# DeliverGPT: Location-Goal Embodied Navigation for Drone Delivery with Large Pre-Trained Models

**Baining Zhao**[1,3], **Chen Gao**[1], **Zile Zhou**[1], **Yanggang Xu**[1],
**Weichen Zhang**[1,3], **Qian Zhang**[1], **Susu Xu**[2], **Jinqiang Cui**[3],
**Xinlei Chen**[1,3], **Yong Li**[1]
[1]Tsinghua University
[2]Johns Hopkins University
[3]Pengcheng Lab
chgao96@gmail.com
chen.xinlei@sz.tsinghua.edu.cn
liyong07@tsinghua.edu.cn

## Abstract

Asking a virtual robot to autonomously navigate to any location in an unexplored environment is one of the fundamental tasks in embodied artificial intelligence, which requires spatial perception and planning capabilities. A particularly demanding scenario is human-like drone delivery in cities. Recipients expect flexible delivery to a specified location based on textual instructions (e.g., "next to the south gate" or "6th-floor balcony"), rather than a fixed package collection point, aiming for a more convenient recipient experience. To fill this gap, we define this task as location-goal embodied navigation and develop a benchmark with simulator and datasets for drone delivery based on real urban environments. Then we propose a large pre-trained model-empowered agent, including four primary modules: perception, planning, motion, and memory. Specifically, within the perception module, a semantic graph is developed to integrate observations and extracting spatial semantic information. The planning module performs reasoning over the semantic graph, facilitating spatial reasoning capabilities. In the memory module, each delivery experience is catalogued to augment the agent's operational proficiency. Experimental results demonstrate that our method significantly exceeds existing solutions by 12.3% on average. Ablation analysis further reveals that the integration of the building semantic graph and memory mechanism leads to more efficient drone delivery.

## 1 Introduction

With the flourishing growth of e-commerce and online food delivery services, consumers increasingly expect rapid and responsive goods delivery [1, 2]. Drone delivery is increasingly coming into public awareness due to its flexible mobility and reduced emissions of greenhouse gases and pollutants [3, 4]. The problem to be addressed is how to deliver goods from the warehouse to the customer's hands. The current solution divides this problem into two processes while logistics companies establish fixed delivery points within a region [5]. The first process involves drones transporting goods from the warehouse to the fixed delivery point. The second process requires the recipient to collect their goods from the fixed delivery point [6]. However, recipients prefer the second process to involve as short a distance as possible. In other words, they expect drones to deliver goods as flexibly as human couriers, delivering to precise locations, such as "*Department 3302*", "*Next to the side door shelf*". Additionally, for the purpose of user-friendly human-computer interaction in delivery scenarios, it is desirable to convey location information solely through language instructions.

Preprint. Under review.

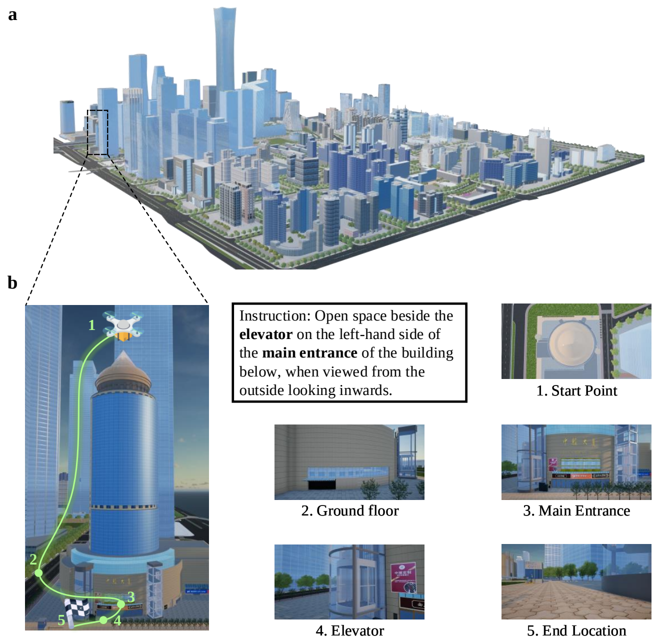



Figure 1: **a.** Our urban simulator based on Beijing, a megacity in China. Drone delivery cases distributed among different buildings are established. **b.** For each delivery case, the delivery drone is required to autonomously reach its destination based on location instructions, navigating through an unfamiliar environment using visual perception. An example trajectory of the UAV is depicted by the green line, while the destination location is represented by flags.

This can be abstracted as a location-goal embodied navigation problem: Given a description of the location, navigate to the goal description based on visual observations and urban context priors. Existing related research can be categorized into two parts: traditional drone delivery algorithms [7, 8, 9] and embodied navigation algorithm studies [10, 11]. The former focuses on how individual drones navigate from the warehouse to approximate locations based on GPS signals, as well as task allocation for multi-drone delivery. However, navigation from the vicinity of a building to a precise location cannot be achieved through GPS due to weak signal reception caused by factors such as the multipath effect [12] and signal blockage near buildings [13]. From the perspective of embodied intelligence, enabling robots to comprehend and execute tasks based on human language instructions is a crucial objective [14]. Recent advancements in large pre-trained models [15, 16] have greatly propelled research in robotics [17], including vision-language navigation [18, 19] and object-goal navigation [20]. However, the former primarily focuses on simply translating a sequence of textual action instructions into specific commands, and the latter pertains to easy indoor tasks, both of which are far away from real practical tasks.

In outdoor urban environments, robots must go beyond object-centric understanding and comprehend spatial relationships, including their location within a region, specific parts of a building object, etc. The object-goal navigation problem in urban environments should be expanded to a finer-grained location-goal navigation problem, which requires the robots to reach specified locations rather than a rough goal of reaching an object. For example, the task goal could be "near the entrance on the ground floor" or "at the center of the rooftop helipad" [21]. In summary, the delivery drone agent is required to navigate in a 3D large-scale space based on location-related instructions provided in natural language, but there remain critical challenges as follows:

- There is no existing simulator for urban environments that supports drone delivery. The simulator should be based on realistic large-scale city settings and enable UAVs to perform aerial flights. Currently, most simulators and datasets are primarily focused on indoor [22, 23] or vision-language navigation [18, 24] tasks, with a lack of simulators and datasets specifically designed for the location-goal embodied navigation task.
- The embodied agent requires fundamental capabilities in vision and language understanding, as well as a grasp of urban commonsense priors. Humans possess the ability to comprehend visual semantic

