move between traversable nodes of the navi-graph, which has a small and fixed topology of the environment. For example, given the task instruction of "navigate to a red sofa", the agent relies on the *navi-graph* to select the next node to reach the destination based on

Preprint. Under review.

its first-view observed environment context. The existing methods for indoor environments [14, 15] consider VLN a graph-search task with learning-based methods. However, these approaches cannot well handle open environments such as cities since both nav-graph and the environment context are far more complex [16].

That is to say: for urban environments, since the nav-graph cannot be predefined, the VLN tasks turn to a **zero-shot** setting, which is crucial and quite challenging and is seldom explored. The challenges are mainly due to agents' high degree of freedom (DOF) and diverse scenarios in city environments, summarized as follows.

- **Zero-shot short-term action reasoning in three-dimensional continuous contextual environments.** Zero-shot short-term action reasoning requires the agent to predict the next step action without pre-training, which is essential for the agent to navigate in new environments. Previous works [17, 8, 18] predict the agent's next action in a rather limited pre-defined action space (such as the topological map) or two-dimensional environment. However, when it comes to three-dimensional continuous environments, the action space grows exponentially, which imposes a great challenge for the agent to search for the correct action sequence.
- **Long-term waypoint planning in large-scale urban scenarios** VLN tasks in the city environment require the agent to travel across multiple streets or blocks [17], leading to very long path lengths, which are usually ten times longer than the indoor setting. As a result, the agent involves a complex chain-like decision process to predict a long waypoint sequence to reach the target location. However, due to the diverse urban scenarios, any errors in the decision process will quickly accumulate and cause the agent to fall into a dead end or get lost in an unexplored position.

In this work, to address the above challenges, we propose an embodied aerial agent called CityNav, comprising semantic map-based spatial reasoning and topological memory graph searching for short-term and long-term waypoint planning. To deal with the first challenge, we build a local 3D semantic map to represent fine-grained spatial information of the surroundings with small storage consumption. We then decompose the spatial reasoning into commonsense reasoning by LLM and waypoint calculation by semantic mapping. Based on the waypoint, the short-term motion selection by the low-level controller can be well applied. For the second challenge, we design a topological memory graph in which the nodes store the information of visited waypoints in historical trajectories, and the edges store the distance and connectivity between nodes. Once the agent reaches a graph node, it will search for a path on the graph that reaches the most likely target node, based on which the long-term planning can be well handled. We evaluate the proposed CityNav with extensive benchmark experiments[19], of which the results show CityNav's effectiveness in city-environment VLN tasks with different task levels. Compared to the state-of-the-art approaches [17], CityNav exhibits superior performance in both success rate (SR) and success weighted by path length (SPL), outperforming it by $19-30\%$ and $10-20\%$. The main contributions of this work are as below:

- To the best of our knowledge, we take the pioneering step to investigate the important VLN task in a zero-shot setting, a fundamental task for embodied intelligence agents in open urban environments.
- We design an innovative embodied aerial agent, CityNav, which well combines the LLM's commonsense reasoning capabilities on the semantic map and historical navigational experiences with open-world short-term and long-term waypoint planning. We decompose complex spatial reasoning into commonsense reasoning and waypoint prediction with LLM and semantic map, which is more feasible for low-level motion planning.
- We conduct extensive experiments on benchmark evaluations, and the results show that our method achieves SOTA performance across multiple urban scenarios, demonstrating promising improvements over existing baselines.

2 Problem Formulation

Given a natural language instruction $\mathcal{I}$ and the agent's egocentric observation $\mathcal{O}$, the goal of zero-shot city-level VLN is to determine a sequence of action for the agent to reach the target location p_d in a *training-free manner*. At each time 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, which is given by:

$$a_t = \pi(o_t, \mathcal{I}). \tag{1}$$

The location of the agent p_t will be updated by a_t and its kinematic model $\mathcal{F}$, which is given by:

