CorNav: Autonomous Agent with Self-Corrected Planning for Zero-Shot Vision-and-Language Navigation

Xiwen Liang[1*], **Liang Ma**[1*], **Shanshan Guo**[2], **Jianhua Han**[3], **Hang Xu**[3], **Shikui Ma**[4], **Xiaodan Liang**[1,5†]
[1]Shenzhen Campus of Sun Yat-sen University, [2]Northeastern University, China,
[3]Huawei Noah's Ark Lab, [4]Dataa Robotics, [5]Peng Cheng Laboratory
https://mligg23.github.io/CorNav-Site

Abstract

Understanding and following natural language instructions while navigating through complex, real-world environments poses a significant challenge for general-purpose robots. These environments often include obstacles and pedestrians, making it essential for autonomous agents to possess the capability of self-corrected planning to adjust their actions based on feedback from the surroundings. However, the majority of existing vision-and-language navigation (VLN) methods primarily operate in less realistic simulator settings and do not incorporate environmental feedback into their decision-making processes. To address this gap, we introduce a novel zero-shot framework called CorNav, utilizing a large language model for decision-making and comprising two key components: 1) incorporating environmental feedback for refining future plans and adjusting its actions, and 2) multiple domain experts for parsing instructions, scene understanding, and refining predicted actions. In addition to the framework, we develop a 3D simulator that renders realistic scenarios using Unreal Engine 5. To evaluate the effectiveness and generalization of navigation agents in a zero-shot multi-task setting, we create a benchmark called NavBench. Our empirical study involves deploying 7 baselines across four tasks, i.e., goal-conditioned navigation given a specific object category, goal-conditioned navigation given simple instructions, finding abstract objects based on high-level instructions, and step-by-step instruction following. Extensive experiments demonstrate that CorNav consistently outperforms all baselines by a significant margin across all tasks.

1 Introduction

Language-driven navigation is a critical skill for robot assistants when it comes to performing a wide range of real-world tasks. Most autonomous agents are trained using predefined datasets and tasks, and perform well in familiar environments. However, the real world is filled with a multitude of objects and scenes, making it challenging to train an agent that can generalize effectively. Recently, large language models (LLMs) (Chowdhery et al., 2022; OpenAI, 2023; Touvron et al., 2023; Chiang et al., 2023; Geng et al., 2023) have demonstrated remarkable effectiveness across various tasks and have emerged as versatile autonomous agents capable of informed decision-making (Sun et al., 2023; Wang et al., 2023; Sumers et al., 2023). These LLMs are pre-trained on massive textual data, endowing them with extensive commonsense knowledge that proves invaluable for navigation tasks. For instance, they can infer that a stove is likely to be found in the kitchen and that a bed is typically located in a bedroom.

The success of GPT has highlighted the efficacy of utilizing human instructions for zero-shot navi-

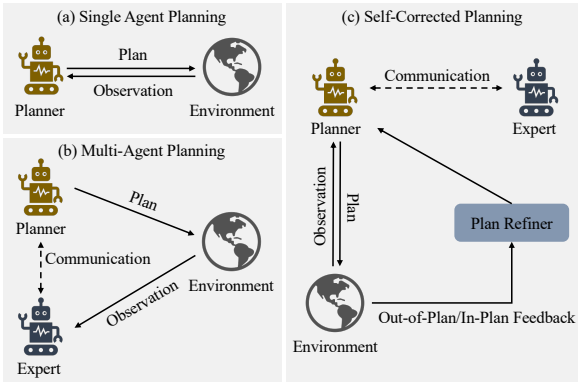

Figure 1: Comparison with existing VLN agents. (a) The single agent planning paradigm requires the agent to analyse and make decisions by itself. (b) Multi-agent planning paradigm enables the agent to communicate with multiple experts and perform complex reasoning. (c) Our self-corrected planning considers in-plan or out-of-plan feedback from a near-realistic environment.

*Equal contribution.
†Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2024, pages 12538–12559

gation tasks. Recently, zero-shot agents based on GPT (Zhou et al., 2023a; Long et al., 2023), have harnessed the power of GPT-4 (OpenAI, 2023) to make decisions in R2R dataset (Anderson et al., 2018). However, R2R dataset is based on a static, discrete, and unrealistic environment that lacks the complexity of real-world scenarios, including obstacle avoidance. These GPT-based methods may struggle when applied to real-world settings due to their limited consideration of environmental feedback. Additionally, other agents (Rajvanshi et al., 2023; Yu et al., 2023) only focus on object navigation but are unable to comprehend complex instructions.

To address the aforementioned challenges, we present CorNav, an autonomous agent with self-corrected planning for zero-shot vision-and-language navigation in continuous environment. CorNav possesses several key capabilities, including the ability to understand complex instructions, engage in self-corrected planning based on both environmental and historical feedback, and consult domain experts for crucial information. Here is a breakdown of CorNav's functionalities: 1) **Self-Corrected Planning:** During exploration, CorNav actively adapts its plan based on feedback. If the agent receives in-plan feedback, indicating that the environmental observation aligns with the plan, it adheres to the generated plan and proceeds with the next action. However, when faced with out-of-plan feedback, it modifies the plan accordingly. 2) **Domain Expert Consultation:** CorNav possesses complex reasoning and more accurate planning by seeking guidance from various domain experts. To manage computing resources and costs effectively, we have incorporated two key experts in addition to the vision perception expert: instruction parsing expert for understanding instructions and decision-making expert for evaluating and verifing the predicted actions. The distinctive features and differences between CorNav and existing VLN agents are illustrated in Figure 1. Through a series of extensive experiments across multiple tasks, our agent has demonstrated outstanding performance, underscoring the effectiveness of its self-corrected planning mechanism and its ability to communicate and collaborate with multiple domain experts.

