
Perceive, Reflect, and Plan: Designing LLM Agent for Goal-Directed City Navigation without Instructions

Qingbin Zeng

Department of Electronic Engineering
Tsinghua University
Beijing, China
qingbinzeng06@gmail.com

Qinglong Yang

Department of Electronic Engineering
Tsinghua University
Beijing, China
yangql20@mails.tsinghua.edu.cn

Shunran Dong

Department of Electronic Engineering
Tsinghua University
Beijing, China
dsn@mails.tsinghua.edu.cn

Heming Du

School of Computing
Australian National University
Australia
heming.du@anu.edu.au

Liang Zhang

School of Computing
Australian National University
Australia
liang.zheng@anu.edu.au

Fengli Xu*

Department of Electronic Engineering
Tsinghua University
Beijing, China
fenglixu@tsinghua.edu.cn

Yong Li

Department of Electronic Engineering
Tsinghua University
Beijing, China
yongli07@tsinghua.edu.cn

Abstract

This paper considers a scenario in city navigation: an AI agent is provided with language descriptions of the goal location with respect to some well-known landmarks; By only observing the scene around, including recognizing landmarks and road network connections, the agent has to make decisions to navigate to the goal location without instructions. This problem is very challenging, because it requires agent to establish self-position and acquire spatial representation of complex urban environment, where landmarks are often invisible. In the absence of navigation instructions, such abilities are vital for the agent to make high-quality decisions in long-range city navigation. With the emergent reasoning ability of large language models (LLMs), a tempting baseline is to prompt LLMs to “react” on each observation and make decisions accordingly. However, this baseline has very poor performance that the agent often repeatedly visits same locations and make short-sighted, inconsistent decisions. To address these issues, this paper introduces a novel agentic workflow featured by its abilities to *perceive*, *reflect* and *plan*. Specifically, we find LLaVA-7B can be fine-tuned to *perceive* the direction and distance of landmarks with sufficient accuracy for city navigation. Moreover, *reflection* is achieved through a memory mechanism, where past ex-

* Corresponding author

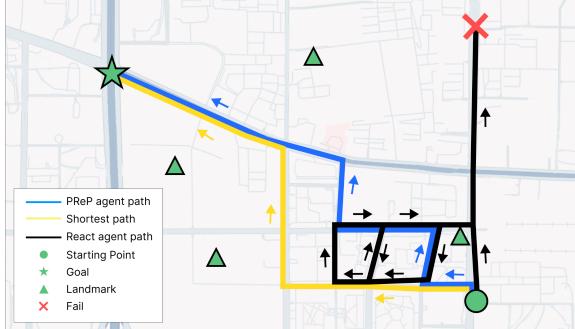


Figure 1: An illustrative comparison of city navigation results. The proposed workflow method (blue) successfully reaches the goal, and its path is similar to the shortest path (yellow). The React method (black, without workflow) keep going around in circles and fails to reach the goal.

periences are stored and can be retrieved with current perception for effective decision argumentation. *Planning* uses reflection results to produce long-term plans, which can avoid short-sighted decisions in long-range navigation. We show the designed workflow significantly improves navigation ability of the LLM agent compared with the state-of-the-art baselines. Our code and datasets are available: <https://anonymous.4open.science/r/PReP-13B5>

1 Introduction

Navigation in complex and unknown urban environment is an important task for artificial intelligent agents. This paper studies goal-directed agent navigation in the city, where an agent is provided with visual perception and goal location described by the relation to some well-known landmarks, *e.g.*, “the destination is approximately 300 meters northeast from the Skyscraper A”. The agent should visually identify the landmarks from street view images, use them as anchors to infer the direction of and distance from the goal, and take a series of actions to navigate to the goal without instructions. The task is challenging because it requires the agent to be aware of its own location and acquire spatial understanding of complex urban environment, where landmarks are sometimes invisible. In the absence of navigation instructions and maps, such abilities are vital for the agent to make high-quality decisions in long-range navigation.

Existing literature does not provide a ready-to-use solution to this task. A few recent works [1] [2] assume the availability of step-by-step language instructions and thus are not applicable to our task. Another branch of literature focus on designing reinforcement learning models [3] [4] [5], which often facing challenge of inefficient data and sensitivity to perturbations of the environment.

We explore the use of large language models (LLMs) for this task. React [6] is a straightforward baseline to ground reasoning ability of LLMs in city environment. At each step, this method visually perceives the street views which is used to make an action decision. This process is iteratively performed until reaching the goal or running out of the navigation budget. While React has some success attempts in indoor environments, it performs poorly in complex urban environments, which can be attributed to two main reasons. First, because each action decision is based only on the current observation, the agent may repeat actions previously taken and find itself going around in circles. See Fig. 1 for an example. Second, React is short-sighted, focusing only on the immediate step. Without considering long-term action sequences, the agent would be prone to taking more actions than actually needed.

This paper proposes an effective agentic workflow that improves the goal-directed city navigation ability of LLMs. To avoid isolated decision making, we propose a memory scheme. The historical trajectories and observations are stored and summarized to learn an intrinsic spatial representation of the environment, *i.e.*, an internal city map. The agent combines the historical experience and current observation to infer the goal direction. To improve over short-sighted actions, we resort to long-term planning. Specifically, considering the reflections and current road network connection, the agent decompose the full navigation path into several sub-goals, ensuring consistent and reasonable movement to the final goal during long-range navigation. In addition, we find that the fine-tuned LLaVA can perceive the direction and distance of landmarks with sufficient accuracy for navigation.

These components form the ‘*Perceive, Reflect, and Plan*’ workflow which allows the agent to perform long-range city navigation.

Our method only requires training the visual perception part with vision-language pairs of landmarks. The memory and planning parts both operate with few-shot examples (but also support fine-tuning). Compared to RL methods, our approach offers a more data-efficient solution. Compared with instruction-following methods, our system does not rely on explicit instructions, allowing for greater autonomy in navigation.

We collect two navigation datasets reflecting CBD scenes in Beijing and Shanghai. They contain complex road networks with thousands of road nodes and street views. On the two datasets, the proposed workflow significantly outperforms methods that could be applied (but are not specific) to our task. We find the perception component produces accurate spatial relations to support city navigation, the success rate of which is only 5% lower than navigation with ground truth perception results. Besides, we show that reflection and planning can further contribute to the success rate and that our method remains useful when dealing with long-range navigation tasks.

2 Related Work

Vision and language navigation (VLN) aims to enable agents to autonomously navigate in visual environments based on natural language instructions [7][8]. The field evolves from indoor to urban settings, with expanded scope of tasks and datasets. Anderson *et al.* created an early VLN dataset, while Mirowski *et al.* [9] introduce cross-modal matching models that leverage attention and reinforcement learning for vision and text integration. Datasets like TOUCHDOWN [1], Retouchdown [10], and StreetNav [11], allow VLN to transition to more complex urban environments. Recently, the use of LLMs has introduced new solutions in VLN [12][13][14], which achieved success with indoor environment. Other works [15][2] focusing on outdoor VLN, which use strong language understanding capabilities of LLMs for navigation based on ground-level instructions. In comparison, our paper does not rely on language instructions, where our city-navigation system explores the spatial cognitive abilities of LLMs.

Agentic workflows with LLMs. The exploration of agentic workflows using LLMs emerges as a effective strategy for planning problems [16]. Agentic workflows emphasize a step-by-step refinement process rather than single-step output generation. This strategy involves four key patterns: reflection, tool-use, planning, and multi-agent collaboration. Studies such as inner monologue [17] and reflexion [18] demonstrate the effectiveness of reflection in enhancing agentic understanding and reducing errors. Meanwhile, interactive planning methods like DEPS [19] and RAP [20] enable more structured and conscious planning. Other workflows, including CaP [21], ProgPrompt [22], CoT [23] and ToT [24], contribute to the collective understanding of how LLMs can be directed towards goal-oriented tasks. In our work, we design the agentic workflow with perception, reflection and plan modules, using inherent abilities of LLMs to tackle complex urban navigation.