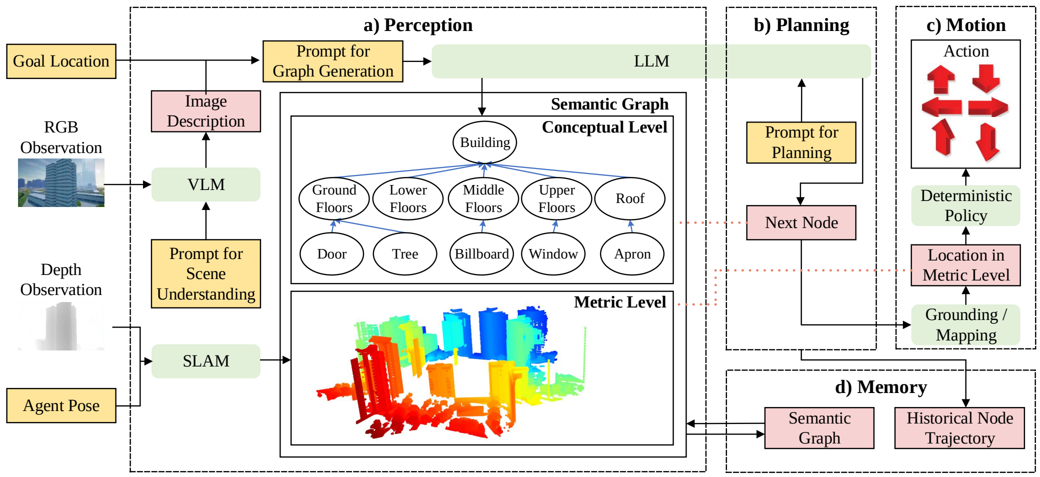



Figure 2: Overview of DeliverGPT framework, which consists of perception, planning, motion, and memory. Spatial perception capability enables the organization of historical observations, leading to the formation of a semantic graph. Spatial planning capability leverages the semantic graph to infer and determine the next node for navigation. The motion module executes the results of the planning process. The memory component records historical delivery information for next similar delivery use.

information and infer the intended location referred to by language, leveraging commonsense priors such as knowledge of building layouts and reasonable object placement. For instance, there is a higher likelihood of finding a café on the ground floor and a billboard on the middle floors. While this task is relatively easy for a human, it poses a formidable challenge for an autonomous agent.

- The agent should be capable of integrating spatial information and executing chain-like planning. In urban environments, the agent may need to cover distances that are significantly greater than those encountered in indoor settings, resulting in a lengthy decision chain. Therefore, it is necessary to identify a robust environment representation method that effectively integrates the continuous visual observations obtained during exploration. Additionally, designing an improved planning framework is crucial to enhancing decision accuracy and reducing the frequency of decisions, ultimately leading to a higher success rate.

To address these challenges, we propose a novel system called **DeliverGPT**, which focus on the location-goal navigation for drone **delivery** task with pre-trained models, specifically **G**enerative **P**re-trained **T**ransformers. We propose a simulator based on Beijing, a megacity in China, to support drone delivery. Additionally, we have developed a large pre-trained model-empowered embodied agent to address the navigation problem within urban scenarios. Recently, significant advancements have been made in the field of embodied intelligence [24, 25], where agents equipped with large language model-based planners have demonstrated remarkable capabilities [26, 24, 27]. DeliverGPT follows this trend to perform the task through a pre-trained vision-language model (**VLM**: e.g. CLIP, GPT-4-vision) and a large language model (**LLM**: e.g. GPT-4, Claude3). Specifically, this work makes the following contributions:

- To the best of our knowledge, this work is the first to investigate the location-goal embodied navigation in the urban outdoor environment. We construct the first benchmark for this important task with a simulator, tasks, datasets, and the open platform.
- We propose an large pre-trained model-empowered agent that enable UAVs to perception, planning, motion, and memory, as depicted in the Figure 2. We design a semantic graph to realize spatial information abstraction. Planning is further performed on the graph, which greatly improve the success rate of spatial chain-like reasoning.
- The result show that the simulator and method are suitable for location-goal navigation tasks in open-world urban scenarios. The proposed agent exhibits significantly higher success rates across various levels of difficulty compared to existing solutions.

## 2 Problem Formulation

The objective of the location-goal embodied navigation problem is to determine how to reach the specified precise location $p_L$, which is given through a natural language instruction $I$. The decision-making process involves following a algorithm $\pi$ to guide the agent through a sequence of observations and actions to reach the target location$p_L$. At each time step $t$, the agent obtains an observation $o_t$ that consists of RGBD images and drone's pose. The agent takes action $a_t$ based on $\pi$:

$$a_t = \pi\left(o_t, I\right) \tag{1}$$

The action $a_t$ can be formed by the arbitrary combination of eight discrete control commands: turn-left, turn-right, move-forward, move-left, move-right, move-back, move-up, and move-down.

In the urban environment, the real position $p_t$ of the agent, which can not be observed by itself, is changed based on physical dynamics rules:

$$p_t = f\left(a_t, p_{t-1}\right) \tag{2}$$

After $T$ steps, the navigation is successful if the agent stops within a Euclidean distance of $\varepsilon$ meters from the target location $p_L$.

$$\|p_T - p_L\| \leq \varepsilon \tag{3}$$

We aim for the agent to reach the target location in diverse scenarios. Assuming there are scenarios $i = 1, 2, ..., N$, the objective can be formally stated as follows:

$$\max_{\pi} \frac{1}{N} \sum_{i=1}^{N} 1\left(\left\|p_T^{(i)} - p_L^{(i)}\right\| \leq \varepsilon\right) \tag{4}$$

where $1\left(\cdot\right)$ is the indicator function, which is 1 if the condition inside is true and 0 otherwise.

## 3 Benchmark Construction in City Environment

The benchmark consists of three components: a 3D urban simulator, a dataset specifically designed for the location-goal embodied navigation task in drone delivery and evaluations.

**Urban Simulator:** In our benchmark, we first construct an urban simulator to provide $o_t$ in Eq. 1 and $f\left(\cdot\right)$ in Eq. 2. As presented in Figure 1(a), the simulator is built upon the layout and architectural structures of Beijing, which is a prominent area within China's capital city. Using Unreal Engine 5 [28] and Microsoft AirSim plugins [29], we have achieved continuous simulation and nearly realistic rendering, in contrast to existing simulators that primarily concentrate on indoor [30, 31, 32, 33], discrete [34] or hypothetical urban settings [35, 18] The proposed simulator encompasses urban infrastructures such as roads, office building clusters, residential areas, and greenery. Additionally, the architectural details, such as windows, entrances, billboards, utility boxes, etc., are also included.