We also develop a near-realistic simulator using Unreal Engine 5, which offers enhanced lighting and intricate details compared to previous simulators (Duan et al., 2022). Our simulator encompasses four scenes carefully modeled from real-world scenarios. Building upon this novel simulator, we have established a multi-task benchmark named NavBench for zero-shot vision-and-language navigation. Unlike traditional data collection methods, we have harnessed the capabilities of powerful GPT-4 to generate high-quality instructions for various tasks. NavBench has been designed to reflect realistic scenarios, covering four distinct tasks: 1) object navigation, namely, goal-conditioned navigation given a specific object category, which is a well-explored aspect of zero-shot navigation; 2) goal-conditioned navigation given a simple instruction, e.g., "I want to go upstairs. Please help me find the elevator"; 3) completing abstract instruction, e.g., "The floor is dirty and I want to sweep it", implying that the robot should locate a broom; 4) step-by-step instruction following, which simulates common realistic navigation scenarios where the agent must follow a series of step-by-step instructions. We have conducted an extensive study involving various large language models and open-vocabulary models, implementing 7 zero-shot baselines within the NavBench framework. Our experiments not only showcase the zero-shot capabilities of these foundational models but also highlight the challenging nature of NavBench as a benchmark.

In summary, our work presents three-fold contributions:

- **CorNav:** We introduce CorNav, a novel zero-shot VLN agent that stands out for its ability to adapt plans based on environmental feedback and its capacity to discuss with domain experts.
- **Realistic Simulator:** We have developed a realistic simulator using Unreal Engine 5, which provides a more immersive and challenging environment for our research.
- **NavBench:** We have established the NavBench benchmark, which leverages GPT-4 to generate and refine instructions in the dataset, eliminating the need for labor-intensive data collection.

2 Related Work

Vision-and-Language Navigation Language-guided visual navigation tasks have been a significant focus in recent research, and various models and tasks have contributed to advancing the field. The indoor navigation tasks such as R2R (Anderson et al., 2018) and RxR (Ku et al., 2020) pro-

https://www.unrealengine.com/

vide a foundation for language-guided navigation in simulated indoor environments. Many research efforts have been built upon these tasks, emphasizing cross-modal learning (Ma et al., 2019a), data augmentation (Fu et al., 2020; Tan et al., 2019; Liang et al., 2022b), waypoint tracking (Deng et al., 2020; Ma et al., 2019b), and pre-training using Transformer models to improve navigation performance (Hao et al., 2020; Hong et al., 2020; Liang et al., 2022a). In addition, there are other tasks including outdoor navigation task Touch-Down (Chen et al., 2019), dialogue-based navigation task CVDN (Thomason et al., 2020), and remote object-grounded navigation, such as REVERIE (Qi et al., 2020) and SOON (Zhu et al., 2021), introducing the challenge of associating time-correlated visual observations with decision-making instructions.

The performance of existing VLN methods often falls short when applied to the challenges of continuous 3D simulated environments, as exemplified by the more demanding task of VLN-CE (Krantz et al., 2020). Recent advancements in the field, driven by the availability of large-scale datasets and the development of continuous environment simulators like Habitat (Savva et al., 2019), GibsonEnv (Eftekhar et al., 2021), and AI2THOR (Kolve et al., 2017)), have enabled a new set of tasks and benchmarks, which include PointGoal navigation (Wijmans et al., 2019; Ye et al., 2021), ObjectGoal navigation (Chaplot et al., 2020a,b; Gervet et al., 2022; Ramakrishnan et al., 2022), and instructions following navigation (Krantz et al., 2021; Raychaudhuri et al., 2021; Hong et al., 2022; An et al., 2022). Wang et al. (Wang et al., 2022a) propose a large-scale indoor dataset designed for multimodal and multitask navigation in continuous and audiovisual complex environments.

Zero-shot Navigation The recent paradigm shifts in machine learning, driven by advancements in large-scale pre-training models, have indeed opened up exciting possibilities for zero-shot learning and have led to notable improvements in various downstream vision-language tasks, as demonstrated by Radford et al. (Radford et al., 2021). In zero-shot navigation, CoW (Gadre et al., 2023) leverages CLIP for localization and frontier-based exploration (FBE) for exploration strategies. Dorbala et al. (Dorbala et al., 2022, 2023) subsequently used a Costmap to handle obstacle avoidance. ESC (Zhou et al., 2023b) leverages a prompt-based language-image grounding model for open-world scene understanding and harnesses LLMs to acquire commonsense knowledge at object and room levels. Inspired by recent advancements and the progress in open vocabulary (Radford et al., 2021; Li et al., 2022; Kirillov et al., 2023; Liu et al., 2023; Kamath et al., 2021) pre-training models, our work aims to empower embodied robots for improved navigation to uncommon objects.

Large Language Model Large Language Models (LLMs) (Chowdhery et al., 2022; OpenAI, 2023; Touvron et al., 2023; Chiang et al., 2023; Geng et al., 2023; Taori et al., 2023; Du et al., 2022; Bai et al., 2022; Ouyang et al., 2022; Alayrac et al., 2022) have ushered in a significant transformation in the field of Artificial Intelligence (AI), particularly in language understanding, generation, and logical reasoning. These models have evolved over the years, with recent breakthroughs primarily attributed to factors such as larger model sizes, enhanced pre-training data, instruction-following Tuning, and reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022; OpenAI, 2023; Bai et al., 2022). Moreover, the development of hierarchical prompting systems for LLMs has gained prominence, aiming to enhance their logical reasoning abilities and response accuracy in specific domains (Wei et al., 2022; Wang et al., 2022b; Yao et al., 2022; Shinn et al., 2023; Yao et al., 2023).