$$p_t = \mathcal{F}(p_{t-1}, a_t). \tag{2}$$

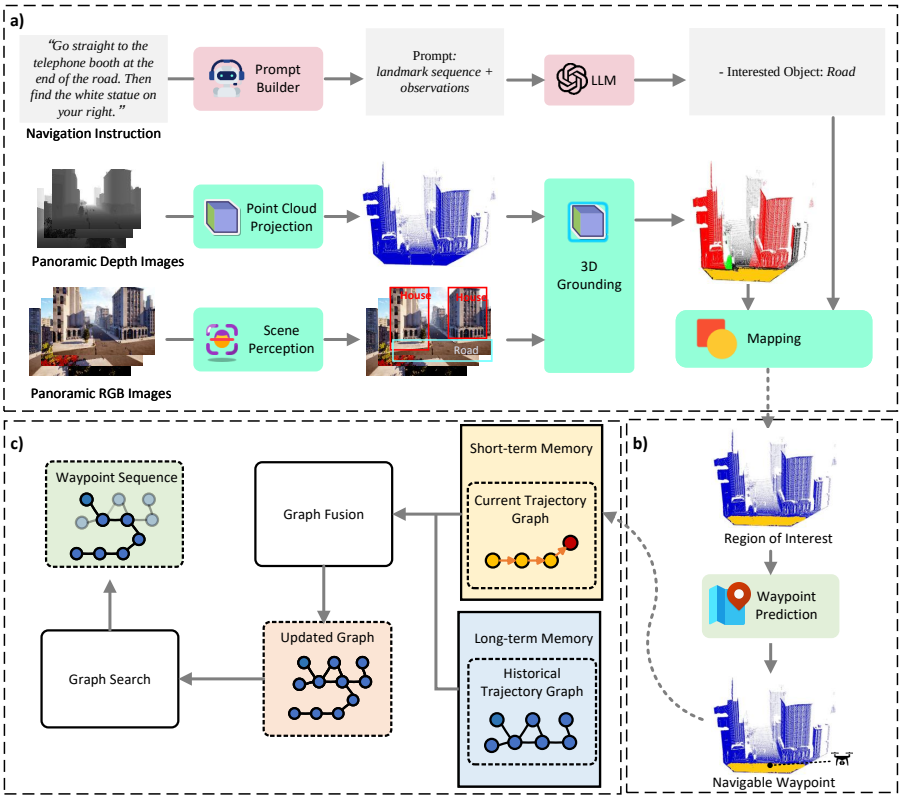

Figure 1: **The overall workflow of the proposed CityNav.** a) The agent builds a local semantic map based on RGB-D observations. Conditioned on the navigation instruction and semantic information, the agent performs commonsense reasoning via an LLM to select an ROI in the map. b) Based on the ROI, the agent computes a navigable waypoint and moves to it by a low-level motion planner. c) Once the agent reaches the memory graph, it merges its history observation into the graph and tries to search for a path to the target in the graph.

When given a sequence of time steps $T = (t_0, \ldots, t_n)$, the agent's final location p_n is obtained and the probability of reaching the target $P_{success}$ is given by

$$P_{success} = P(||p_n - p_d|| < \epsilon) = P(||\mathcal{F}(\pi(\mathcal{O}, \mathcal{I}), p_0) - p_d|| < \epsilon), \tag{3}$$

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

$$\pi^* = \text{argmax}_\pi P(||\mathcal{F}(\pi(\mathcal{O}, \mathcal{I}), p_0) - p_d|| < \epsilon), \tag{4}$$

3 The Proposed Approach: CityNav

In this section, we present the workflow of the proposed CityNav for zero-shot VLN in urban environments. As shown in Figure 1, the CityNav framework first converts the input panoramic RGB-D images into a local semantic map. Then, it leverages LLM's commonsense reasoning capacity to reason for the spatial relation between landmarks in the navigation instruction and common objects to predict the next waypoint for short-term planning (Section 3.1). Further, if the agent reaches a visited area represented by a topological memory graph, it tries to find a path from the graph with the highest probability to the target described in the navigation instruction by a graph search algorithm (Section 3.2).

3.1 Semantic Map-based Commonsense Reasoning for Waypoint Prediction

For zero-shot VLN, the agent is required to be capable of spatial reasoning, *i.e.*, gathering spatial information perception from the surroundings and reasoning for the agent locomotion. Existing