3 Task Description, Dataset, and Baseline

3.1 Task Description

In this study, an agent navigates in the urban environment to find the goal with visual perception and goal description. To define the task exactly, we give following definitions:

Definition 1 (Urban Environment) The urban environment for navigation task can be described as an undirected graph $G = \langle V, E \rangle$. Each node $v_i \in V$ represents an position in the road, while street views $S_i = \{s_i^1, s_i^2, \dots, s_i^k\}_{v_i}$ are the attached visual information for node v_i . The edge $e_{ij} \in E$ donates the moving path between v_i and v_j . Define E_i as the collection of all edges connect to the node v_i . What’s more, landmarks $LM = \{lm_1, lm_2, \dots, lm_n\} \in V$ are defined as isolated vertex in the urban environment G . As the graph corresponds to a real urban scene, and the relative positions of each node are fixed. We define the relative position relationship of v_i with respect to v_j as $R(v_i, v_j)$.

Definition 2 (Urban Navigation Task) The urban navigation task can be formulated as finding a path from the start node v_s to the goal node v_g in the graph G . To be exact, given a navigation task $T = \langle v_s, v_g, D \rangle$, the target is to find a shortest path to navigate to the goal. Description $D = \{R_1, R_2\}$ is to determine the goal, containing $R_1 = \{R(v_g, lm) | lm \in LM_s\}$ and $R_2 =$

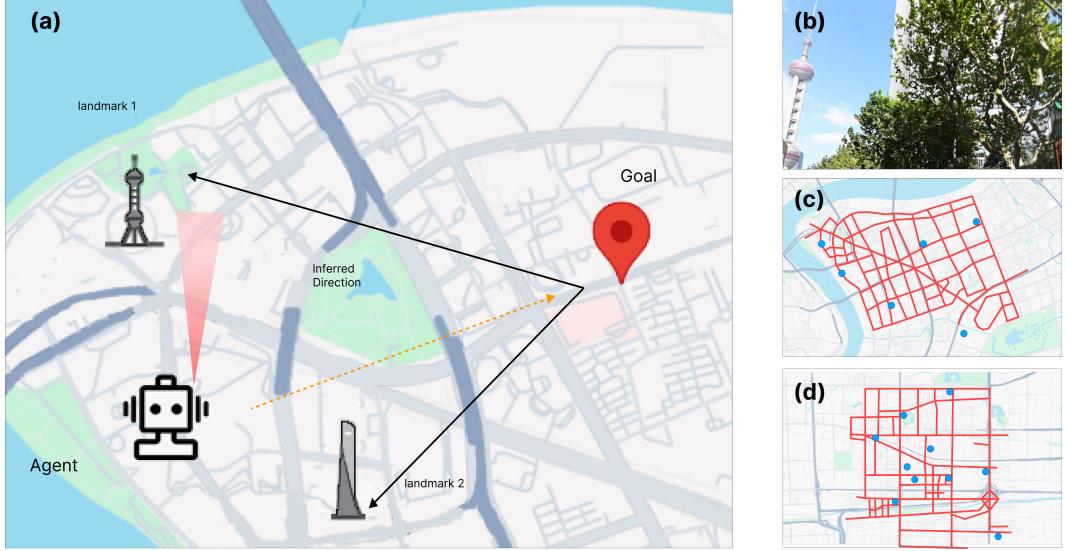


Figure 2: **Task example and dataset regions.** A task example is shown in (a). The instruction to the agent is the relative location of the goal *w.r.t* the landmarks in the city environment. The agent perceives the street views like (b). The agent has to infer the goal position relative to its current location using its observations of landmarks and move through the urban space. The road network in the chosen CBD areas in Shanghai (c) and Beijing (d). Blue points represent the landmarks used in the datasets, while red lines are roads.

$\{R(lm_i, lm_j) | lm_i, lm_j \in LM\}$, where R_1 is the relative position between goal and landmarks visible in goal nodes, and R_2 is the relative position among all landmarks in the environment.

Definition 3 (Agent for Urban Navigation Task) At timestamp t and node v_t , the agent makes a decision to move to the next node $v_{t+1} = agent(T, S_t, E_t)$, utilizing the street views and road connectivity. The agent recognizes the landmarks from the street views and then infer the goal direction and distance $R(v_t, v_g)$. Utilizing the above information, the agent navigates in the city street to find a path to the goal $p = [v_s, \dots, v_t, v_{t+1}, \dots, v_g]$.

3.2 Dataset

We collect data from central business districts (CBDs) of Beijing and Shanghai, which have a radius of a few kilometers. From this range, road network data are extracted and discretized at intervals of 50 meters forming the urban environment G . Each node of the road network is associated with the corresponding street view images. The number of street view images is related to the degree of the node. Road network visualization and a few examples of the dataset are shown Fig. 2(a).

Specifically, the selected area in Beijing is the Guomao CBD area, with a radius of approximately 3 kilometers, which includes a total of 1,134 nodes and 2,742 street view images, along with 10 landmark buildings. In Shanghai, the selected area is the Lujiazui CBD area, also with the similar radius, containing a total of 1,038 nodes and 2,366 street view images, along with 10 landmark buildings. The landmarks chosen in the dataset are well-known buildings with unique features. Street view images were obtained from Baidu Map Street View API, with a field of view of 90° and an elevation angle of 20° for each image. The image resolution is 1,024×512 pixels.

3.3 Baseline

A straightforward baseline iterates between two steps: visual perception and react. Specifically, after perceiving a street view from its current location, the agent predicts the direction of and distance to the target. Then, based on these predictions, the agent decides the next move. These two steps iterate until the agent reaches the goal or runs out of the navigation budget.

Here, visual perception is performed using fine-tuned LLaVA. We feed the street view images to LLaVA with the question “*Is [landmark] in the image? If so, estimate its direction and distance.*”. We ask this question for all landmarks in this region. The agent already has the relative position of the goal from different landmarks, so it can infer the goal direction from the perception results. Based on the perception and direction inference results, also considering the current road connection, the agent uses the zero-shot LLM reasoning ability to decide the next move. Note that the perception method with LLaVA is also used in the proposed agent workflow. More details are provided in Section 4.2.

4 Proposed Agent Workflow

4.1 Workflow Overview

The overall agent workflow is shown in the Figure 3, which is consist of three parts: visual perception, reflection with memory, and planning. While visual perception uses LLaVA, both reflection and memory uses large language models (LLMs). As described in Section 4.2, *visual perception* allows the agent to recognize the landmarks in the street view images and predict the direction and distance of the target. Perception results are passed to the *reflection* part, where the agent reevaluates the perception results and reflects the goal location. In reflection, long-term memory is set up to summarize and learn from the historical trajectory for constructing intrinsic map representations, a topological map between nodes that the agent has visited. The *planning* module is the decision core of the workflow. It generates a navigation plan by considering both the goal direction after reflection and the current road connections. The agent follows this plan to make the next move. Then the agent state will be updated to start next iteration. These steps compose a workflow that enables the agent to sense its surroundings, remember past experience, and plan its actions. As shown in Section 5 , this workflow yields significantly higher navigation success rates compared to the ‘React’ baseline.