3 CorNav

Our CorNav mainly comprises two pivotal components, i.e., multiple domain experts and plan refinement with environmental feedback. An overview of our architecture is shown in Figure 2. In this section, we will delve into the intricacies of these components, followed by a discussion of our navigation discussion mechanism and local policy.

3.1 Domain experts

Domain experts in our framework are powered by large models. In this section, we introduce three core experts: the instruction parsing expert, the vision perception expert, and the decision-making expert. For the purpose of cost-efficiency, we have implemented the instruction parsing expert and decision-making expert using the open-source Large Language Model (LLM) Vicuna v1.5 (Chiang et al., 2023). Additionally, we have conducted comparisons with GPT-4 Turbo on a subset of our experiments to further evaluate performance.

Instruction Parsing Expert. Our benchmark en-

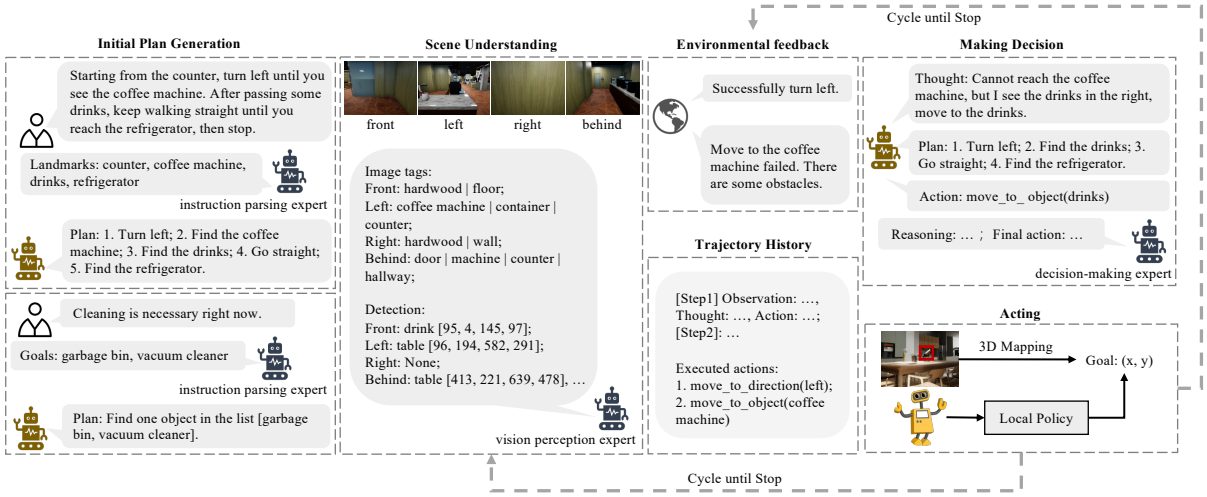

Figure 2: The overall architecture of our CorNav. After receiving the instruction, the instruction parsing expert extracts landmarks or figures out the needed objects. Then the agent generates the initial plan based on the instruction and information from the instruction parsing expert. The vision perception expert is driven by an image tagging model and an open-vocabulary grounding model, and performs scene understanding given four perspectives. Environmental feedback records both in-plan and out-of-plan feedback, while trajectory history maintains the reasoning process and executed actions. The decision-making expert assists the agent in deciding the final action. Finally, the local policy would plan a path for the robot.

compasses a diverse range of navigation tasks, spanning from simple to complex instructions. The instruction parsing expert excels at handling these instructions, particularly those of a more intricate nature. For complex instructions, the role of the expert is to extract crucial information and deduce the intended goal objects. In scenarios involving step-by-step instruction following, where the instruction may span several steps, the task of the expert is to extract relevant landmarks. This process is depicted in the top left of Figure 2. Utilizing the original instruction and the extracted landmarks, the planner agent generates an initial navigation plan, setting the stage for successful execution. In tasks such as completing abstract instructions, where specific object names may not be explicitly mentioned, the instruction parsing expert steps in to decipher the intent of the user. For instance, when presented with the instruction "I am thirsty", the expert deduces that the user likely requires water or a beverage and produces a list of potential options, such as "[water, drink]". This capability is showcased in the bottom left of Figure 2.

Vision Perception Expert. Vision perception is a fundamental module in VLN, tasked with providing comprehensive and accurate environmental information to aid navigation. To achieve this, we integrate two key components into this expert: an image tagging model and an open-vocabulary grounding model. For image tagging, we employ the robust and state-of-the-art model RAM++ (Huang et al., 2023). The vision perception expert processes input images captured from four different perspectives of the robot: front, left, right, and rear. RAM++ then predicts object tags for each of these perspectives. It is important to note that occasional noise in the predicted results can disrupt the planner's decision-making process. To mitigate the potential impact of noise in the predicted results, we introduce an open-vocabulary grounding model Grounding DINO (Liu et al., 2023) for object detection. This model prompts the planner to pay more attention on the objects that have been reliably detected. By combining image tagging and object detection capabilities, the expert enhances th overall environmental perception for the agent.

Decision-making Expert. The decision-making expert plays a pivotal role in overseeing and refining the predicted actions. The system employs four primary types of global actions, each serving a distinct purpose: move_to_object(), move_to_room(), move_to_direction(), and stop(). move_to_object(o) signifies the intention to approach and navigate near object o_i, while move_to_room(r) indicates the desire to enter and navigate into room r. If the agent faces uncertainty regarding object or room selection, then it would move in the most probable direction (i.e., front,

left, right, rear) considering the available environmental information. The primary responsibility of the expert is to review and validate the decisions generated by the planner. Vicuna v1.5 is not as intelligent as GPT-4, and might present a suboptimal decision. For example, the planner thinks that "Since the kettle is not detected in front of me, I will move to the left to check the shelf", and gives the action "move_to_direction(left)". Given the thought from the planner, the decision-making expert should recognize that the kettle may be on the shelf, and modify the action to a more suitable one "move_to_object(shelf)".