**Embodied Drone Delivery Dataset:** There is currently no dedicated dataset available for location-goal navigation for drone delivery in urban environments. Based on the characteristics of the buildings and surrounding urban facilities in the simulator, $N = 112$ delivery trajectories were established. As shown in Figure 1(b), each delivery trajectory $i$ consists of a starting point $p_0^{(i)}$, an endpoint $p_L^{(i)}$ along with the corresponding textual instruction, and the ground truth route. The routes are obtained by manually controlling the drones. Through the construction of these realistic scenarios, we can evaluate agent's ability to navigate within open-world urban environments and accurately locate delivery destinations.

The drone can only plan and navigate through embodied sensing. This requires the agent to first understand the mapping between location instructions and the spatial environment. During the movement process, the agent continuously accumulates perception of the environment and utilizes reasoning abilities to plan actions until the target location is reached.

**Metrics:** We utilize three standard metrics to evaluate the location-goal embodied navigation: Success Rate (SR), Success Weighted by Path Length (SPL), and Distance to Goal (DTG) [36, 22, 27]. SR indicates the proportion of delivery episodes where the agent successfully reaches the target location within a 10-meter margin of error. Similar to Eq. 4, it is calculated using $\mathrm{SR} = \frac{1}{N}\sum_{i=1}^{N} s_i$, where $N$ is the number of delivery episodes and $s_i$ represents the success of the $i$-th delivery, where it takes a value of 1 for success and 0 for failure. As a metric that considers both navigation precision and efficiency, SPL comprehensively takes into account the SR and the corresponding ratio of the optimal path length $l_i$ to the actual delivery path length $g_i$. The calculation formula is represented as $\mathrm{SPL} = \frac{1}{N}\sum_{i=1}^{N} s_i \frac{l_i}{\max(l_i,g_i)}$. DTG is computed by $\mathrm{DTL} = \frac{1}{N}\sum_{i=1}^{N} d_i$, where $d_i$ denotes average distance from the agent's final location to the destination.

# 4 Methodology

To address the mentioned above challenges, we propose an embodied agent named DeliverGPT, presented in Fig. 2. We first construct agent's comprehensive perception of the environment (Section 4.1). We then develop spatial planning capabilities for the agent (Section 4.2) and convert the planning results into drone motion (Section 4.3). Additionally, we design the memory module to assist the agent in spatial perception and planning for similar delivery tasks (Section 4.4).

## 4.1 Spatial Perception

Spatial perception forms the foundation of planning, particularly in large-scale urban environments where semantic information is sparse, which requires integrating historical observations and extracting spatial semantic information. To address this, we develop a semantic graph to facilitate this process, composed of a textual conceptual level $C_t$ and a metric level $M_t$. The conceptual level facilitates understanding and reasoning for the LLM, while the metric level facilitates navigation for the UAV. The conceptual level of the 3D Semantic Graph Generation is a layered graph that encompasses spatial concepts (nodes, denoted as $N_{j,t}$) at multiple levels of abstraction, along with their corresponding relationships (edges, denoted as $E_{k,t}$). The edges of the graph illuminate the topological relationships among semantic concepts across various nodes. This representation has recently emerged as a notable high-level abstraction for capturing 3D urban buildings or scenes using drones.

The generation and iterative process of the semantic graph are as follows. Suppose the observation $o_t$ at time $t$ consists of RGB $v_t$, Depth $d_t$, and pose $m_t$. VLM possesses the vision-language understanding capability for a single image. Employing a prompt of scene understanding $P_{scene}$ (Figure 6a), the VLM can describe the building elements in the RGB image in textual form $b_t$:

$$b_t = \mathrm{VLM}(v_t, P_{scene}) \tag{5}$$

Then the conceptual level of semantic graph $C_t = \{(N_{j,t}, E_{k,t})\}$ is derived by:

$$C_t = \mathrm{LLM}(b_t, C_{t-1}, P_{graph}) \tag{6}$$

where the LLM is continuously employed to update the graph taking the image description $b_t$, previous conceptual part $C_{t-1}$, and graph generation prompt $P_{graph}$ (Figure 6b) as input. For the metric level, we obtain it through the SLAM algorithm, which leverages depth and pose sensors carried by the UAV to perceive surrounding structures, roads, and other environmental features. By continuously observing the environment and estimating the robot's position, SLAM algorithms can simultaneously construct an accurate map (Appendix A.2).

$$M_t = \mathrm{SLAM}(d_t, m_t) \tag{7}$$

Additionally, the building semantic graph can be initially established by leveraging prior knowledge of urban architectural structure. For instance, it is well-established that a building's main entrance is typically situated on the ground floor, while the rooftop often features an apron.

## 4.2 Spatial Planning

Spatial planning capability refers to the agent's ability to determine "where to go" in a three-dimensional open world. Through dynamic perception and a series of decisions, the agent gradually approaches the goal location. We combine the conceptual level of the semantic graph with the commonsense reasoning ability of LLM to construct this capability.

We emphasize that instead of directly generating drone actions, this module focuses on high-level planning: determining which node within the conceptual level of the semantic graph to navigate to next. This effectively harnesses the commonsense reasoning capabilities of LLM while mitigating its limitations in embodied capabilities. The layered structure of the conceptual level can be easily converted into textual form and understood by the LLM. The thought process behind the prompt for planning is as follows: first, clarifying the current node where the agent is located and emphasizing the target to be found; second, finding a node in the next layer of the current node, which either belongs to the final target location or is close in proximity to the final location, as shown in Figure 6(c) in Appendix. This effectively leverages the commonsense reasoning ability of LLM to its full potential. For instance, when the drone needs to navigate to "underneath the red tree next to the building", even if the red tree is not yet observed in its field of view, LLM can still infer the next step to proceed towards the node "ground floors" based on the reasoning that "trees are mostly found on ground floors."

$$N_{i,t}^{\text{goal}} = \text{LLM}(C_t, P_{plan}) \tag{8}$$

### 4.3 Motion: Spatial Mobility Capability

After determining the next node to navigate to, we need to address how the drone moves. This process can be divided into three steps: grounding, mapping, and action generation. Grounding involves approximating the position of the selected node within our RGB observation. Subsequently, by incorporating depth and pose information, we obtain the coordinate point $(x_t, y_t, z_t)$ of that position on the metric level in the semantic graph.

$$N_{i,t}^{\text{goal}} \to (x_t, y_t, z_t) \sim M_t \tag{9}$$

After deriving the exact coordinate on the metric level of semantic map, the delivery agent is equipped with a deterministic policy $G(\cdot)$ to help it navigate to the goal. The full motion algorithm is in Appendix A.3.

$$a_t = G\left((x_t, y_t, z_t), M_t\right) \tag{10}$$

### 4.4 Short-Term and Long-Term Memory

**Short-term memory:** For each navigation, in addition to the semantic graph, historical decisions generated by the LLM are recorded to facilitate the next reasoning step.