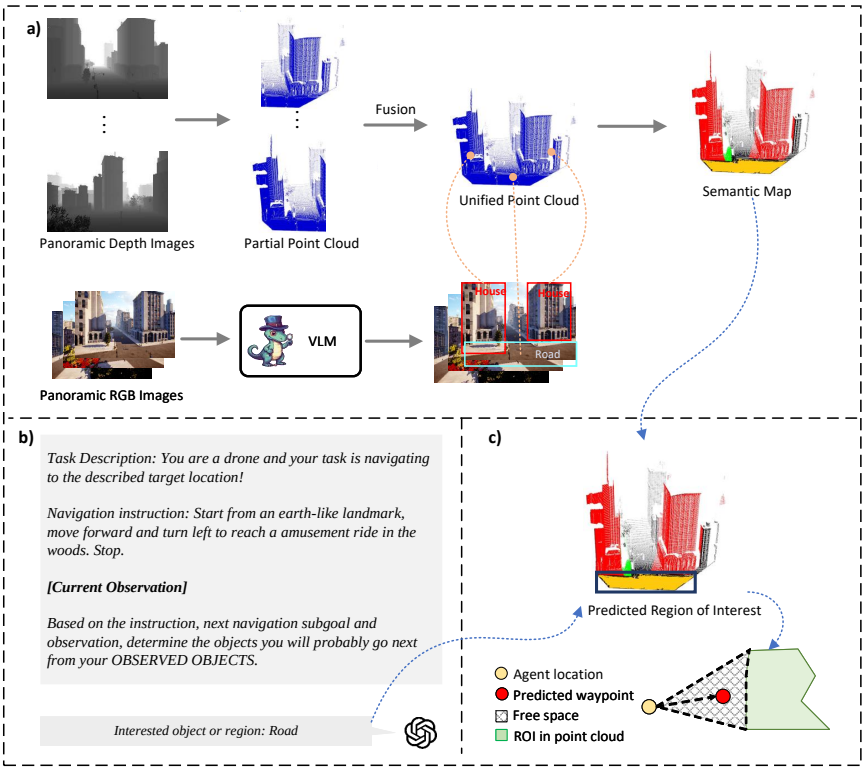

Figure 2: **Semantic map-based commonsense reasoning for waypoint prediction.** a) The agent builds a unified point cloud via panoramic depth images. Meanwhile, it detects the scene objects via a VLM to label the point cloud. b) The prompt template for LLM's commonsense reasoning. The text of current observations will be updated at each time step. c) After obtaining the ROI, the waypoint is computed by averaging the coordinates of free space between the agent and the ROI.

work [17, 8, 20] follows the paradigm of considering the LLM as the agent's motion planner in navi-graphs. However, when it comes to continuous space without the navi-graph in zero-shot city-level VLN tasks, the agent has to predict the precise coordinate of the next waypoint by itself. In this case, we introduce a semantic map for fine-grained spatial representation of the surroundings and decompose the spatial reasoning into LLM-based commonsense reasoning, semantic map-based waypoint prediction, and low-level motion planning. The agent first builds a local semantic map based on the RGB-D observations with small storage consumption. Then, with a carefully designed prompt input, the LLM utilizes commonsense knowledge to reason for the region in the map that has the most semantic relation to the target or landmarks in navigation instruction. Based on the region of interest (ROI) and semantic map, the agent computes a waypoint in the free space between the agent's location and the ROI. Lastly, the precise coordinate of the waypoint is passed to the low-level path planner for motion control.

Semantic Map Construction. To build a local semantic map, the first thing is to extract the objects in visual observations. We define a set of common objects in the urban environment and utilize a large multimodal model (LMM) named GroundingDino [21] to detect all these objects in the agent's panoramic RGB images. For each object, the LMM outputs a bounding box and a corresponding confidence score. Only those bounding boxes with a confidence score higher than the threshold are preserved and the rest of the areas are considered part of the background lacking semantic information. As a result, each pixel could be assigned with a semantic label $c \in C$. With well-aligned RGB-D images and camera intrinsic matrix, each pixel in RGB images can be projected into a 3D point cloud in the world coordinate system as $P_w^{(i)(c)} = R_w \cdot Z \cdot K^{-1} \cdot p_o^{(i)(c)} + T_w$, where $p_o^{(i)(j)}$ is the homogeneous coordinate of the pixel with label c in i-th view of n discrete panoramas, $P_w^{(i)(c)}$ is the counterpart in 3D world coordinate, K is the camera intrinsic matrix, and R_w, T_w are the pose of the

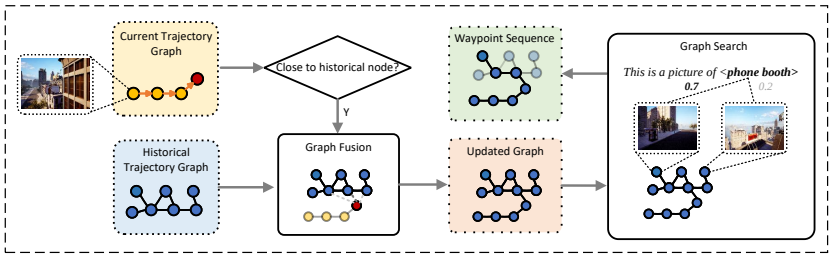

Figure 3: **Memory graph-based exploitation.** The agent stores its visual observation in the current trajectory graph. Once it reaches the node in the historical trajectory graph, the agent fuses these two graphs and searches for a path with the highest probability to the target. The probability is measured by a similarity score between the landmark phase in the instruction and visual observation stored in the node.

agent. After merging all point clouds across views, the final semantic map $\mathcal{M}_s$ is given by:

$$\mathcal{M}_s = R_e^{(1)} \cdot P_w^{(c)} \odot R_e^{(2)} \cdot P_w^{(c)} \cdots \odot R_e^{(n)} \cdot P_w^{(c)}, \tag{5}$$

where $\odot$ is the concatenation operation, and $R_e^{(i)}$ is the rotation matrix between i-th view of panoramas and the agent pose.