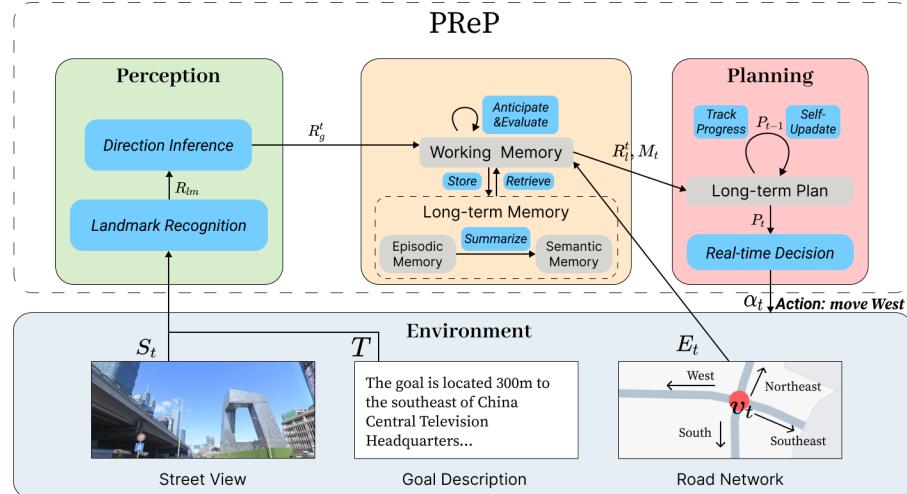


Figure 3: **Overview of PReP workflow.** PReP has three steps: perception, reflection, and planning. Blue boxes represent LLMs or LLaVA, while gray boxes indicate variables stored by natural language. The definition of mathematical symbols are in section 3.

4.2 Perception

Perceiving the landmarks. At timestamp t and node v_t , the agent gets the street views $S_t = \{s_t^1, s_t^2, \dots, s_t^k\}$. The agent then detects the landmarks and estimates their direction and distances $R_{lm} = \{R(lm_i, v_t) | lm_i \in LM\}$ to the agent itself. We perform the task using LLaVA [25]. Zero-shot LLaVA has poor recognition accuracy because the landmarks we use are probably not in its training data. We therefore fine-tunes LLaVA using the LoRA method [26]. To do so, we collect 5,000 landmark images and generate 30k Q&A conversation data. More details can be seen in the Appendix A.1.

Inferring directions to the goal. While the agent acquires the landmarks position to the agent R_{lm} , it is necessary to infer the goal direction combining R_{lm} and task description T . We can use an LLM to perform this inference. The output $R_g^t = R(v_g, v_t)$ of is the direction and distance of the goal relative to the agent. The essence of this problem is a calculation process of the cosine theorem, so the results can be further improved using calculators.

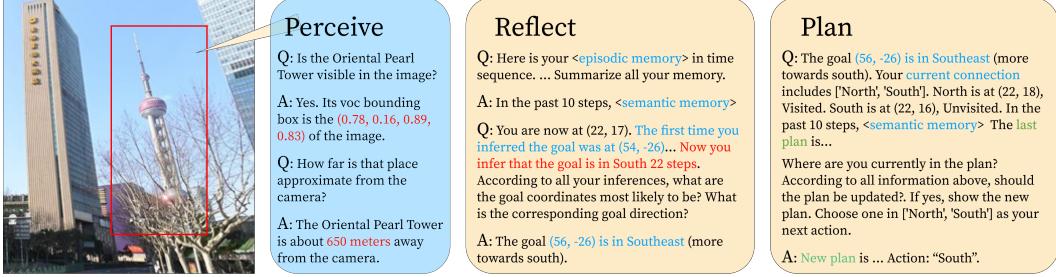


Figure 4: **Sample prompts and responses in the PReP workflow.** In perception, a vision language model locates the landmarks and estimates their distances to the agent. In reflection, the agent presents all past memory and short-term memory, and gives an estimate of the location and direction of the goal. In planning, the agent uses the output from reflection to update the plan. Prompts have been simplified while retaining their original meaning. The full prompts are provided in Appendix E

4.3 Reflection

Reflection is critical in our workflow, which summarizes past experience and reflects on visual perception results. This step has two main components: long-term memory and working memory. Long-term memory consists of episodic memory and semantic memory, where episodic memory stores navigation data and semantic memory saves summary of history navigation experience. Working memory serves as a data buffer to process the visual perception results and retrieved memory.

Episodic memory. Episodic memory is a list of the navigation data in natural language. When the agent moves from v_t to v_{t+1} , this action and the perception results R_g^t in v_t are processed into a sentence and stored. The detail of the sentence format is in Appendix E. Since the past navigation data are stored, the agent can retrieve the goal inference at history location and detect whether connected nodes are visited. For example, when the agent has saved navigation data in v_t and navigated to v'_t , it can retrieve R_g^t and the status of connected nodes $E_{t'}$ to help reflection and planning. These retrieved memories are buffered in working memory space for further processing.

Semantic memory. While episode memory records the experiences, the agent uses LLMs to summarize and learn from the episodic memory to form the semantic memory. The semantic memory is a high-level cognitive function that assists the agent in constructing an intrinsic representation of the navigation map. Like a human, it can understand the environment based on historical experience and learn more advanced navigation strategies, such as detours required to reach the destination. These strategies can be retrieved to working memory and further beneficial to the planning process. As the agent navigates, episodic memory and semantic memory are updated accordingly.

Working memory. Working memory receives visual perception results R_g^t and retrieves relevant experiences from long-term memory. It has an anticipation-reevaluation mechanism to solve the problem where the agent cannot detect any landmarks in the street views and loses goal direction. Specifically, the agent uses historical perception results $R_g^{t'}$ and moving direction to anticipate the potential goal direction R_p^t , if any landmark can be observed, the agent reevaluates whether the current inference R_g^t is reasonable, and synthesizes a new inference $R_l^t = LLM_{reflect}(R_g^t, R_p^t)$. The output of working memory to the planning part are the synthesized inference R_l^t and retrieved memory. This enables the agent to tackle complex environments regardless whether landmarks are visible or not, making the agent more flexible and robust.

4.4 Planning

Instead of reacting directly to observations, we use a planning module in our workflow. It involves long-term planning and short-term decision-making. Specifically, long-term planning uses reflected goal inference R_l^t , retrieved memory M_t , and the old plan P_{t-1} as input. It updates the navigation plan at timestamp t $P_t = LLM_{plan}(P_{t-1}, R_l^t, M_t)$ by analysing the old plan and breaking down the possible path into sub-goals. The agent firstly analyses what stage does the old plan has been carried out, and then synthesizes the goal inference, retrieved memory and the connection status to decide whether the plan should be updated. If so, the agent updates the plan by predicting the possible route to the goal, and breaking down the full route into few sub-goals like “*move [east] until [an intersection]*”. Meanwhile, the short-term decision maker translates the plan into action $\alpha_t = LLM_{action}(P_t, E_t)$ based on the road connections E_t . The action α_t means a move from node v_t to node v_{t+1} , making the agent updates its location and explore the goal in the environment.

5 Experiments

5.1 Experimental setup

We experimentally evaluate the performance of the proposed agentic workflow on the simulated urban navigation task described in section 3. We use success rate (SR) and success rate weighted by path length (SPL) to measure system effectiveness and efficiency, respectively [27].

We use two test sets, one for Beijing CBD, and the other for Shanghai CBD. Each test sets have 100 different navigation tasks with different starting points. Each starting point is at a road node, where at least one landmark must be visible; otherwise, the agent will randomly wander. The minimum number of steps required from the starting point to the goal followed a normal distribution with $\mu=30$ steps and $\sigma=10$ steps. Because each steps translates to 50 meters on the map, it means the average step is 30, and average navigation distance is 1,500 meters. We set the iteration limit as 2.5 times as the minimum steps. If the agent moves more than the limited step, we think the navigation task is failed.

5.2 Main Evaluation

Comparison with existing methods that might give a solution. We compared PReP with existing language-based methods, including code as policies (CaP) [21], ProgPrompt [22], inner monologue [17], and chain of thought (CoT) [23]. These method are using well-designed prompts to fit our urban navigation task and we list the prompts for different methods in Appendix E. We also implement two baselines that are not based on LLMs. The ‘random’ method selects a random direction from the current connection each time. Reinforcement learning (RL) [9] is trained for 1 million steps in the environment to learn the policy for finding the goal. The perception module for all the methods is the same. All the language-based methods use GPT-4-turbo as the base model, and all the hyper-parameters of LLMs are the same for a fair comparison.