3.2 Environmental feedback

After generating the initial plan, the planner proceeds to predict the next global action based on this plan. However, real-world environments can introduce unexpected challenges, occasionally necessitating adjustments to the original plan to accommodate these changes.

In-Plan Feedback. In scenarios where the predicted action is move_to_object(o) or move_to_room(r), we leverage the detection model Grounding DINO to localize object o or room r. Then we can obtain the location of o or r in the simulator by transforming pixels in 2D images into 3D voxels, taking into account the agent's location, camera perspective, and depth information. To obtain more precise object locations, we employ SAM (Kirillov et al., 2023) for semantic segmentation. Then we can calculate the distance between the the agent's stopping location and the detected location of o or r. If this distance falls below a predefined threshold, we consider the action successful, indicating that the agent has effectively reached the vicinity of the object or room. Conversely, if the distance exceeds the threshold, the action is considered a failure. When the agent successfully moves near object o or room r, it signifies that the environment aligns with the anticipated plan. In such cases, the in-plan feedback is documented as "successfully move to o or r". For predicted actions of move_to_direction(d), the agent is expected to traverse a specific distance in the specified direction. Analogous to the previous cases, if the agent accomplishes this task without issues, the in-plan feedback is registered as "successfully turn to direction d". When the feedback indicates success, the planner retains the previous plan, as it accurately corresponds to the environment.

Out-of-Plan Feedback. As outlined earlier, our system provides a means to determine whether the agent successfully executes a planned action. When an action is executed unsuccessfully, the system registers out-of-plan feedback, indicating that obstacles or challenges were encountered during the execution. If the agent's attempt to move to object o or room r results in failure, it is indicated as "move to o or r failed". This feedback suggests that obstacles or impediments prevented the successful execution of the action, requiring further adaptation. Similarly, for actions involving move_to_direction(d), a failure is noted as "turn to direction d failed". Let $\{a_1, a_2, ..., a_t\}$ be the original plan. Upon receiving out-of-plan feedback at a particular step i, the planner takes corrective action. Specifically, the planner discards the subsequent actions in the original plan, resulting in the formulation of a new plan $\{a_1, a_2, ..., a_{i-1}, a'_i, ..., a'_t\}$ that considers the environmental feedback and observations. This revised plan is designed to guide the planner in making informed and adaptive decisions.

3.3 Navigation Discussion Mechanism

In this part, we delve into the comprehensive navigation discussion process, depicted in Figure 2. Initially, the planner calls upon the instruction parsing expert $\mathcal{I}$ to extract landmarks or infer goals from the instruction I. Subsequently, the planner generates an initial plan p based on the instruction and insights provided by the instruction parsing expert:

$$p = \mathcal{P}(I, \mathcal{I}(I)). \tag{1}$$

Simultaneously, the vision perception expert $\mathcal{V}$ processes observations $\mathcal{O}$ received from the environment. This expert summarizes the outcomes of image tagging and object detection. The trajectory history buffer $\mathcal{H}$ serves as a repository of historical information, encompassing observations, thoughts, and executed actions (a). At each time step t, if the planner receives out-of-plan feedback f, it triggers a series of actions to generate a new plan and decide as follows:

$$\begin{aligned} p' &= \mathcal{P}(\mathcal{H}, \mathcal{V}(\mathcal{O}), I, \mathcal{I}(I), p, \{a\}, f), \\ T_{t+1}, a_{t+1} &= \mathcal{P}(\mathcal{H}, \mathcal{V}(\mathcal{O}), I, \mathcal{I}(I), p', \{a\}), \end{aligned} \tag{2}$$

where T_{t+1} indicates thought for taking action a_{t+1}. In the absence of out-of-plan feedback, the planner relies on the original plan p for decision-making. Subsequently, the planner engages the decision-

making expert $\mathcal{D}$ to arrive at a final decision:

$$T'_{t+1}, a'_{t+1} = \mathcal{D}(T_{t+1}, a_{t+1}). \quad (3)$$

3.4 Local Policy

Once the agent has ascertained the goal location, as outlined in Section 3.2, the local policy engages the Fast Marching Method (Sethian, 1996) to formulate a path from the current location to the designated goal. This path planning process ensures efficient and effective navigation to reach the specified objective.

4 NavBench

Our simulator is featured for indoor navigation. It stands out by seamlessly integrating a state-of-the-art real-time physics engine, which significantly elevates the quality of visual rendering. Notably, our simulator boasts dynamic global illumination and diffuse global illumination, allowing for more highly detailed geometry rendering than ever before. Table 1 offers a comprehensive comparison between our simulator and existing counterparts, highlighting the distinctive features and capabilities that set ours apart. We conduct a visual quality assessment involving 60 human participants on previous and our proposed simulators in Table 1.

Our benchmark, NavBench, is purposefully designed to address the challenges of zero-shot multi-task vision-and-language navigation. It encompasses a wide range of tasks, including navigation to specific goal objects, abstract objects, and specific locations, all guided by natural language instructions. Unlike previous benchmarks relying on manual data labeling, we collect data using large language models. Table 2 succinctly outlines the key distinctions between NavBench and previous benchmarks.

4.1 Task Definition

To construct our benchmark, we have categorized it into four distinct tasks, including object navigation given a category, goal-conditioned navigation given a simple instruction, completing abstract instruction, and step-by-step instruction following.