Commonsense Reasoning for Region of Interest. Based on the constructed semantic map, a prompt (in Fig. 3) that comprises the navigation instruction and observed objects is updated by the scene perception result. Then, it is utilized by LLM to reason for the map region most semantically relevant to the target. Induced by the prompt, LLM leverages commonsense knowledge to reason for region of interest (ROI) most pertinent to the landmarks or the target specified in the instruction. With the predicted ROI, the waypoint can be computed by the coordinates of the corresponding point cloud.

Semantic Map-based Waypoint Prediction. As shown in Figure 3(b), based on the reasoned ROI, the precise waypoint in the map can be computed. Instead of directly averaging the coordinates of all points in ROI, we select the central point of the free space between agent and ROI to avoid collision.

Low-level Motion Planning. This part is to plan a sequence of basic actions of the agent from its action space. The agent's action space is (*"move forward"*, *"turn left"*, *"turn right"*, *"go up"*, and *"go down"*). The moving step is 5 meters, and each rotation turns the agent by $15°$. Given the predicted coordinate of the waypoint, the motion planner will output an action sequence by the greedy algorithm for the agent to reach the waypoint.

3.2 Memory Graph-based Exploitation

Although the semantic map-based waypoint prediction with commonsense reasoning can deal with short-term planning in continuous space, it cannot guarantee long-term waypoint planning due to the complex open-world environment and fast accumulation of navigation errors. To counteract this, we utilize the agent's historical navigation paths as a priori knowledge for current navigation tasks, enabling the agent to leverage past experiences to select paths more likely to the target when undertaking similar tasks.

Memory Graph Construction At each time step t, the agent arrives at a waypoint w_t and captures panoramic observations o_t. An undirected topological graph $G(N, E)$ is employed to store the waypoints. Each graph node $n \in N$ encapsulates both the coordinates of the waypoints and their associated panoramic observations. An edge $e \in E$ is established between two nodes only when their respective waypoints are adjacent. Then, an edge's weight is determined based on the spatial distance between the connected waypoints. Employing this methodology, each historical navigation path can be represented as a sub-topological graph and merged into the original memory graph.

Memory Graph Merge Once the agent completes a navigation task, a new subgraph is obtained using the method described above. This new subgraph needs to be integrated into the original memory graph for an update. The merge process consists of two parts: node update and edge update. For the node update, we directly merge the set of new subgraph's nodes N_x into the set of memory graph's node N. Similarly, for the edge update, edge sets from two graphs are directly merged. Moreover, it is necessary to determine whether the new subgraph and the original memory graph are connected; that is, if there exist new edges between the two graphs. To this end, we calculate the distance between every pair of points in N_x and N. If the distance between any pair of points is less than a threshold

H=15m, it is inferred that these points are adjacent, prompting the addition of a new edge to the memory graph.

Graph Search for Navigation: When confronted with a new navigation task, the agent could follow the landmarks mentioned in the navigation instructions to reach the destination. Consequently, when reaching a waypoint in the memory graph, the agent determines a path in the graph with the highest likelihood of traversing all landmarks. Similar to the methodologies of LM-Nav [22], 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 [23] is designed for calculating W^*. We simply revisit the graph search method in Alg. 1.

Algorithm 1 Graph Search [22]

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**
    $\forall w \in W, Q[i, w] \leftarrow Q[i-1, w] + \text{LLM}(w, \ell_i)$
    DIJKSTRA_ALGORITHM$(G, Q[i, *])$
end for
$destination \leftarrow \arg\max(Q[n, *])$
return BACKTRACK$(destination, Q[n, *])$

4 Experiments

4.1 Experimental Setup

Datasets We evaluated CityNav's performance on a simulator with city environments: AirVLN [19]. AirVLN consists of 25 different city-level environments and more than 8,000 VLN tasks. Since AirVLN only provides relatively coarse navigation instruction, we further sample some of the tasks and enrich their navigation instructions as our test set. A VLN task includes the start position, target position, fine-grained navigation instruction, and a reference path. The VLN tasks are categorized into three levels of difficulty based on their length: easy tasks traverse two landmarks, normal tasks pass through three to four landmarks, and hard tasks involve navigating past five or more landmarks.