From Table 1, we clearly observe that PReP yields the best navigation performance compared with existing methods. We have two observations. **First**, the success rate of Random is nearly 0, indicating the significant challenge of this task. Existing language-based methods have improved performance, suggesting that LLMs possess the capability to navigate in cities based on goal direction. **Second**, we achieve **SR = 63% and 57%** on Beijing and Shanghai test sets, respectively, and **SPL = 47.67% and 42.15%** on Beijing and Shanghai test sets, respectively. The second best method is CoT, the SR of which is 5% and 17% lower than our method, on Beijing and Shanghai test sets, respectively. It indicates the effectiveness of PReP.

Effectiveness of the proposed planning and reflection methods. We conduct ablation studies to validate the usefulness of the reflection and planning methods. We keep perception part unchanged, assuming all variants can recognize landmarks and infer goal directions. Results are summarized in Table 2. ‘PReP’ indicates the complete PReP workflow, which includes both planning and reflection methods. ‘w/o Reflection’ indicates that the agent receives the perception and retrieved the episode memory but without constructing semantic memory and reflection on goal inference. ‘w/o Planning’ indicates that the agent normally get the reflected goal inference and retrieved memory, but makes decisions without formulating a long-term plan. ‘Plain’ is the combination of ‘w/o Reflection’ and

Table 1: **Comparing PReP with previous methods.** ‘Random’ and ‘RL’ are not based on LLMs, while the latter methods, although using different structures, use LLMs as planners and decision-makers. The full prompts are in Appendix E.

Methods	Beijing		Shanghai	
	SR (%)	SPL (%)	SR (%)	SPL (%)
Random	1	0.41	2	1.02
RL	14	12.35	13	10.75
CaP	15	10.72	13	10.32
ProgPrompt	25	16.66	23	16.53
InnerMonologue	38	28.64	26	19.47
CoT	58	41.61	34	22.76
PReP	63	47.67	57	42.15

Table 2: **Comparing method variants.** ‘Plain’ means PReP with neither planning nor reflection.

Methods	Beijing		Shanghai	
	SR (%)	SPL (%)	SR (%)	SPL (%)
Plain	33	24.94	25	18.21
PReP	63	47.67	57	42.15
PReP w/o Planning	59	45.86	52	39.59
PReP w/o Reflection	43	28.78	26	19.43

‘w/o Planning’, where the agent makes decisions directly based on perception results and history experience in a list without planning and reflection.

We clearly see that the full system has the best performance. For example, on the Beijing test set, the full system yields a higher success rate of 30%, 4%, and 20% over ‘Plain’, ‘PReP w/o Planning’, and ‘PReP w/o Reflection’, respectively. This indicates the necessity of having both steps in our system. We also observe that removing reflection leads to larger performance drop comparing with removing planning. It suggests the importance of reflection.

5.3 Further Analysis

Benefit of fine-tuning LLaVA over zero-shot LLaVA. In Table 3, we compare fine-tuned LLaVA with zero-shot LLaVA. Zero-shot LLaVA has much poorer performance: on the Shanghai test sets, its SR and SPL is 46% and 33.26% lower than its fine-tuned version. It indicates that LLaVA does not naturally recognize landmarks through Baidu Maps. But interestingly, zero-shot LLaVA still has 10+% success rate. This can be explained by its 19% accuracy, 6% precision, 93% recall, and 0.64\$ IoU. In fact, zero-shot LLaVA has good building detection capacity and assume that most images contain landmarks, leading to a high recall. Its precision is low (6%), but sometimes is fine for the agent to find the goal.

We also compare with an oracle setting, where the perception results are replaced with groundtruth directions and distances measured by GPS. Compared with oracle results, fine-tuned LLaVA is 4% and 5% lower in SR on Beijing and Shanghai test sets, respectively. This is not a significant gap, indicating the effectiveness of fine-tuning.

Comparing different LLMs. We now use different LLMs to perform inference (blue boxes in Fig. 4). These models include GPT-3.5-turbo, GLM-4 [28], Mistral-7B [29], LLaMA3-8B [30] and GPT-4-turbo. From Table 4, we observe that GPT-4-turbo significantly outperforms other LLMs without fine-tuning. Moreover, we then use the question-answering data generated by GPT-4-turbo to fine-tune LLaMA3 [31]. The fine-tuned LLaMA3 model achieves performance that was second only to GPT-4-turbo among all models.

Table 3: **Comparisons among ablation variants of the perception module.** A, P, R represent Accuracy, Precision, and Recall of landmark recognition. IoU represents the degree of overlap between the predicted bounding box and the actual bounding box of the landmark.

Methods	A	P	R	IoU	Beijing		Shanghai	
					SR(%)	SPL(%)	SR(%)	SPL(%)
LLaVA	0.19	0.06	0.93	0.64	15	13.01	11	8.89
LLaVA-FT(ours)	0.99	0.98	0.96	0.91	63	47.67	57	42.15
Oracle	-	-	-	-	67	51.27	62	45.20

Table 4: **Performance of PReP using different LLMs.** Zero-shot spatial reasoning capabilities of most LLMs are limited, and PReP achieves best performance with GPT-4-turbo

Methods	Beijing		Shanghai	
	SR (%)	SPL (%)	SR (%)	SPL (%)
GPT-3.5-turbo	9	5.40	7	4.53
GLM-4	18	12.77	17	12.30
Mistral-7B	1	0.41	2	1.02
LLaMA3-8B	8	4.75	6	5.03
LLaMA3-8B-FT	22	16.76	24	15.80
GPT-4-turbo (ours)	63	47.67	57	42.15

Impact of goal distance. We analyze whether the distance between the goal and the agent has an impact on success rate. In Fig. 6a, we observe that the success rate does not decrease a lot when the goal is as far as 2 kilometers (40 - 50 steps) from the agent. We also notice that the curves drops at 20-40 steps but increases at 40-50 steps. The possible reason is that the iteration limit increases as the distance increases, the agent may fully explores the environments and find the goal more easily.

Furthermore, in Fig. 6b, we study how the goal distance affects the number of steps taken by PReP. As the distance increases, generally the plots become more scattered, indicating the task is becoming more challenging. Basically the agent takes 1 or 2 times more of the minimum number of steps to reach the goal.

Impact of landmark visibility. When the agent performs a navigation task, some nodes it passes through can observe landmarks, while others cannot. We study how landmark visibility along its path impacts its success rate. In Fig. 6c, we plot the relationship between the number of nodes in the path that can observe any landmark and the total number of nodes in successful tasks. From the plot, we can find some valuable results. When the PReP agent performed tasks in the Beijing test set, it could identify landmarks in approximately 50% of the nodes traveled on average. In the Shanghai test set, this percentage dropped to less than 20%. Despite this challenge, performance in the Shanghai test set was not significantly lower than in the Beijing test set. This demonstrates that the PReP agent can efficiently navigate even in environments with sparse landmark visibility.

Computational cost. The primary training cost is LLaVA fine-tuning, which requires one NVIDIA GeForce RTX A100 GPU with 80G memory for approximately 3 hours with 30k conversation data. Each request-response cycle of the fine-tuned LLaVA on the same GPU takes 6 to 8 seconds, while calling the LLM API takes 2 to 5 seconds (varies among different models). Each iteration for an agent step takes about 12 seconds. In future we will work to optimize the inference process.