**Long-term memory:** Equipped with a memory of historical deliveries, the agent acquires knowledge of the environment surrounding the delivery location and gains experience in spatial planning. Leveraging this prior experience enables agent to execute tasks with greater efficiency, mirroring the enhancement of human skills through repetitive practice. We have devised a distributed memory mechanism within urban spaces that facilitates the storage and retrieval of acquired knowledge, as shown in Fig. 3. This encompasses the construction of semantic maps and the documentation of historical delivery trajectories. Contrary to reinforcement learning-based models, which encapsulate knowledge within model parameters, our approach renders the delivery knowledge explicit, logical, and more closely resonant with human cognitive processes. Upon the drone's approach to the delivery location, the relevant memory pertaining to that location is activated. This obviates the need to reconstruct the semantic graph from scratch based on current observations with each delivery attempt. Instead, the pre-remembered semantic graph serves as a more comprehensive and precise initial condition, derived from multiple delivery iterations. This not only provides the agent with a fundamental comprehension of the delivery environment but also enhances the efficiency and accuracy of the delivery process. Moreover, when tasked with delivering to a novel destination, the agent may initially necessitate multiple exploratory attempts to ascertain the precise location. However, upon subsequent deliveries, leveraging accrued experience, the agent is able to expedite the process by navigating directly to the delivery site via an optimized sequence of nodes.

## 5 Experiment

### 5.1 Experimental Setup

**Implementation Details:** To explore the maximum potential of the DeliverGPT framework, we employ advanced large pre-trained models, *gpt-4-vision-preview* [37] for the VLM and *gpt-3.5-turbo* [38] for the LLM. The parameters for capturing RGBD observations and pose from the drone are set to the default configuration in the AirSim platform [29].

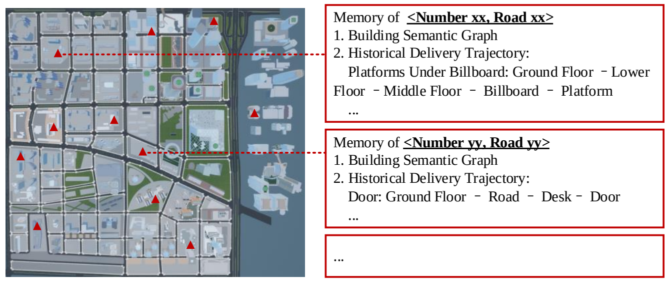



Figure 3: Within the memory module, long-term memories are stored in a city map. Upon reaching the vicinity of a designated building, the building's semantic graph and historical delivery trajectories that are close to the current delivery destination point are accessed.

**Baselines:** The compared methods include **Random**, **Action Sampling**, **AG-GPT4** [39], **NavGPT** [27], **CoW** [40], and **SayNav** [33][1]. For location-goal navigation in the drone delivery task, no direct solutions are available for direct comparison. Thus, we not only construct three common benchmarks for comparison purposes [18], but we also we adapt existing vision-language navigation approaches [27] and indoor object navigation [40, 33] to our scenario.

### 5.2 Comparison Results

To ensure fairness, considering the absence of long-term memory capability in other comparative algorithms, we exclusively compare their performance in the most challenging zero-shot scenarios. All delivery cases were categorized into three difficulty levels: easy, normal, and hard, corresponding to navigation distances of 0-100m, 100m-200m, and >200m, respectively. The results, presented in Table 1, lead us to the following three observations.

- **The action space in urban simulator is large and suitable for embodied navigation.** Both the random and action sample methods exhibit SR and SPL scores close to 0. This indicates that the proposed urban simulator encompasses a vast action space, providing support for the validation of location-goal navigation in the drone delivery task. The agent fails to reach or even approach the destination without understanding the instructions, visual perceptions, and their alignment.
- **The direct application of large pre-trained models for generating drone control commands yields unsatisfactory performance.** The success rate of AG-GPT4 is also below 6%, and its DTG is only slightly higher than that of the Action Sampling method. This indicates that existing multimodal large-scale models, while possessing fundamental vision-language understanding and commonsense reasoning capabilities, are still incapable of handling embodied tasks in 3D open-world environments.
- **Spatial perception and planning capabilities are crucial for embodied navigation.** DeliverGPT outperforms other methods across various difficulty scenarios, especially in the hard scenario where it achieves SR and SPL scores that are more than **twice** those of the other methods. Additionally, its DTG is significantly lower than that of other methods, indicating that the drone's final position is close to the goal location. This highlights the importance of the agent's spatial perception and spatial planning abilities in outdoor urban conditions, emphasizing the significance of these capabilities compared to relevant embodied methods in indoor settings.

### 5.3 Ablation Study

To evaluate the effectiveness of the semantic graph and long-term memory module, we conduct an ablation study by either removing each component or substituting it with a simplified module. The semantic graph is the key component of the proposed agent's spatial perception and spatial planning capabilities. To simulate the randomness and repeatability of drone delivery, we conduct 1,000 Monte Carlo sampling iterations with replacement for all cases, and the sampled cases are used for experiments. The outcomes of this study are presented in Table 2.

[1]The details of these baselines are introduced in Section A.4 of Appendix.