Metrics We use 3 standard metrics [7] to measure navigation performance. Navigation Error (NE): average distance in meters between the agent's stop location and the target. Success Rate (SR): success ratio of the navigation task. The task is considered a success if NE is within 20 meters. Success weighted by Path Length (SPL): SR normalized by path length, given by $\frac{1}{N}\sum_i^N S_i \frac{l_i}{\max(p_i, l_i)}$ where S_i is a binary success indicator, p_i is the path length taken by the agent, l_i is the shortest path of the task and N is the total count of test tasks.

Baselines We compare CityNav against four outdoor zero-shot VLN baselines:

- **Random:** The agent randomly selects an action at each step. It stops when the 'stop' action is selected or it reaches the maximum step size.
- **Action Sample (AC):** The agent predicts the next action by sampling actions according to the action distribution of the AirVLN dataset.
- **VELMA:** An embodied VLN agent for discrete street-view environments that leverages LLM to predict agent next action directly. We adapt the action space of VELMA to the aerial agent for a fair comparison.
- **LM-Nav:** A ground agent for outdoor VLN, which relies entirely on a viewpoint memory graph and formulates the navigation task as a graph search problem for path planning.

It is worth mentioning that existing SOTA indoor VLN methods [24, 8, 20] such as NavGPT [8] and MapGPT [24] that rely on the pre-defined navi-graph directly predict the waypoint on the graph. As such, these methods cannot be directly evaluated in our continuous environment as baselines because navi-graphs are not available in our scenarios.

Table 1: Overall performance comparisons under different task difficulties.

Method	Easy			Normal			Hard			Mean		
	SR/% ↑	SPL/% ↑	NE/m ↓	SR/% ↑	SPL/% ↑	NE/m ↓	SR/% ↑	SPL/% ↑	NE/m ↓	SR/% ↑	SPL/% ↑	NE/m ↓
Random	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
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
CityNav(Ours)	**25.0**	**21.3**	**74.7**	**27.8**	**23.3**	**93.4**	**33.3**	26.3	121.5	**28.3**	**23.5**	**95.1**

4.2 Result and Analysis

4.2.1 Comparison with Baselines (RQ1)

To fully evaluate the capacity of exploration and exploration of the agent, the navigation task comprises the first unseen path and the following seen path in the memory. The agent spawned in a random place needs first to explore the path to the memory graph and navigate to the target by the memory graph. In Table 1, we compare the overall performance of CityNav with SOTA zero-shot outdoor navigation methods, from which we make the following observations.

- **The vast action space exhibits naive sampling-based methods.** Both the SR and SPL of random and action sample methods are close to 0, and the NE of these two methods are significantly high. This result illustrates that the long-term urban VLN involves an extremely large action space. It is possible for naive sampling methods to merely search for a correct action sequence without the understanding of the instruction and the surrounding information.
- **LLM fails to be a motion planner in continuous space.** VELMA that leverages LLM for scene understanding and motion planning also has 0 SR and high NE. We attribute this result to the fact that LLM lacks the ability of fine-grained spatial reasoning and predicts coarse-grained actions such as "turn left 90 degrees". Thus, in continuous space, the coarse-grained actions will cause errors that accumulate along the path, leading to great deviation from the target.
- **The agent is not more vulnerable to longer navigation tasks.** Surprisingly, both LM-Nav and CityNav achieve the highest SR and SPL on hard tasks. We speculate that the reason for this result is that longer paths provide more landmarks and task descriptions, making the paths more identifiable to the agent.

Our CityNav method significantly outperforms previous SOTAs on all metrics. Specifically, CityNav improves SR by approximately 28.3% and 4.7% over VELMA and LM-Nav, respectively. This demonstrates the critical importance and navigation efficiency of semantic map-based waypoint planning for continuous outdoor navigation. Furthermore, in terms of SPL, our approach achieves improvements of 23.5% and 2.5% compared to VELMA and LM-Nav, respectively, indicating that CityNav predicts navigation paths that are closer to the ground truth. The lowest error rate shows that even for those failed cases, our method still stops relatively close to the target. For easy and normal tasks, our method consistently surpasses the baseline by at least 5% in SR and SPL. For hard tasks, our method still has a similar performance to the best method.