A case study. We conduct a case study to illustrate the role of reflection and planning (see Fig. 6). In this case, the PReP agent deviates from the shortest path in the beginning but still successfully reaches the goal. Initially, the agent infers that the goal was located to the east. However, as there was no direct path to due east, the agent planned a detour: it first heads northeast and then turns towards either east or southeast. The agent continuously makes inferences of the goal location each time it observes a landmark along its route. When the agent cannot perceive landmarks in the street views and lose the goal direction, it can reflect on history memory including the moving trajectory and goal

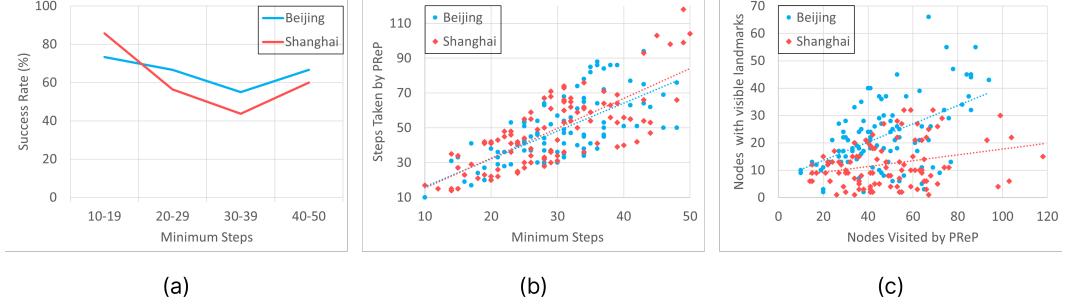


Figure 5: **Performance of PReP across varying task difficulties.** (a): Success rate vs. minimum number of steps required to reach the goal. (b): Number of steps taken by PReP vs. the number of minimum steps required to reach the goal. (c) : Number of nodes where agent can observe landmarks (y axis) vs. number of nodes visited in one task (x axis).

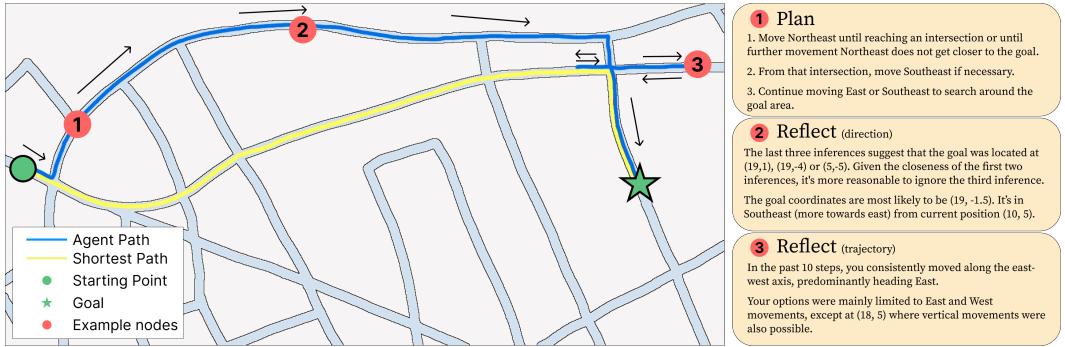


Figure 6: **Success case study.** Here we choose three typical reasoning process in the whole navigation task where the arrows show the direction of travel of the agent. The agent makes a long-term plan at location (1) so that it can move consistently without moving back. At location (2), the agent cannot perceive landmarks in the street views and infer goal direction, but it can reflect history memory and anticipate the goal position. At location (3) the agent summarize its moving trajectory and find itself moving in the deviate direction, thus returning to the right route.

inference, and then anticipate the goal direction from current position. When the agent moves in the deviate direction, it can reflect its trajectory and re-plan the right route.

6 Conclusion

We propose an agentic workflow for goal-directed city navigation without step-by-step language instructions or maps. The workflow includes a fine-tuned LLaVA model for spatial perception, a memory module for synthesizing and reflecting perception results and retrieved memory, and a planning module for navigation route planning. As our approach only requires training the visual perception part, it is a more data-efficient solution compared to RL methods. Owing to the well-designed reflection and planning part, the agent can perform the long-term navigation task in complex environment and achieves a success rate of about 60%. Further experiments show that the agent performs well in two cities and various difficulty levels, demonstrating robustness and flexibility. Our contributions not only present an effective agentic workflow for using LLMs in goal-directed urban navigation, but also validate the potential of LLMs for complex spatial tasks.

References

- [1] 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.

- [2] 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.
- [3] Piotr Mirowski, Razvan Pascanu, Fabio Viola, Hubert Soyer, Andrew J Ballard, Andrea Banino, Misha Denil, Ross Goroshin, Laurent Sifre, Koray Kavukcuoglu, et al. Learning to navigate in complex environments. *arXiv preprint arXiv:1611.03673*, 2016.
- [4] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3357–3364. IEEE, 2017.
- [5] Yi Wu, Yuxin Wu, Georgia Gkioxari, and Yuandong Tian. Building generalizable agents with a realistic and rich 3d environment. *arXiv preprint arXiv:1801.02209*, 2018.
- [6] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.
- [7] 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.
- [8] Wansen Wu, Tao Chang, Xinxing Li, Quanjun Yin, and Yue Hu. Vision-language navigation: a survey and taxonomy. *Neural Computing and Applications*, 36(7):3291–3316, 2024.
- [9] Piotr Mirowski, Matt Grimes, Mateusz Malinowski, Karl Moritz Hermann, Keith Anderson, Denis Teplyashin, Karen Simonyan, Andrew Zisserman, Raia Hadsell, et al. Learning to navigate in cities without a map. *Advances in neural information processing systems*, 31, 2018.
- [10] Harsh Mehta, Yoav Artzi, Jason Baldridge, Eugene Ie, and Piotr Mirowski. Retouchdown: Adding touchdown to streetlearn as a shareable resource for language grounding tasks in street view, 2020.
- [11] Karl Moritz Hermann, Mateusz Malinowski, Piotr Mirowski, Andras Banki-Horvath, Keith Anderson, and Raia Hadsell. Learning to follow directions in street view. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11773–11781, 2020.
- [12] 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.
- [13] Vishnu Sashank Dorbala, James F Mullen Jr, and Dinesh Manocha. Can an embodied agent find your “cat-shaped mug”? llm-based zero-shot object navigation. *IEEE Robotics and Automation Letters*, 2023.
- [14] Weiqin Zu, Wenbin Song, Ruiqing Chen, Ze Guo, Fanglei Sun, Zheng Tian, Wei Pan, and Jun Wang. Language and sketching: An llm-driven interactive multimodal multitask robot navigation framework. *arXiv preprint arXiv:2311.08244*, 2023.
- [15] 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.
- [16] Theodore R Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas L Griffiths. Cognitive architectures for language agents. *arXiv preprint arXiv:2309.02427*, 2023.
- [17] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022.
- [18] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.
- [19] Zihao Wang, Shaofei Cai, Guanzhou Chen, Anji Liu, Xiaojian Ma, and Yitao Liang. Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents. *arXiv preprint arXiv:2302.01560*, 2023.

- [20] Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. Reasoning with language model is planning with world model, 2023.
- [21] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9493–9500. IEEE, 2023.
- [22] Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. Progprompt: Generating situated robot task plans using large language models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11523–11530. IEEE, 2023.
- [23] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.
- [24] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems*, 36, 2024.
- [25] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024.
- [26] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.
- [27] Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018.
- [28] Zhipu. Zhipu ai devday glm-4. <https://zhipuai.cn/en/devday>, 2024.
- [29] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.
- [30] Meta. Introducing meta llama 3: The most capable openly available llm to date. <https://ai.meta.com/blog/meta-llama-3/>, 2024.
- [31] Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyuan Luo, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. *arXiv preprint arXiv:2403.13372*, 2024.

A Additional Experimental Details

A.1 Fine-tuning LLaVA

The methodology for fine-tuning LLaVA uses the LoRA (Low-Rank Adaptation) technique. This approach introduces trainable low-rank matrices to simulate parameter updates, enabling rapid task adaptation without significantly increasing model complexity. We collected 250 images for each of the 20 landmarks from the Internet, totaling 5,000 images. These images were manually annotated with the binary visibility and bounding box of each landmark. We split the data into an 80% training set and a 20% test set. Using these images, we generated dialogue data in a question-and-answer format. The questions we asked are as following step by step, aiming to form a pattern of CoT to improve its understanding and reasoning ability.