Object navigation given a category (ObjectNav). This task revolves around the navigation to specific objects based on their predefined categories within our simulated environments.

Goal-conditioned navigation given a simple instruction (Simple). In this task, the language instructions provided to the agent contain references to object categories or closely related terms. For example, instructions generated by GPT-4 may resemble, "Proceed toward the nearest mug that is detectable", where the term "mug" is included in the instruction.

Completing abstract instruction (Abstract). Here, the instructions issued to the agent are intentionally abstract and do not explicitly mention object names. Instead, the agent must infer the intended goal based on the user's abstract intent. For instance, when presented with the instruction "I am thirsty", the agent should deduce that the user requires water or a beverage and output a list such as "[water, drink]".

Step-by-step instruction following (Step-by-step). In this task, the agent is required to follow detailed step-by-step instructions provided in the language. This mirrors real-world navigation scenarios where complex instructions guide the agent's actions.

It is noteworthy that for the last three tasks, we employ the cutting-edge GPT-4 (OpenAI, 2023) to generate the dataset. GPT-4's remarkable language generation capabilities are instrumental in crafting realistic and diverse instructions for these tasks, enabling a comprehensive evaluation of the agent's performance.

4.2 Dataset Statistics

Our benchmark encompasses four distinct scenes, including a restaurant, cafe, nursing room, and home settings. In total, our dataset comprises a substantial corpus of 1,615 instructions. Specifically, the distribution of instructions across tasks is as follows: 81 instructions for ObjectNav, 494 instructions for the simple tasks, 278 for the abstract task, and 762 for the step-by-step task. More details are shown in Appendix.

5 Experiment

5.1 Experimental Setup

Navigation metrics. We use standard navigation metrics to measure performance: Success Rate (SR), the fraction of episodes where the agent successfully reaches within 1.5m of the target object or location; Success Rate weighted by inverse path Length (SPL), success weighted by the oracle shortest path length and normalized by the actual path length (Batra et al., 2020); and Distance to Success (DTS), the distance of the agent from the success

Table 1: Comparison with embodied AI simulators. Physics simulation: basic physics features (B) and advanced physics features (A). Model library support: built-in (L) and user-extensible (E). Action interactivity: navigation (N), object manipulation (M), and human-computer interaction using virtual reality (VR) devices (H). Pedestrian: adding walking and stationary pedestrians. Visual quality: 5 indicates most realistic, while 1 represents least.

Simulator	Simulation Engine	Physics	Models	Action	Pedestrian	Object number	Object category	Visual quality
DeepMind Lab (Beattie et al., 2016)	Quake II Arena Engine	-	-	N	✗	-	-	1.6
CHALET (Yan et al., 2018)	Unity 3D Engine	B	-	N, M	✗	1740	150	2.7
VirtualHome (Puig et al., 2018)	Unity 3D Engine	-	-	N, M, H	✗	2142	308	2.5
VRKitchen (Gao et al., 2019)]	Unreal Engine 4	B	-	N, M	✗	880	-	2.4
Habitat-Sim (Savva et al., 2019)	Bullet	B	-	N	✗	92	-	2.2
AI2-THOR (Kolve et al., 2017)	Unity 3D Engine	B	L	N, M	✗	609	-	3.2
iGibson (Xia et al., 2020)	PyBullet	B	L	N, M	✗	570	-	2.8
SAPIEN (Xiang et al., 2020)	PhysX Physical engine and ROS	B	L	N, M	✗	2346	46	1.8
ThreeDWorld (Gan et al., 2020)	Unity 3D Engine	B, A	L, E	N, M, H	✗	2500	200	3.2
BEHAVIOR-1K (Li et al., 2023)	Nvidia's Omniverse	B, A	L, E	N, M	✗	5215	1265	3.5
NavBench (Ours)	Unreal Engine 5	B, A	L, E	N, M, H	✓	4758	2165	4.3

Table 2: Comparison with existing vision-and-language benchmarks.

Benchmark	Simulator	Continuous	Tasks	Instruction Type
R2R (Anderson et al., 2018)	Matterport3D (Anderson et al., 2018)	✗	1	Route-oriented
RoomNav (Wu et al., 2018)	House3D (Wu et al., 2018)	✓	1	Goal-oriented
LANI (Misra et al., 2018)	AI2-THOR (Kolve et al., 2017)	✓	1	Goal-oriented
3D Doom (Chaplot et al., 2018)	VizDoom (Kempka et al., 2016)	✓	1	Goal-oriented
VNLA (Nguyen et al., 2019)	Matterport3D	✗	1	Oracle guidance
HANNA (Nguyen and Daumé III, 2019)	Matterport3D	✗	1	Oracle guidance
R4R (Jain et al., 2019)	Matterport3D	✗	1	Route-oriented
CVDN (Thomason et al., 2020)	Matterport3D	✗	1	Dialogue
R6R, R8R (Zhu et al., 2020)	Matterport3D	✗	1	Route-oriented
RxR (Ku et al., 2020)	Matterport3D	✗	1	Route-oriented
VLNCE (Krantz et al., 2020)	Habitat (Savva et al., 2019)	✓	1	Route-oriented
REVERIE (Qi et al., 2020)	Matterport3D	✗	1	Goal-oriented
SOON (Zhu et al., 2021)	Matterport3D	✗	1	Goal-oriented
BnB(Guhur et al., 2021)	-	✗	1	Route-oriented
ROBUSTNAV (Chattopadhyay et al., 2021)	ROBOTHOR (Deitke et al., 2020)	✓	2	Goal-oriented
PASTURE (Gadre et al., 2023)	ROBOTHOR	✓	3	Goal-oriented
NavBench (Ours)	Ours	✓	4	Goal-oriented, Route-oriented

threshold boundary when the episode ends (Chaplot et al., 2020b).