Table 1: Results of zero-shot location-goal embodied navigation.

| Method | Easy | | | Normal | | | Hard | | |
|---|---|---|---|---|---|---|---|---|---|
| | SR/% ↑ | SPL/% ↑ | DTG/m ↓ | SR/% ↑ | SPL/% ↑ | DTG/m ↓ | SR/% ↑ | SPL/% ↑ | DTG/m ↓ |
| Random | 0 | 0 | 80.2 | 0 | 0 | 146.1 | 0 | 0 | 227.9 |
| Action Sampling | 0.2 | 0.1 | 209.7 | 0.1 | 0 | 305.5 | 0 | 0 | 389.4 |
| AG-GPT4 | 5.5 | 3.2 | 112.1 | 2.6 | 1.7 | 180.4 | 1.4 | 0.8 | 297.1 |
| NavGPT | 17.8 | 10.6 | 57.4 | 12.3 | 7.4 | 109.9 | 7.0 | 5.8 | 231.5 |
| CoW | 9.0 | 5.9 | 72.6 | 5.8 | 3.1 | 132.8 | 3.5 | 2.4 | 207.7 |
| SayNav | 25.9 | 19.7 | 55.0 | 19.7 | 15.3 | 90.2 | 10.8 | 7.4 | 183.5 |
| **DeliverGPT** | **41.4** | **33.7** | **45.2** | **35.9** | **30.6** | **72.3** | **23.1** | **16.8** | **130.3** |

Table 2: Results of Ablation Study

| DeliverGPT Ablation | | Evaluation Metrics | | |
|---|---|---|---|---|
| Semantic Graph | Long-Term Memory | SR/% ↑ | SPL/% ↑ | DTG/m ↓ |
| ✗ | ✗ | 8.5 | 6.0 | 148.4 |
| ✗ | ✓ | 10.7 | 9.3 | 120.1 |
| ✓ | ✗ | 32.6 | 27.2 | 89.4 |
| ✓ | ✓ | 40.9 | 34.6 | 75.7 |

**Effect of semantic graph**. In the absence of the building semantic graph, the agent directly selects an object or location within the field of view that most likely approximates the precise delivery location. This results in a 24.1%+ and 21.2%+ decrease in SR and SPL, respectively, and an 58.7%+ increase in DTG. Analysis of the reasoning process via the LMM reveals that the semantic graph enables the agent to comprehend the spatial relationships between the exact delivery location, buildings, and its own location, thereby enhancing the efficiency and success rate of locating the delivery point. In the semantic map, the conceptual level represents spatial or object-related information, while the metric map records the robot's metric measurements. The conceptual map synthesizes environmental information, forming the basis of spatial perception. When combined with the commonsense reasoning abilities of the LLM (Language and Vision Model), the conceptual level contributes to the agent's spatial planning capabilities. The metric map supports the agent's motion ability. These three abilities collectively determine the performance of location-goal embodied navigation.

Figure 4 provides a case study, showcasing the step-by-step process and outcomes of perception and planning. Without the semantic graph, the navigation sequence is prone to errors. However, with the inclusion of the semantic graph, the agent strictly follows spatial relationships and successfully reaches the precise locations. For comparison, we omit the long-term memory module, rendering the system incapable of benefiting from historical delivery experiences, as shown in Table 2. For the agent with the semantic graph, long-term memory results in an 8.2% improvement in SR and a 7.4% improvement in SPL, along with an average reduction of 13.7m in DTG.

**Effect of long-term memory**. As depicted in Figure 5, with an increase in repeated memory times, both SR and SPL exhibit a gradual increase, while DTG shows a progressive decrease. The enhancement in SR stems from referencing successful navigation experiences, where historical successful node select series provide valuable insights during planning, mitigating the instability observed in LLM output to some extent. In addition to the increased SR, SPL improvement is attributed to minimizing unnecessary detours. For example, a recorded trajectory includes the node sequence: ground floors - yellow tree - door - middle floors - billboard - ground floors - ground. The deduced optimal sequence for this trajectory is established as: middle floors - billboard - ground floors - ground. This memory mechanism enhances the delivery efficiency across similar tasks.

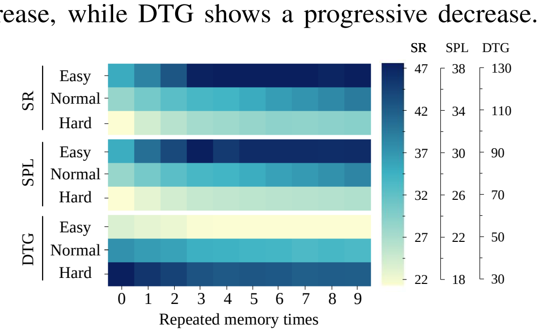



Figure 5: The variations in location-goal embodied navigation performance when repeatedly delivering to similar locations. More challenging tasks exhibit greater improvement.

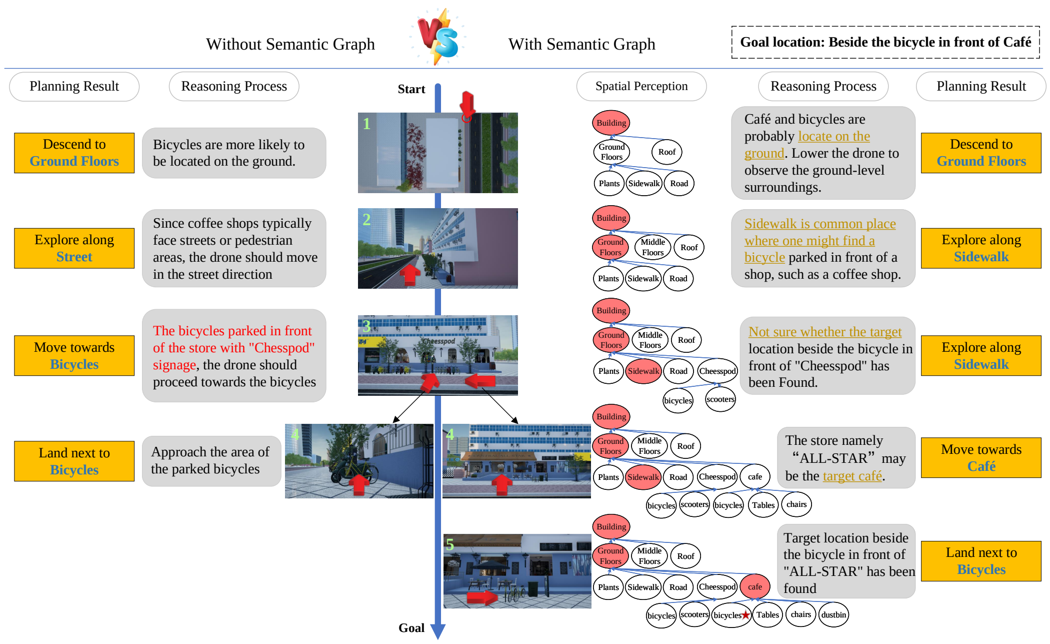



Figure 4: The comparative case analysis of the step-by-step reasoning process and planning decisions between the agent without the use of the semantic graph and that with the semantic graph. In the absence of a semantic graph, the agent mistakenly identified bicycles. Comparatively, the presence of a semantic graph structured the spatial relationships for the agent, resulting in an improvement in navigation accuracy.