1. "Is the $landmark_i$ visible in the image?"
2. "The $landmark_i$ is visible in the image, what's the bounding box of it in the image?"
3. "The $landmark_i$ is visible in the image and its bounding box is $(x_{min}, y_{min}, x_{max}, y_{max})$, how far is it actually away from the camera?"

We generated about 30k turn dialogues to fine-tune the llava-v1.5-7b. The fine-tuning process was carried out on an NVIDIA GeForce RTX A100 GPU and took about 3 hours. The scripts to fine-tuning is modified from the official repository ² with the default parameters.

The outcome of this fine-tuning process is a model that demonstrates remarkable accuracy in landmark recognition and segmentation. While the distance estimation is not very accurate, the rough estimation results are still effective in subsequent navigation steps. The fine-tuned LLaVA model is essential for the agent to perceive the environment and obtain goal information.

Table 5: The ability to recognize and segment landmark in the street view images.

	Accuracy	Precision	Recall	F1-Score	IoU
LLaVA-base	0.1873	0.0576	0.9347	0.1072	0.6432
LLaVA-FT	0.9980	0.9868	0.9695	0.9779	0.9152

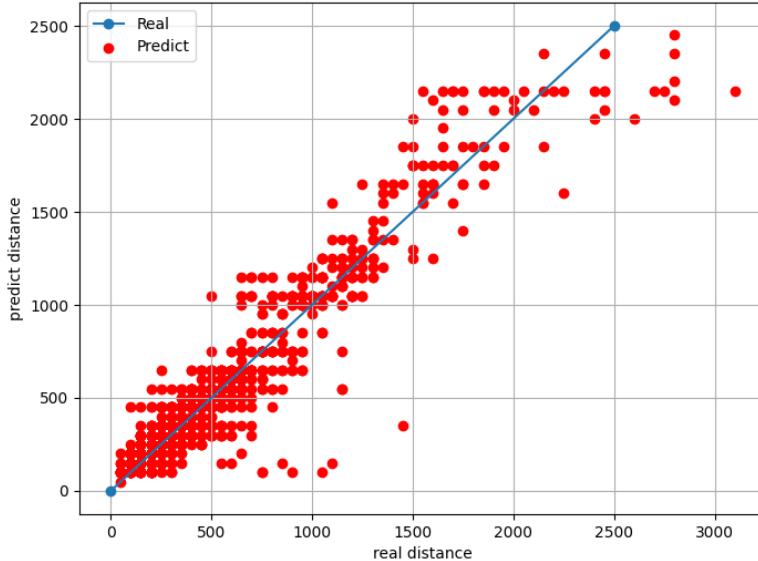


Figure 7: The distance estimation results of fine-tuned LLaVA.

²<https://github.com/haotian-liu/LLaVA>

A.2 Fine-tuning LLaMA3-8B

We are trying to transfer knowledge from a much large model (GPT-4-turbo) to a small model (LLaMA3-8B). The method involves using data generated during navigation with GPT-4-turbo to fine-tune LLaMA3-8B. We filter the successful samples from all the saved data and process them into the ShareGPT format. To avoid data breaches, we separate the Beijing and Shanghai datasets. This means we use data generated in Beijing to fine-tune LLaMA, which is then tested in Shanghai, and vice versa. We generated about 20k dialogue turns for each city dataset and fine-tuned LLaMA-8B using the LoRA method with one NVIDIA GeForce RTX A100 GPU. This process took about 30 minutes using the LLaMA-Factory tool³. All parameters in the fine-tuning process were set to default.

The results are shown in Table 4. We see that while LLaMA3-8B performs slightly worse than GPT-3.5-turbo in our task, it significantly outperforms other LLMs (except GPT-4-turbo) after fine-tuning. Although there is still a gap compared to GPT-4-turbo, increasing the amount of fine-tuning data may improve its performance.

B Limitations

Despite the advancements and innovative approaches in our work, there are several limitations to consider. A notable challenge is the dependence on powerful closed-source models, such as GPT-4-turbo, for superior results. Although we fine-tuned the open-source LLaMA-8B model, its performance, while better than other LLMs, does not reach half that of GPT-4-turbo. This highlights a significant gap. Future work should explore more effective fine-tuning strategies for open-source models to enhance their capabilities, ensuring our progress in urban navigation can be widely adopted and further developed. Another issue is the limited size of our test set, which may cause fluctuations in the success rate. A larger and more diverse test set could provide a more comprehensive evaluation of the model’s performance and robustness.

C Ethical Analysis

The data we collected is open access including *Baidu StreetViews API* and *Open Street Map*, without privacy issues. Our ethical analysis confirms that all data was gathered in compliance with the code of ethics. We ensured that no personally identifiable information was collected, maintaining the anonymity and privacy of individuals. By adhering to the ethical guidelines, we avoid any potential privacy concerns and support the broader scientific community’s efforts to build on and verify our work.

D Broader Impacts

Developing the LLM agent for goal-directed city navigation has many positive social impacts. This technology extends beyond navigation robots, providing invaluable help to the visually impaired by enhancing their mobility and independence. Easier navigation in urban environments can greatly improve their quality of life and help them participate more fully in society. Additionally, using this technology in disaster relief can save lives by helping rescue teams navigate affected areas quickly and efficiently. However, there are potential negative social impacts to consider. Relying on AI for navigation might decrease human spatial awareness and problem-solving skills. To maximize the positive impact and minimize negative consequences, it is crucial to develop and implement this technology ethically with strong safeguards in place.

E Prompt for Different Methods

E.1 PReP Prompt

³<https://github.com/hiyouga/LLaMA-Factory/>

prompt 1: Prompt for PReP

Perception
Q:
Is the Beijing_Zhoxin_Building visible in the image?
A:
Yes. Its voc bounding box is the (0.58984, 0.46680, 0.63672, 0.72461) of the image.
Q:
The Beijing_Zhoxin_Building is visible in the image and its voc bounding box is (0.58984, 0.46680, 0.63672, 0.72461). How far is that place actually from the camera?
A:
The Beijing_Zhoxin_Building is about 1600 meters away from the camera.

Reflection
Q:
You are conducting a navigation task and here is your memory list in time sequence.
1. You were at (14, 14). You could move to ['Northeast', 'North', 'West', 'South'] from here. You chose to move to Northeast. You then arrived at (15, 15).
2. You were at (15, 15). You could move to ['East', 'Northwest', 'Southwest'] from here. You chose to move to East. You then arrived at (16, 15).
3. You were at (16, 15). You could move to ['West', 'East'] from here. You chose to move to East. You then arrived at (17, 15).
4. You were at (17, 15). You could move to ['West', 'East'] from here. You chose to move to East. You then arrived at (18, 15).
5. You were at (18, 15). You could move to ['Northeast', 'West'] from here. You chose to move to Northeast. You then arrived at (19, 16).
6. You were at (19, 16). You could move to ['Northeast', 'Southwest'] from here. You chose to move to Northeast. You then arrived at (20, 17).
7. You were at (20, 17). You could move to ['Northeast', 'Southwest'] from here. You chose to move to Northeast. You then arrived at (21, 18).
8. You were at (21, 18). You could move to ['Northeast', 'Southwest'] from here. You chose to move to Northeast. You then arrived at (22, 19).
9. You were at (22, 19). You could move to ['South', 'East', 'Southwest', 'North'] from here. You chose to move to South. You then arrived at (22, 18).
10. You were at (22, 18). You could move to ['North', 'South'] from here. You chose to move to South. You then arrived at (22, 17).
Summarize all your memory, what can you learn from it? Answer in no more than 2 sentence.
Format Example of Answer: 'In the past 15 steps, you initially wandered between East and West. You then headed East and then North to reach your current position. Now you are on a north-south road. If you keep going South, you will reach an intersection (1, -5) which can move to North, West and East. If you keep going North, you will reach a dead end (1, 5) with only one navigable direction.'