Embodiment. We define four actions: *Move Forward*, *Turn Left*, *Turn Right*, and *Stop*. The *Move Forward* action advances the agent by 20cm, while *Turn Left* and *Turn Right* actions turn 15°horizontally.

5.2 Comparison of Zero-Shot Methods

We have implemented a total of seven baseline models, leveraging three distinct methods for zero-shot object navigation in a continuous environment, as detailed in Table 3. Recognizing that these methods are primarily designed for object navigation and may struggle with longer instructions, we employ LLMs to parse instructions, which is the same as our instruction parsing expert. The results of these baselines are summarized in Table 3. Notably, the models incorporating GLIP or Grounding DINO tend to outperform the CoW baseline. Interestingly, the co-occurrence knowledge from LLMs in ESC appears to have a lesser impact on the results. It is worth noting that selecting boundaries based on common sense may not be ideal in our specific scenarios.

5.3 Compare CorNav with Previous Methods

In our evaluation, we compare CorNav with previous methods, and the results are summarized in Table 3. Notably, our method outperforms all baselines across all four tasks, achieving an average Success Rate (SR) of 28.1%. This represents a significant improvement, with a 7.6% increase compared to the best-performing baseline. Particularly noteworthy is CorNav's remarkable performance in the step-by-step task, where it achieves an 8.6% increase in SR. This outcome underscores the effectiveness of our approach, which incorporates environmental feedback and leverages trajectory history to enhance navigation capabilities.

Table 3: Multi-task navigation results on NavBench.

		ObjectNav		Simple		Abstract		Step-by-step		Avg.
Model	Detector	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR
CoW (Gadre et al., 2023)	CLIP-Grad.	12.2	10.2	15.0	11.5	28.4	25.8	18.5	6.7	18.5
FBE (Yamauchi, 1997)	GLIP (Li et al., 2022)	18.3	14.3	17.4	13.7	28.1	24.0	17.7	13.8	20.4
FBE (Yamauchi, 1997)	Grounding DINO (Liu et al., 2023)	17.1	11.4	15.6	10.8	27.1	21.2	19.5	12.6	19.8
ESC (Vicuna v1.5-13B) (Zhou et al., 2023b)	GLIP	15.8	12.9	14.8	11.6	25.6	22.5	17.7	13.5	18.5
ESC (Vicuna v1.5-13B) (Zhou et al., 2023b)	Grounding DINO	18.3	11.3	15.2	9.2	25.9	20.1	20.8	11.2	20.1
ESC (GPT-4) (Zhou et al., 2023b)	GLIP	15.9	13.3	16.8	13.2	26.0	23.6	20.5	16.2	19.8
ESC (GPT-4) (Zhou et al., 2023b)	Grounding DINO	18.3	11.7	15.6	10.0	30.2	22.9	18.0	9.9	20.5
CorNav (Vicuna v1.5-13B)	Grounding DINO	**23.4**	**16.1**	**23.7**	**19.8**	**36.0**	**29.0**	**29.4**	**23.1**	**28.1**

5.4 Ablation Study

The effect of environmental feedback. To assess the significance of environmental feedback, we conducted an ablation study, the results of which are presented in Table 4. Notably, the inclusion of the plan refiner with environmental feedback (row 2) yields a remarkable improvement over the baseline (row 1). This highlights the crucial role that environmental feedback plays in enhancing the realism of navigation.

The effect of trajectory history. Examining the impact of trajectory history, as demonstrated in Table 4 reveals that planning with trajectory history (row 3) further improves the results compared to the plan refiner alone (row 2). This observation aligns with the inherent logic of navigation, where past actions and experiences inform future decisions.

The effect of multiple experts. Our study also delves into the effects of consulting with domain experts, specifically, the decision-making expert and the instruction parsing expert. As depicted in Table 4, involving the decision-making expert contributes to improved navigation outcomes, suggesting instances where the agent's decisions might have been sub-optimal. Further insights emerge from Table 5, where the instruction parsing expert exhibits significant enhancements in the abstract task (SR +8.3%). Parsing instructions becomes particularly important in scenarios where object names are absent, emphasizing its importance.

The effect of environmental description. The vision perception expert incorporates both an image tagging model and an object detection model. An ablation study, detailed in Table 6, reveals that while utilizing either image tags or detection results alone yields similar performance, combining both aspects results in significantly improved performance.

Compare between Vicuna v1.5 and GPT-4. We conducted a comparative analysis between Vicuna v1.5-13B and GPT-4 Turbo, focusing on a subset of our dataset. For this subset, we randomly selected three instructions from each scene for each task, resulting in a total of 48 instructions. The results are presented in Table 7. Remarkably, GPT-4 Turbo exhibited a substantial improvement in performance (+6.3% on SR) compared to Vicuna v1.5. This suggests that GPT-4 Turbo operates as a more intelligent agent.

Table 4: Ablation study about different components of CorNav on ObjectNav.

Method	SR	SPL	DTS (m)
Baseline	18.5	13.8	7.85
+ Environmental Feedback	21.0	**16.1**	7.74
+ Trajectory History	22.2	16.0	7.80
+ Decision-making Expert	**23.4**	**16.1**	**7.52**

Table 5: Ablation study about instruction parsing expert on complex tasks.