4.2.2 Ablation Study (RQ2)

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 2, 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 CityNav (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 memory module during the navigation and replace the graph search algorithm with the random walk strategy. Presented in the first row in Table 2, the lack of memory graph results in a 16.6%, 14.4% decrease in SR, SPL, and a 116.7% increase in NE over CityNav. 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 scenarios, thereby ensuring the stability of navigation performance.

Table 2: Effectiveness of different modules in CityNav.

Semantic Map	Memory Graph	LLM		SR/% ↑	SPL/% ↑	NE/m ↓
		GPT-3.5	GPT-4			
✓			✓	11.7	9.1	206.1
	✓		✓	23.6	19.2	119.4
✓	✓	✓		23.3	16.1	98.9
✓	✓		✓	28.3	23.5	95.1

Effect of different LLMs. We also evaluate the effectiveness of different LLMs for commonsense reasoning in the exploration phase (Section 3.1). Although CityNav with GPT-3.5 demonstrates a SOTA performance, replacing GPT-3.5 with GPT-4 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-4 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.

4.2.3 Case Analysis (RQ3)

In this part, we illustrate the quantitative and qualitative navigation process of CityNav to further illustrate how the commonsense reasoning and memory graph work. As shown in Fig. 4, the NE of CityNav decreases faster than baselines and stops closest to the target. VELMA also gets close to the target quickly, but it stops at a distant location. LM-Nav first moves far from the target due to its random exploration process and gets closer to the target quickly and accurately once it reaches the memory graph. As depicted in Fig. 5, 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, CityNav 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 Fig. 5). Once the agent reaches a place visited before, CityNav 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 Fig. 5).

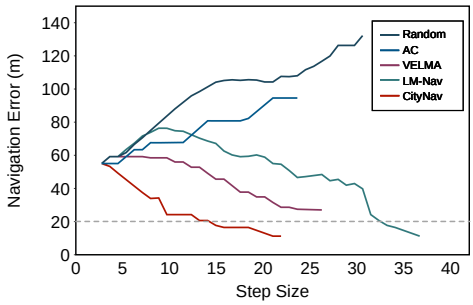

Figure 4: The navigation processes.

5 Related Works

Visual Language Navigation (VLN) Task The VLN tasks necessitate agents to interpret or follow provided instructions in unseen environments to desired objects or destinations [10, 25, 2, 3]. As interest in Visual Language Navigation (VLN) grows, an increasing array of datasets and tasks [26, 1, 27, 28, 29] have emerged. Based on the presence of a preset navigation graph, VLN tasks can be categorized into VLN in discrete environments (VLNDE) [1, 30, 13, 31, 32] and VLN in continuous environments (VLNCE) [14, 15]. R2R [1] introduced the first indoor VLNDE with navigational graphs enabling agents to teleport between adjacent nodes without motion planning. Later, R4R [30] further expanded the R2R tasks by connecting its path to longer trajectories. RxR [31] expanded the tasks in terms of language types, task scale, and quality of path-instruction pairs. Outdoor VLNDE is proposed by Touchdown [33] which requires the agent to follow the instructions to find hidden targets in Google Street View. However, those VLNDE tasks imply the following assumptions: known topology of environments, perfect motion control, and accurate localization, which yields the transfer of the agent from simulators to real scenes. To address these limitations, Krantz et al. [14] introduced R2R-CE tasks that adapt R2R trajectories for continuous environments. Irshad et al. [15] developed the RoboVLN dataset, which provides long-distance navigation in a continuous action space, making VLNCE tasks more practical and challenging.