A:
In the past 10 steps, you primarily moved in a northeastern direction, transitioning from (14, 14) to (22, 19), and then shifted to a southern direction, moving from (22, 19) to (22, 17). Your journey involved navigating through options that often included moving east or northeast, with a recent change to north-south movement.

Q:
You are now at (22, 17). The first time you inferred the goal was at (54, -26). The second time you inferred the goal was at (58, -25). The third time you inferred the goal was at (19, -24). Now you infer that the goal is in South 22 steps. According to all your inferences, what are the goal coordinates most likely to be? What is the corresponding goal direction from current position?
Take (0,0) as the reference, (0,+1) as the North, (+1,0) as the East. When one of your inference differs significantly from other inferences, you should not consider it when calculating the most likely goal coordinates. You should think step by step to answer each question. Answer in the json format and keys are ['Thought_Q1', 'Answer_Q1', 'Thought_Q2', 'Answer_Q2']. Format Example of 'Answer_Q1': 'The goal coordinates are most likely to be (3, -2). Format Example of 'Answer_Q2': 'The goal (3, -2) is in Southeast(more towards east) from current position (0, 0).'
A:
"Thought_Q1": "To determine the most likely goal coordinates, consider the reasonable inferences and ignore the outlier. The first inference is (54, -26), the second is (58, -25), and the third is significantly different at (19, -24). Ignoring the third inference, I calculate the average of the first two coordinates: ((54 + 58)/2, (-26 -25)/2) = (56, -25.5), which simplifies to (56, -26) as coordinates are typically integers. Additionally, the current position is (22, 17), and the latest inference suggests a southern shift. Thus, moving 22 steps south from (22, 17) gives new coordinates (22, -5). This inference is also an outlier when compared to the other two reasonable estimations. Therefore, the most probable goal coordinates considering the first two inferences are (56, -26)."
"Answer_Q1": "The goal coordinates are most likely to be (56, -26)."
"Thought_Q2": "To determine the direction of the goal (56, -26) from the current position (22, 17), note the change in x and y coordinates. The x-coordinate changes from 22 to 56, an increase of 34, and the y-coordinate changes from 17 to -26, a decrease of 43. This indicates the goal is to the east and south of the current position, specifically more towards the south. Thus, the goal is southeast of the current position, but more towards the south."
"Answer_Q2": "The goal (56, -26) is in Southeast (more towards south) from current position (22, 17)."
Planning
Q:
You are a helpful navigate agent in the city. The goal coordinates are most likely to be (56, -26). The goal (56, -26) is in Southeast (more towards south) from current position (22, 17). You are now at (22, 17). Your current connection includes ['North', 'South']. North is at (22, 18), Visited. South is at (22, 16), Unvisited. In the past 10 steps, you primarily moved in a northeastern direction, transitioning from (14, 14) to (22, 19), and then shifted to a southern direction, moving from (22, 19) to (22, 17). Your journey involved navigating through options that often included moving east or northeast, with a recent change to north-south movement. The plan is ['1. Move South to head directly towards the goal while exploring unvisited areas.', '2. Continuum moving South if the path allows, to make progress directly towards the goal.', '3. Switch to East after making significant southward progress to address the eastward component of the goal location.']. Where are you currently in the plan? According to all information above, should the plan be updated?. If yes, show the new plan. According to your plan and current connection, choose one in ['North', 'South'] as your next action. Answer in the json format, and keys are ['current_state', 'yes_or_no', 'update_reason'(if has), 'new_plan'(if has), 'action_reason', 'action'].
You should follow these instructions: (1) The plan should indicate the direction without any specific nodes. (2) The plan should be a 3 steps list, e.g. When the goal is in the East direction and you are in a South-North lane, the plan should be [1. Move North until an intersection(because you can't directly move East); 2. From that intersection, move East if possible(because the goal is in the East); 3. Move South to search the goal.(because you first move North which deviates from the goal in y-coordinate)] (3) Note that the action must be one of the ['North', 'South']. Try not to move back unless necessary. (4) If your trajectory memory indicates you are wandering between two directions, you should move along a certain direction to break the loop. (5) Even if you have arrived at your inferred goal coordinates, you haven't found the goal yet, so you should search among unvisited areas nearby.
A:
"current_state": "Step 1"
"yes_or_no": "No"
"action_reason": "Currently at an unvisited south connection, which aligns with the plan to move directly towards the goal's more southern location."
"action": "South"

prompt 2: Prompt for PReP without planning

Q:
You are a helpful navigate agent in the city. The goal coordinates are most likely to be (56, -26).The goal (56, -26) is in Southeast (more towards south) from current position (22, 17). Your current connection includes ['North', 'South']. North is at (22, 18), Visited. South is at (22, 16), Unvisited. In the past 10 steps, you primarily moved in a northeastern direction, transitioning from (14, 14) to (22, 19), and then shifted to a southern direction, moving from (22, 19) to (22, 17). Your journey involved navigating through options that often included moving east or northeast, with a recent change to north-south movement. According to all information above, choose one in ['North', 'South'] as your next action. Answer in the json format, and keys are ['action_reason', 'action'].

A:
"action_reason": "Currently at an unvisited south connection, you should explore the unvisited area."
"action": "South"

prompt 3: Prompt for PReP without reflection

Q:
You are a helpful navigate agent in the city. You are now at (22, 17). The first time you inferred the goal was at (54, -26). The second time you inferred the goal was at (58, -25). The third time you inferred the goal was at (19, -24). Now you infer that the goal is in South 22 steps. Your current connection includes ['North', 'South']. North is at (22, 18), Visited. South is at (22, 16), Unvisited. Here is your memory list in time sequence... The last plan is ['1. Move South to head directly towards the goal while exploring unvisited areas.', '2. Continue moving South if the path allows, to make progress directly towards the goal.', '3. Switch to East after making significant southward progress to address the eastward component of the goal location.']. Where are you currently in the plan? According to all information above, should the plan be updated?. If yes, show the new plan. According to your plan and current connection, choose one in ['North', 'South'] as your next action. Answer in the json format, and keys are ['current_state', 'yes_or_no', 'update_reason'(if has), 'new_plan'(if has), 'action_reason', 'action'].

A:
"current_state": "Step 1"
"yes_or_no": "No"
"action_reason": "Currently at an unvisited south connection, which aligns with the plan to move directly towards the goal's more southern location."
"action": "South"

prompt 4: Prompt for plain PReP

Q:
You are a helpful navigate agent in the city. You are now at (22, 17). The first time you inferred the goal was at (54, -26). The second time you inferred the goal was at (58, -25). The third time you inferred the goal was at (19, -24). Now you infer that the goal is in South 22 steps. Your current connection includes ['North', 'South']. North is at (22, 18), Visited. South is at (22, 16), Unvisited. Here is your memory list in time sequence... According to all information above, choose one in ['North', 'South'] as your next action. Answer in the json format, and keys are ['action_reason', 'action'].