	Abstract		Step-by-step	
Method	SR	SPL	SR	SPL
w/o instruction parsing expert	27.7	21.7	28.1	21.7
CorNav (Vicuna v1.5-13B)	**36.0**	**29.0**	**29.4**	**23.1**

6 Conclusion

In this paper, we introduce CorNav, an innovative autonomous agent designed for zero-shot VLN. CorNav excels in leveraging environmental feedback to refine its plans in realistic scenarios, ensuring adaptability to dynamic surroundings. It also incorporates multiple domain experts for instruction parsing, scene comprehension, and action refinement. Our experimental results demonstrate CorNav's significant performance advantages over baseline methods across various navigation tasks. Furthermore, we contribute to the field by developing a more realistic simulator powered by Unreal Engine 5. To evaluate our agent's capabilities,

Table 6: Comparison of results from different environmental description on ObjectNav.

Description	SR	SPL	DTS (m)
Image Tags	21.0	15.2	7.95
Detection	21.0	14.6	8.29
Image Tags + Detection	**23.4**	**16.1**	**7.52**

Table 7: Comparison between Vicuna v1.5-13B and GPT-4 Turbo on a subset containing four tasks.

Method	SR	SPL	DTS (m)
CorNav (Vicuna v1.5-13B)	20.8	13.0	7.01
CorNav (GPT-4 Turbo)	**27.1**	**18.5**	**5.87**

we create NavBench, a comprehensive multi-task benchmark for open-set zero-shot VLN. Leveraging the powerful GPT-4, we generate and self-refine a range of free-form instructions for different tasks within NavBench, including goal-conditioned navigation, abstract object retrieval, and step-by-step instruction following. Our benchmark offers a challenging platform for assessing navigation methods.

Limitations While CorNav has demonstrated remarkable performance across tasks, it relies on the outcomes of the image tagging and object detection models. These models may introduce noise or miss certain objects in the environment. Future research could explore fine-tuning existing vision-language models specifically for navigation, potentially yielding even better results.

Acknowledgements

This work was supported in part by the National Science and Technology Major Project under Grant No. 2020AAA0109704, Guangdong Outstanding Youth Fund (Grant No. 2021B1515020061), Mobility Grant Award under Grant No. M-0461, Shenzhen Science and Technology Program (Grant No. GJHZ20220913142600001), Nansha Key RD Program under Grant No.2022ZD014, CAAI-Huawei MindSpore Open Fund. We thank MindSpore for the partial support of this work, which is a new deep learning computing framework.

References

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. 2022. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736.

Dong An, Yuankai Qi, Yangguang Li, Yan Huang, Liang Wang, Tieniu Tan, and Jing Shao. 2022. Bevbert: Topo-metric map pre-training for language-guided navigation. *arXiv preprint arXiv:2212.04385*.

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

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. 2022. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*.

Dhruv Batra, Aaron Gokaslan, Aniruddha Kembhavi, Oleksandr Maksymets, Roozbeh Mottaghi, Manolis Savva, Alexander Toshev, and Erik Wijmans. 2020. Objectnav revisited: On evaluation of embodied agents navigating to objects. *arXiv preprint arXiv:2006.13171*.

Charles Beattie, Joel Z Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, et al. 2016. Deepmind lab. *arXiv preprint arXiv:1612.03801*.

Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. 2020a. Learning to explore using active neural slam. *arXiv preprint arXiv:2004.05155*.

Devendra Singh Chaplot, Dhiraj Prakashchand Gandhi, Abhinav Gupta, and Russ R Salakhutdinov. 2020b. Object goal navigation using goal-oriented semantic exploration. *Advances in Neural Information Processing Systems*, 33:4247–4258.

Devendra Singh Chaplot, Kanthashree Mysore Sathyendra, Rama Kumar Pasumarthi, Dheeraj Rajagopal, and Ruslan Salakhutdinov. 2018. Gated-attention architectures for task-oriented language grounding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Prithvijit Chattopadhyay, Judy Hoffman, Roozbeh Mottaghi, and Aniruddha Kembhavi. 2021. Robustnav: Towards benchmarking robustness in embodied navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15691–15700.

Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. 2019. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *Proceedings of the*

https://www.mindspore.cn/

IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 12538–12547.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2022. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*.

Matt Deitke, Winson Han, Alvaro Herrasti, Aniruddha Kembhavi, Eric Kolve, Roozbeh Mottaghi, Jordi Salvador, Dustin Schwenk, Eli VanderBilt, Matthew Wallingford, et al. 2020. Robothor: An open simulation-to-real embodied ai platform. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3164–3174.

Zhiwei Deng, Karthik Narasimhan, and Olga Russakovsky. 2020. Evolving graphical planner: Contextual global planning for vision-and-language navigation. *Advances in Neural Information Processing Systems*, 33:20660–20672.

Vishnu Sashank Dorbala, James F Mullen Jr, and Dinesh Manocha. 2023. Can an embodied agent find your" cat-shaped mug"? llm-based zero-shot object navigation. *arXiv preprint arXiv:2303.03480*.

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

Zhengxiao Du, Yujie Qian, Xiao Liu, Ming Ding, Jiezhong Qiu, Zhilin Yang, and Jie Tang. 2022. GLM: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 320–335.

Jiafei Duan, Samson Yu, Hui Li Tan, Hongyuan Zhu, and Cheston Tan. 2022. A survey of embodied ai: From simulators to research tasks. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 6(2):230–244.

Ainaz Eftekhar, Alexander Sax, Jitendra Malik, and Amir Zamir. 2021. Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10786–10796.

Tsu-Jui Fu, Xin Eric Wang, Matthew F Peterson, Scott T Grafton, Miguel P Eckstein, and William Yang Wang. 2020. Counterfactual vision-and-language navigation via adversarial path sampler. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*, pages 71–86. Springer.

Samir Yitzhak Gadre, Mitchell Wortsman, Gabriel Ilharco, Ludwig Schmidt, and Shuran Song. 2023. 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.