## 6 Related Work

**Large Pre-Trained Models in Embodied Intelligence.** The application of large pre-trained models in robotics has sparked widespread research interest due to its ability to enable robots to communicate and think like humans [39, 14]. Current research focuses on leveraging vision-language understanding and commonsense reasoning abilities of large pre-trained models to handle tasks like navigation [10, 26], manipulation [41], task planning [42, 43], and human-robot interaction [44]. In this work, we explore large pre-trained models to construct embodied agent for location-goal navigation.

**Vision-Language Navigation.** In this task, the agent must comprehend navigational instructions presented in text format (e.g., "Go towards a park bench, take a left at the stop sign, then stop at the trunk"), and subsequently integrate visual observations to execute the corresponding actions [11]. Research [26, 24, 27] utilize LLMs to extract landmarks and actions from the instructions. LLMs then proceed with sequence planning to determine the next action for the robot. This task bears some resemblance to location-goal navigation, as both involve the comprehension of textual instructions and the integration of visual information for navigation.

**Zero-shot Object Navigation.** This task is to navigate an agent to a specific goal object within an unknown environment [40, 45, 20]. Prior works [45, 40] recognize the object goal based on contrastive language-image pre-training model (CLIP) [46]. In [33], LLMs directly generates high-dimensional action instructions, which are then combined with object grounding algorithms to achieve navigation. Existing studies primarily focus on indoor environments for ground robots, while the location-goal embodied navigation for drone delivery is concerned with 3D urban environments. The proposed task places an emphasis on understanding spatial relationships, such as "**mid-levels** of buildings" or "entrances **adjacent to** convenience stores."

## 7 Conclusion

We introduce the location-goal embodied task for drone delivery and construct a benchmark including the simulator, dataset, and platform. Furthermore, we propose a large pre-trained model-empowered agent, which builds its spatial perception and planning capabilities around a semantic graph tailored for large-scale urban environments. The long-term memory mechanism simulates the human skill acquisition process, emphasizing that practice leads to proficiency. The experiment results illustrate the effectiveness of our simulator and method from different perspectives.

## 8 Limitation and Future Work

Limitations of our method include the large size of VLM and LLM, which poses challenges for deployment on actual drones, and the need for further improvements in delivery success rate for practical application. Despite these challenges, our findings provide valuable insights into the development of embodied robotics using large pre-trained models, particularly in granting them spatial perception and planning capabilities. We also plan to test the method in real-world UAVs.

## References


[1] Yi Ma, Xiaotian Hao, Jianye Hao, Jiawen Lu, Xing Liu, Tong Xialiang, Mingxuan Yuan, Zhigang Li, Jie Tang, and Zhaopeng Meng. A hierarchical reinforcement learning based optimization framework for large-scale dynamic pickup and delivery problems. *Advances in neural information processing systems*, 34:23609–23620, 2021.

[2] Riccardo Mangiaracina, Alessandro Perego, Arianna Seghezzi, and Angela Tumino. Innovative solutions to increase last-mile delivery efficiency in b2c e-commerce: a literature review. *International Journal of Physical Distribution & Logistics Management*, 49(9):901–920, 2019.

[3] Hojoon David Yoo and Stanislav M Chankov. Drone-delivery using autonomous mobility: An innovative approach to future last-mile delivery problems. In *2018 ieee international conference on industrial engineering and engineering management (ieem)*, pages 1216–1220. IEEE, 2018.

[4] Kuan-Wen Chen, Ming-Ru Xie, Yu-Min Chen, Ting-Tsan Chu, and Yi-Bing Lin. Dronetalk: An internet-of-things-based drone system for last-mile drone delivery. *IEEE Transactions on Intelligent Transportation Systems*, 23(9):15204–15217, 2022.

[5] Hasan Yetis and Mehmet Karakose. A new smart cargo cabinet application for unmanned delivery in smart cities. In *2018 International Conference on Artificial Intelligence and Data Processing (IDAP)*, pages 1–5. IEEE, 2018.

[6] Gino Brunner, Bence Szebedy, Simon Tanner, and Roger Wattenhofer. The urban last mile problem: Autonomous drone delivery to your balcony. In *2019 international conference on unmanned aircraft systems (icuas)*, pages 1005–1012. IEEE, 2019.

[7] Kevin Dorling, Jordan Heinrichs, Geoffrey G Messier, and Sebastian Magierowski. Vehicle routing problems for drone delivery. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(1):70–85, 2016.

[8] Aurello Patrik, Gaudi Utama, Alexander Agung Santoso Gunawan, Andry Chowanda, Jarot S Suroso, Rizatus Shofiyanti, and Widodo Budiharto. Gnss-based navigation systems of autonomous drone for delivering items. *Journal of Big Data*, 6:1–14, 2019.

[9] Victor RF Miranda, Adriano MC Rezende, Thiago L Rocha, Héctor Azpúrua, Luciano CA Pimenta, and Gustavo M Freitas. Autonomous navigation system for a delivery drone. *Journal of Control, Automation and Electrical Systems*, 33:141–155, 2022.

[10] Jinzhou Lin, Han Gao, Rongtao Xu, Changwei Wang, Li Guo, and Shibiao Xu. The development of llms for embodied navigation. *arXiv preprint arXiv:2311.00530*, 2023.

[11] 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.

[12] Franco Fuschini, Hassan El-Sallabi, Vittorio Degli-Esposti, Lasse Vuokko, Doriana Guiducci, and Pertti Vainikainen. Analysis of multipath propagation in urban environment through multidimensional measurements and advanced ray tracing simulation. *IEEE Transactions on Antennas and Propagation*, 56(3):848–857, 2008.

[13] Phuc Nguyen, Hoang Truong, Mahesh Ravindranathan, Anh Nguyen, Richard Han, and Tam Vu. Matthan: Drone presence detection by identifying physical signatures in the drone's rf communication. In *Proceedings of the 15th annual international conference on mobile systems, applications, and services*, pages 211–224, 2017.