A:
"action_reason": "Currently at an unvisited south connection, you should explore the unvisited area."
"action": "South"

E.2 Baselines Prompt

prompt 5: Prompt for InnerMonologue

Q:
Scene:
You are now at (0, -27). The first time you inferred the goal was at (0, -42). The second time you inferred the goal was at (0, -42). The third time you inferred the goal was at (0, -41). Your current connection includes ['Southeast', 'Southwest', 'Northeast', 'North']. Southeast is at (1, -28), Unvisited. Southwest is at (-1, -28), Unvisited. Northeast is at (1, -26), Unvisited. North is at (0, -26), Visited. Here is your memory list in time sequence...
Planner:
Head towards (0, -48) by moving in the general direction of South when possible.
Accoding to all information above, should the plan be updated?. If yes, show the new plan. According to your plan and current connection, choose one in ['Southeast', 'Southwest', 'Northeast', 'North'] as your next action. Answer in the json format, and keys are ['Robot Thought', 'Successful Action', 'yes_or_no', 'new_plan'(if has), 'action']

A:
"Robot Thought": "Continue moving South to reach the goal at (0, -48)."
"Successful Action": "False"
"yes_or_no": "yes"
"new_plan": "Head towards (0, -48) by moving in the general direction of South when possible. Since South is not directly available, choose Southwest as the closest direction to South."
"action": "Southwest"

prompt 6: Prompt for CaP

Q:
You are now at (12, 4). The first time you inferred the goal was at (42, -42). The second time you inferred the goal was at (43, -41). The third time you inferred the goal was at (45, -41). Your current connection includes ['Southwest', 'East']. Southwest is at (11, 3), Visited. East is at (13, 4), Unvisited. Here is your memory list in time sequence... The plan is:
def search_goal():
 # step 1: Move North until an intersection because you can't directly move East
 while not get_to('intersection'):
 walk('North')
 # step 2: If you are at an intersection, you should move East because the goal is in the East
 if get_to('intersection'):
 assert can_go('East'), walk('North')
 walk('East')
 # step 3: If you are at an intersection again, you should move South because you first move North which
 # deviates from the goal in y-coordinate
 if get_to('intersection'):
 assert can_go('South'), walk('East')
 walk('South')
According to all information above, should the plan be updated?. If yes, show the new plan. According to your plan and current connection, choose one in ['Southwest', 'East'] as your next action. Answer in the json format, and keys are ['yes_or_no', 'new_plan'(if has), 'action'].

A:
"yes_or_no": "yes"
"new_plan":
"def explore_unknown():
 # step 1: Choose a direction from the current connections that leads to an unvisited location
 if can_go('\\'East\\'):
 walk('\\'East\\')
 elif can_go('\\'Northeast\\'):
 walk('\\'Northeast\\')
 elif can_go('\\'Southeast\\'):
 walk('\\'Southeast\\')
 elif can_go('\\'Southwest\\'):
 walk('\\'Southwest\\')"
"action": "walk('\\'East\\'")

prompt 7: Prompt for ProgPrompt

Q:
You are now at (25, 4). The first time you inferred the goal was at (42, -42). The second time you inferred the goal was at (43, -41). The third time you inferred the goal was at (45, -41). Your current connection includes ['West', 'East']. West is at (24, 4), Visited. East is at (26, 4), Unvisited. Here is your memory list in time sequence... The plan is:
def search_goal():
 # 1: Move North because you can't directly move East\n walk('North')
 # 2: If you are at an intersection, you should move East because the goal is in the East
 assert('at intersection')
 else: walk('North')
 walk('East')
 # 3: If you are at an intersection again, you should move South because you first move North which deviates
 # from the goal in y-coordinate
 assert('at intersection')
 else: walk('East')
 walk('South')
According to all information above, should the plan be updated?. If yes, show the new plan. According to your plan and current connection, choose one in ['West', 'East'] as your next action. Answer in the json format, and keys are ['yes_or_no', 'new_plan'(if has), 'action'].".

A:
"yes_or_no": "yes"
"new_plan":
"def search_goal():
 # Since the goal is inferred to be at (45, -41) and the current connection includes 'Southwest', 'East'
 # 'Southwest' has been visited and does not lead closer to the goal direction. 'East' moves closer in the x-
 # coordinate and is unvisited. Choose 'East' for exploration as it aligns better with the goal's direction and
 # is unvisited.
 walk('East')"
"action": "East"

prompt 8: Prompt for CoT

Q:

You are now at (-8, 0). The first time you inferred the goal was at (-19, -19). The second time you inferred the goal was at (-18, -17). The third time you inferred the goal was at (-24, -21). Your current connection includes ['East', 'West']. East is at (-7, 0), Visited. West is at (-9, 0), Visited. Here is your memory list in time sequence... The plan is: Begin exploration northward directly from the current position (-7, 0) to find potential routes leading southward towards the goal. According to all information above, should the plan be updated?. If yes, show the new plan. According to your plan and current connection, choose one in ['East', 'West'] as your next action. Answer in the json format, and keys are ['Thinking_Q1', 'yes_or_no', 'Thinking_Q2', 'new_plan'(if has), 'Thinking_Q3', 'action'].

A:

"Thinking_Q1": "Considering the current position is (-8, 0) and the previous movements have been primarily along the East-West axis without any progress towards the South, where the inferred goals have been located, the existing plan to explore northward from (-7, 0) seems ineffective. The goal locations are significantly southward, and there has been no exploration in the north-south direction from the current or nearby positions."

"yes_or_no": "Yes"

"Thinking_Q2": "A new plan should involve moving southward directly from the current position or nearby accessible points, as the goal is inferred to be in the southern direction. Since the current connections only allow movement East or West and both directions have been thoroughly explored without any progress towards the goal, it's crucial to find a way to start moving southward."

"new_plan": "Begin exploration southward directly from the current position (-8, 0) or find the nearest point that allows southward movement to approach the inferred goal locations."

"Thinking_Q3": "Given the current position at (-8, 0) and the connections available (\\"East\\" at (-7, 0) and \\"West\\" at (-9, 0)), both of which have been visited and lead to no new directions, the next action should still aim to find a new route. Since both directions are equally visited, choosing \\'East\\' or \\'West\\' does not significantly change the situation, but \\'East\\' may potentially lead back to a point that might connect southward if explored differently."

"action": "East"

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: This paper aims to address the following problem: an AI agent is provided with language descriptions of the goal location with respect to some well-known landmarks; By only observing the scene around, including recognizing landmarks and road network connections, the agent has to make decisions to navigate to the goal location without instructions. The primary contribution of this paper is the introduction of PReP, a novel agentic workflow based on LLM. improves navigation ability of the agent compared with the state-of-the-art baselines. The article provides detailed introduction to the workflow of PReP, and verifies its performance through sufficient experiments. The main claims made 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: Appendix B of the paper provides a detailed discussion of the limitations of this study. The primary issues highlighted are the overreliance on the GPT-4-turbo model, manifested as suboptimal performance on other open-source models,. Another major limitation is the limited size of the test set.

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: [NA]

Justification: [NA]

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: Sections 3- 5 provide detailed descriptions of the structure and components of PReP and the experimental setup.

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
 - If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - 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: The paper provides open access to both the data and code necessary to reproduce the main experimental results. Detailed instructions, including environment setups and execution commands, are clearly outlined in the supplemental materials. This ensures that anyone can faithfully replicate the results reported in the paper. Access to data and code is facilitated via a publicly available <https://anonymous.4open.science/r/PReP-13B5> provided in the supplementary section.

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: Sections 3 to 5 provide detailed descriptions of the experimental setup and the structure and components of PReP. Details of the experimental setup, such as the configuration of prompts, are provided in the Appendix/Supplemental Material E.

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 use student's t-test to validate our improvement in performance.

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: In the “Further Analysis” section 5.3, the “Computational Cost” subsection, as well as the “Additional Experimental Details” section, the computational resources required for completing the experiments, including the costs of both testing and fine-tuning, are discussed in detail.

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: We ensure that this work complies with the NeurIPS Code of Ethics and make ethical analysis in Appendix C.

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 the broader impact in Appendix D. The PReP agent we designed may have positive impacts for helping the blind to navigate in unknown environment and also other applications.

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:[NA]

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 have cited the original paper that produced the code package or dataset, respectively the LLaVA with Apache-2.0 and LLaMA-Factory with Apache-2.0.

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: The article introduces a new model PReP and its corresponding dataset. The relevant content can be viewed on <https://anonymous.4open.science/r/PReP-13B5>

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: No 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: No 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.