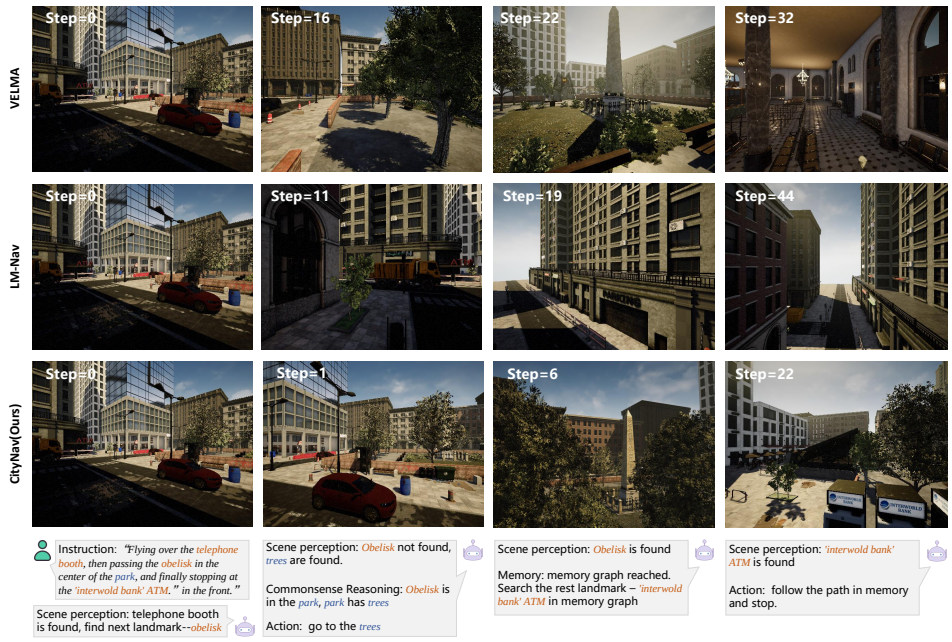

Figure 5: Qualitative result of the navigation process. The first three rows are the first-person view of VELMA, LM-Nav and CityNav during the navigation. The last row is the reasoning process of CityNav. *Orange* and *blue* represents the navigable landmarks in the instruction and common objects that semantically relevant to the landmarks, respectively.

Large Pre-trained Model for Visual-language Navigation Existing VLN methods can be summarized in two categories. One is end-to-end learning-based approaches [34, 35, 36, 37] that directly output an agent's next action with inputting step observation and language instruction. We focus on another category of VLN methods that leverage a large pre-trained model [38, 39, 40, 41] for zero-shot VLN. Pioneering works [10] use foundation models for scene understanding and instruction interpretation. NavGPT [8] utilizes LLM's commonsense reasoning ability to select the next viewpoint in the navi-graph. MapGPT [24] stores a historical topology map and prompts LLM to reason for the next map node to explore. However, these methods only work in discrete environments. A^2Nav [42] combines LLM with a small policy model for VLNCE. It uses GPT-3 [41] to decompose the task into subgoals and trains a policy network for action planning for each subgoal. As for outdoor VLN, VELMA [17] designs a verbalizer that converts the perception results of CLIP [43] into plain text for LLM to predict the next route point from discrete street panoramas. LM-Nav [22] is an outdoor VLNCE method that is mostly relevant to our work. However, it requires the agent to have the full topology map of the environment, which is not always practical. In this work, we propose CityNav, an agent for outdoor VLNCE without the reliance on a topology map.

6 Conclusion, Limitation, and Future Work

In this paper, we approach the problem of zero-shot vision-language navigation by proposing an embodied aerial agent, CityNav, which leverages the pre-trained knowledge in large models in three-dimensional urban spaces. The agent is composed of two modules: the semantic map-based exploration module and the memory graph-based exploitation module. The former enables the agent to explore unseen targets or landmarks with the object reasoning ability of LLM. The latter exploits historical experience to reduce the risk of long-distance exploration and improve navigation stability. The experimental results illustrate the efficacy and robustness of our method from different perspectives. One limitation of our work is that the whole system has not been deployed on a real drone. The second is that the agent lacks a backtracking mechanism. In the future, the lightweight implementation of LLM-empowered agent, along with fine-grained multi-modal spatial reasoning, will be explored to address the real-world deployment problem.

References

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

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

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

[4] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 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, 2020.

[5] Hao Wen, Yuanchun Li, Guohong Liu, Shanhui Zhao, Tao Yu, Toby Jia-Jun Li, Shiqi Jiang, Yunhao Liu, Yaqin Zhang, and Yunxin Liu. Empowering llm to use smartphone for intelligent task automation. *arXiv preprint arXiv:2308.15272*, 2023.

[6] Samir Yitzhak Gadre, Mitchell Wortsman, Gabriel Ilharco, Ludwig Schmidt, and Shuran Song. Cows on pasture: Baselines and benchmarks for language-driven zero-shot object navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23171–23181, 2023.

[7] Kaiwen Zhou, Kaizhi Zheng, Connor Pryor, Yilin Shen, Hongxia Jin, Lise Getoor, and Xin Eric Wang. Esc: Exploration with soft commonsense constraints for zero-shot object navigation. In *International Conference on Machine Learning*, pages 42829–42842. PMLR, 2023.

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

[9] Shizhe Chen, Pierre-Louis Guhur, Makarand Tapaswi, Cordelia Schmid, and Ivan Laptev. 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, 2022.

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

[11] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *arXiv preprint arXiv:1709.06158*, 2017.