[14] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

[15] Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heintz, and Dan Roth. Recent advances in natural language processing via large pre-trained language models: A survey. *ACM Computing Surveys*, 56(2):1–40, 2023.

[16] Haoyi Duan, Yan Xia, Zhou Mingze, Li Tang, Jieming Zhu, and Zhou Zhao. Cross-modal prompts: Adapting large pre-trained models for audio-visual downstream tasks. *Advances in Neural Information Processing Systems*, 36, 2024.

[17] Fanlong Zeng, Wensheng Gan, Yongheng Wang, Ning Liu, and Philip S Yu. Large language models for robotics: A survey. *arXiv preprint arXiv:2311.07226*, 2023.

[18] 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.

[19] Peihao Chen, Xinyu Sun, Hongyan Zhi, Runhao Zeng, Thomas H Li, Gaowen Liu, Mingkui Tan, and Chuang Gan. A2 nav: Action-aware zero-shot robot navigation by exploiting vision-and-language ability of foundation models. *arXiv preprint arXiv:2308.07997*, 2023.

[20] 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.

[21] Yile Liang, Haocheng Luo, Haining Duan, Donghui Li, Hongsen Liao, Jie Feng, Jiuxia Zhao, Hao Ren, Xuetao Ding, Ying Cha, et al. Meituan's real-time intelligent dispatching algorithms build the world's largest minute-level delivery network. *INFORMS Journal on Applied Analytics*, 54(1):84–101, 2024.

[22] Bangguo Yu, Hamidreza Kasaei, and Ming Cao. L3mvn: Leveraging large language models for visual target navigation. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3554–3560. IEEE, 2023.

[23] Karmesh Yadav, Arjun Majumdar, Ram Ramrakhya, Naoki Yokoyama, Alexei Baevski, Zsolt Kira, Oleksandr Maksymets, and Dhruv Batra. Ovrl-v2: A simple state-of-art baseline for imagenav and objectnav. *arXiv preprint arXiv:2303.07798*, 2023.

[24] 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.

[25] Jiangyong Huang, Silong Yong, Xiaojian Ma, Xiongkun Linghu, Puhao Li, Yan Wang, Qing Li, Song-Chun Zhu, Baoxiong Jia, and Siyuan Huang. An embodied generalist agent in 3d world. *arXiv preprint arXiv:2311.12871*, 2023.

[26] 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.

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

[28] Epic Games. Unreal engine 5. `https://www.unrealengine.com/en-US/unreal-engine-5`, 2022.

[29] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics*, 2017.

[30] 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.

[31] 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.

[32] 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.

[33] Abhinav Rajvanshi, Karan Sikka, Xiao Lin, Bhoram Lee, Han-Pang Chiu, and Alvaro Velasquez. Saynav: Grounding large language models for dynamic planning to navigation in new environments. *arXiv preprint arXiv:2309.04077*, 2023.

[34] 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.

[35] Yue Fan, Winson Chen, Tongzhou Jiang, Chun Zhou, Yi Zhang, and Xin Eric Wang. Aerial vision-and-dialog navigation. *arXiv preprint arXiv:2205.12219*, 2022.

[36] 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.

[37] 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.

[38] OpenAI. Gpt api models overview. `https://platform.openai.com/docs/models/gpt-3-5-turbo`, 2021.

[39] Sai Vemprala, Rogerio Bonatti, Arthur Bucker, and Ashish Kapoor. Chatgpt for robotics: Design principles and model abilities. Technical Report MSR-TR-2023-8, Microsoft, February 2023.

[40] 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.

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

[42] Zirui Zhao, Wee Sun Lee, and David Hsu. Large language models as commonsense knowledge for large-scale task planning. *Advances in Neural Information Processing Systems*, 36, 2024.

[43] 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.

[44] Hong Qiao, Ya-Xiong Wu, Shan-Lin Zhong, Pei-Jie Yin, and Jia-Hao Chen. Brain-inspired intelligent robotics: Theoretical analysis and systematic application. *Machine Intelligence Research*, 20(1):1–18, 2023.

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

[46] 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.

[47] 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.

[48] Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics*, 33(5):1255–1262, 2017.

[49] Carlos Campos, Richard Elvira, Juan J Gómez Rodríguez, José MM Montiel, and Juan D Tardós. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021.

[50] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

# A Appendix

## A.1 Details of Prompts

We provide an illustration of the used prompts in Figure 6.

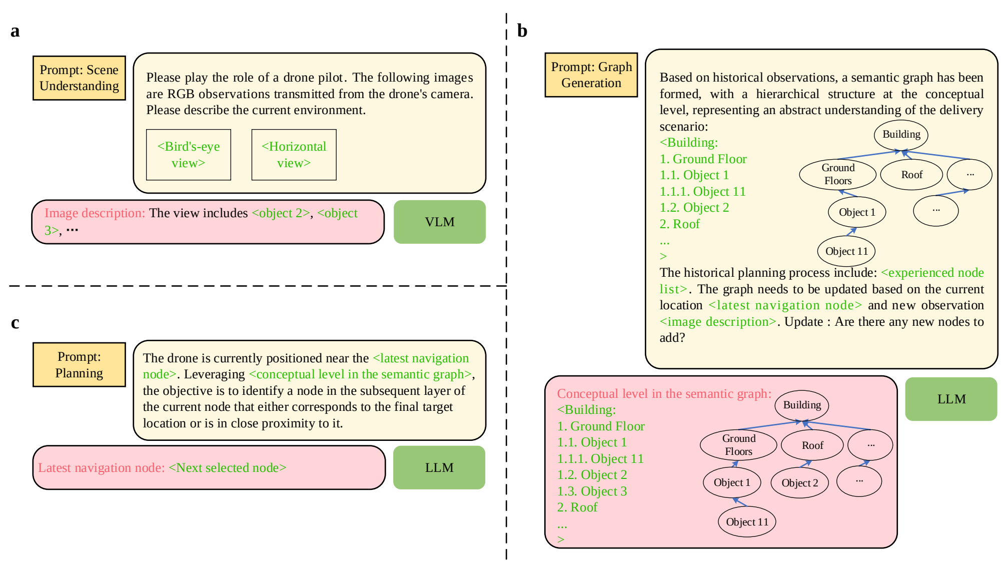



Figure 6: Prompt used for **a.** scene understanding with VLM; **b.** graph generation with LLM; **c.** planning with LLM.

## A.2 Details of SLAM

Simultaneous Localization and Mapping (SLAM) is a fundamental algorithm in the fields of robotics and autonomous systems, which has a wide range of applications in autonomous vehicles, drones, and service robots [47]. It enables drones to navigate an unknown environment autonomously by concurrently constructing a map of the surroundings and determining its own location within this map. It is particularly effective in environments where GPS signals are weak or absent, such as indoors or densely built urban areas.

The process of visual SLAM in drones involves several key steps, each of which contributes to the accurate mapping and localization capabilities of the system. The first step in the SLAM process is acquiring data from depth cameras to capture visual information about the environment, which will be processed to extract meaningful features for mapping and localization.

The next step in SLAM is to extract distinctive features from the captured depth images. ORB (Oriented FAST and Rotated BRIEF) [48] [49] detects keypoints in the depth images and computes descriptors for each keypoint, which are then used to match features between successive frames. Mathematically, let $d_t$ represent the image captured at time $t$. The set of keypoints $\{\mathbf{p}_i\}_{i=1}^N$ in the image can be extracted using a feature detector:

$$\{\mathbf{p}_i\}_{i=1}^N = \mathrm{ORB}(\mathrm{d_t}) \tag{11}$$

where $\mathbf{p}_i$ denotes the $i$-th keypoint.

Then with the matched keypoints between consecutive frames, the relative motion of the drone can be estimated. Given a set of 3D points $\mathbf{P}_i$ and their corresponding 2D projections $\mathbf{p}_i$ in the image, the goal is to estimate the camera's rotation $\mathbf{R}$ and translation $\mathbf{t}$ such that:

$$\mathbf{p}_i \approx \mathbf{K}[\mathbf{R}|\mathbf{t}]\mathbf{P}_i \tag{12}$$

where $\mathbf{K}$ is the camera intrinsic matrix.

Finally, as the drone navigates, the map of the environment is gradually built and updated. Loop closure detection is used to recognize when the drone revisits a previously mapped area, which helps in correcting drift and improving the accuracy of the map. With the help of pose graph optimization, the entire map is well adjusted to ensure consistency when a loop closure is detected. The poses of the drone are represented as nodes in a graph, and the edges represent the relative transformations between these poses. The optimization problem can be expressed as:

$$\min_{\mathbf{x}} \sum_{i,j} \|\mathbf{z}_{ij} - h(\mathbf{x}_i, \mathbf{x}_j)\|^2_{\mathbf{\Omega}_{ij}} \tag{13}$$

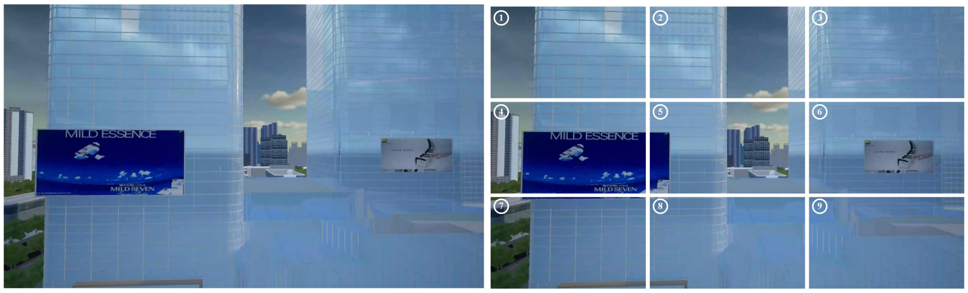
where $\mathbf{x}$ represents the poses, $\mathbf{z}_{ij}$ is the measured transformation between poses $i$ and $j$, and $h(\mathbf{x}_i, \mathbf{x}_j)$ is the predicted transformation based on the current pose estimates.

### A.3 Details of Grounding / Mapping in Motion

After determining the next node to proceed to, there are multiple ways to determine its position in the RGB image. Existing Vision-Language Models (VLM) such as Qwen [50] can directly achieve grounding, providing the coordinates of the grounding box. Here, we present a coarse grounding approach as illustrated in the figure. We divide the field of view into nine sub-images and leverage the vision-language understanding capabilities of VLM to determine the most relevant sub-image. Once selected, we obtain the pixel coordinates of the center point of that sub-image. By combining the correspondence between the RGB and Depth cameras, we can obtain the coordinates of that pixel on the metric map. We move forward a certain distance along that coordinate direction and repeat the above process until we approach the final target coordinates.

### A.4 Details of Baselines

- **Random**. The drone randomly selects actions (e.g., forward, backward, upward, downward, left, and right) at each location until it meets the maximum number of steps or reaches the goal location. This approach effectively showcases the solution space's magnitude.
- **Action Sampling**. Action Sampling agents employ action sampling based on the action distribution derived from the proposed urban drone delivery dataset in the simulator, thus emphasizing the role of spatial chain planning.
- **Action Generation with GPT-4 (AG-GPT4)** [39]. As one of the most powerful multimodal large models, *gpt-4-vision-preview* can continuously receive instructions and RGBD observations as inputs and generates discrete drone commands. This approach is used to showcase the performance of directly applying multimodal large models and evaluate their embodied capabilities.




**Instruction:** From the current perspective, does it include a blue billboard? I have divided the complete image into nine sub-images. If it is present, in which sub-images or which specific sub-image is it located?

Figure 7: After determining the present selected node, the current perspective is divided into nine sub-images. The large model further identifies the sub-image in which the object is located, enabling rough grounding. By incorporating depth imaging to estimate its distance, multiple iterations are performed to eventually reach the vicinity of the target object.

- **NavGPT** [27] is a purely LLM-based instruction following navigation agent by performing zero-shot sequential action prediction for navigation. This method primarily aims to show the performance of vision-language navigation algorithms when transferred to the location-goal problem.
- **CLIP on Wheels (CoW)** [40] uses a gradient-based visualization technique on CLIP to localize the goal object in the egocentric view and employs a frontier-based exploration technique for zero-shot object goal navigation.
- **SayNav** [33] employs a novel grounding mechanism to build a 3D scene graph of the explored environment. This graph is used as input to LLMs to generate high-level navigation plans, which are then executed by a pre-trained low-level planner. CoW and SayNav demonstrate the performance of indoor object navigation methods.

## 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.