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

[13] An Yan, Xin Eric Wang, Jiangtao Feng, Lei Li, and William Yang Wang. Cross-lingual vision-language navigation. *arXiv preprint arXiv:1910.11301*, 2019.

[14] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. 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, 2020.

[15] Muhammad Zubair Irshad, Chih-Yao Ma, and Zsolt Kira. Hierarchical cross-modal agent for robotics vision-and-language navigation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13238–13246. IEEE, 2021.

[16] Theophile Gervet, Soumith Chintala, Dhruv Batra, Jitendra Malik, and Devendra Singh Chaplot. Navigating to objects in the real world. *Science Robotics*, 8(79):eadf6991, 2023.

[17] Raphael Schumann, Wanrong Zhu, Weixi Feng, Tsu-Jui Fu, Stefan Riezler, and William Yang Wang. 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, 2024.

[18] Mohamed Aghzal, Erion Plaku, and Ziyu Yao. Can large language models be good path planners? a benchmark and investigation on spatial-temporal reasoning. *arXiv preprint arXiv:2310.03249*, 2023.

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

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

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

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

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

[24] Jiaqi Chen, Bingqian Lin, Ran Xu, Zhenhua Chai, Xiaodan Liang, and Kwan-Yee K Wong. Mapgpt: Map-guided prompting for unified vision-and-language navigation. *arXiv preprint arXiv:2401.07314*, 2024.

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

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

[27] Keji He, Yan Huang, Qi Wu, Jianhua Yang, Dong An, Shuanglin Sima, and Liang Wang. Landmark-rxr: Solving vision-and-language navigation with fine-grained alignment supervision. *Advances in Neural Information Processing Systems*, 34:652–663, 2021.

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

[29] Fengda Zhu, Xiwen Liang, Yi Zhu, Qizhi Yu, Xiaojun Chang, and Xiaodan Liang. Soon: Scenario oriented object navigation with graph-based exploration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12689–12699, 2021.

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

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

[32] Piotr Mirowski, Andras Banki-Horvath, Keith Anderson, Denis Teplyashin, Karl Moritz Hermann, Mateusz Malinowski, Matthew Koichi Grimes, Karen Simonyan, Koray Kavukcuoglu, Andrew Zisserman, et al. The streetlearn environment and dataset. *arXiv preprint arXiv:1903.01292*, 2019.

[33] Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. 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, 2019.

[34] Fengda Zhu, Yi Zhu, Xiaojun Chang, and Xiaodan Liang. Vision-language navigation with self-supervised auxiliary reasoning tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10012–10022, 2020.

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

[36] Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. Towards learning a generic agent for vision-and-language navigation via pre-training. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13137–13146, 2020.

[37] Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6629–6638, 2019.

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

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

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

[41] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

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

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

[44] Tong Qin, Peiliang Li, and Shaojie Shen. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics*, 34(4):1004–1020, 2018.

A Appendix

A.1 Implementation Details

For all experiments, we employ GPT-4 [40] for object reasoning and landmark phase extraction. We utilize GroundingDino [21] image grounding. 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 640x480 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 moving direction. The agent's action space is (*"move forward"*, *"turn left"*, *"turn right"*, *"go up"*, and *"go down"*). The moving step is 5 meters and each rotation turns the agent by 15°. The agent will receive its GPS location at each step.

Before the evaluation, the agent initializes a memory graph of the current city scene. The agent is randomly spawned at an arbitrary location and cruises in the city by manual control. The agent records its visual observations and GPS coordinates every 5 meters. Note that GPS information is unnecessary since the off-the-shelf outdoor SLAM algorithm such as VINS [44] could be adopted to obtain the agent's accurate poses. Subsequently, a memory graph is initialized following the same procedures mentioned in Section 3.2. The graph contains some landmark observations in the test set and helps the agent's visual language navigation. Although the agent captures panoramic observations, only the front and back observations of the agent are stored to reduce the memory consumption of the memory graph.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We have made main claims in the abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We have discussed the limitations of our work.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.

- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: We have provided the full set of assumptions and a complete (and correct) proof for each theoretical result.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We have fully disclosed all the information needed to reproduce the main experimental results of the paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same

dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We have provided open access to the data and code.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We have specified all the training and test details necessary to understand the results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We have reported appropriate information about the statistical significance of the experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We have provided sufficient information on the computer resources needed to reproduce the experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: The research conducted in the paper conforms, in every respect, with the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: We have discussed potential positive societal impacts of the work performed.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The paper poses no such risks

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.

- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite the original paper that produced the code package and dataset.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: New assets are introduced in the paper well documented and is the documentation provided alongside the assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.