Chuang Gan, Jeremy Schwartz, Seth Alter, Damian Mrowca, Martin Schrimpf, James Traer, Julian De Freitas, Jonas Kubilius, Abhishek Bhandwaldar, Nick Haber, et al. 2020. Threedworld: A platform for interactive multi-modal physical simulation. *arXiv preprint arXiv:2007.04954*.

Xiaofeng Gao, Ran Gong, Tianmin Shu, Xu Xie, Shu Wang, and Song-Chun Zhu. 2019. Vrkitchen: an interactive 3d virtual environment for task-oriented learning. *arXiv preprint arXiv:1903.05757*.

Xinyang Geng, Arnav Gudibande, Hao Liu, Eric Wallace, Pieter Abbeel, Sergey Levine, and Dawn Song. 2023. Koala: A dialogue model for academic research. Blog post.

Theophile Gervet, Soumith Chintala, Dhruv Batra, Jitendra Malik, and Devendra Singh Chaplot. 2022. Navigating to objects in the real world. *arXiv preprint arXiv:2212.00922*.

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

Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. 2020. Towards learning a generic agent for vision-and-language navigation via pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13137–13146.

Yicong Hong, Cristian Rodriguez, Yuankai Qi, Qi Wu, and Stephen Gould. 2020. Language and visual entity relationship graph for agent navigation. *Advances in Neural Information Processing Systems*, 33:7685–7696.

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

Xinyu Huang, Yi-Jie Huang, Youcai Zhang, Weiwei Tian, Rui Feng, Yuejie Zhang, Yanchun Xie, Yaqian Li, and Lei Zhang. 2023. Inject semantic concepts into image tagging for open-set recognition. *arXiv preprint arXiv:2310.15200*.

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

Aishwarya Kamath, Mannat Singh, Yann LeCun, Gabriel Synnaeve, Ishan Misra, and Nicolas Carion. 2021. Mdetr-modulated detection for end-to-end multi-modal understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1780–1790.

Michał Kempka, Marek Wydmuch, Grzegorz Runc, Jakub Toczek, and Wojciech Jaśkowski. 2016. Vizdoom: A doom-based ai research platform for visual reinforcement learning. In *2016 IEEE conference on computational intelligence and games (CIG)*, pages 1–8.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. 2023. Segment anything. *arXiv preprint arXiv:2304.02643*.

Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, et al. 2017. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*.

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

Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. 2020. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *European Conference on Computer Vision*, pages 104–120.

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

Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Michael Lingelbach, Jiankai Sun, et al. 2023. Behavior-1k: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation. In *Conference on Robot Learning*, pages 80–93. PMLR.

Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. 2022. Grounded language-image pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10965–10975.

Xiwen Liang, Fengda Zhu, Lingling Li, Hang Xu, and Xiaodan Liang. 2022a. Visual-language navigation pretraining via prompt-based environmental self-exploration. *arXiv preprint arXiv:2203.04006*.

Xiwen Liang, Fengda Zhu, Yi Zhu, Bingqian Lin, Bing Wang, and Xiaodan Liang. 2022b. Contrastive instruction-trajectory learning for vision-language navigation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 1592–1600.

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

Yuxing Long, Xiaoqi Li, Wenzhe Cai, and Hao Dong. 2023. Discuss before moving: Visual language navigation via multi-expert discussions. *arXiv preprint arXiv:2309.11382*.

Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. 2019a. Self-monitoring navigation agent via auxiliary progress estimation. *arXiv preprint arXiv:1901.03035*.

Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, and Zsolt Kira. 2019b. The regretful agent: Heuristic-aided navigation through progress estimation. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 6732–6740.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. Self-refine: Iterative refinement with self-feedback. *arXiv preprint arXiv:2303.17651*.

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

Khanh Nguyen and Hal Daumé III. 2019. Help, anna! visual navigation with natural multimodal assistance via retrospective curiosity-encouraging imitation learning. *arXiv preprint arXiv:1909.01871*.

Khanh Nguyen, Debadeepta Dey, Chris Brockett, and Bill Dolan. 2019. Vision-based navigation with language-based assistance via imitation learning with indirect intervention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12527–12537.

OpenAI. 2023. Gpt-4 technical report. *ArXiv*, abs/2303.08774.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al.

2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Debjit Paul, Mete Ismayilzada, Maxime Peyrard, Beatriz Borges, Antoine Bosselut, Robert West, and Boi Faltings. 2023. Refiner: Reasoning feedback on intermediate representations. *arXiv preprint arXiv:2304.01904*.

Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. 2018. Virtualhome: Simulating household activities via programs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8494–8502.

Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. 2020. 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.

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

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

Santhosh Kumar Ramakrishnan, Devendra Singh Chaplot, Ziad Al-Halah, Jitendra Malik, and Kristen Grauman. 2022. Poni: Potential functions for objectgoal navigation with interaction-free learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18890–18900.

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

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

James A Sethian. 1996. A fast marching level set method for monotonically advancing fronts. *proceedings of the National Academy of Sciences*, 93(4):1591–1595.

Noah Shinn, Beck Labash, and Ashwin Gopinath. 2023. Reflexion: an autonomous agent with dynamic memory and self-reflection. *arXiv preprint arXiv:2303.11366*.

Theodore Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas L Griffiths. 2023. Cognitive architectures for language agents. *arXiv preprint arXiv:2309.02427*.

Haotian Sun, Yuchen Zhuang, Lingkai Kong, Bo Dai, and Chao Zhang. 2023. Adaplanner: Adaptive planning from feedback with language models. *arXiv preprint arXiv:2305.16653*.

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

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. 2020. Vision-and-dialog navigation. In *Conference on Robot Learning*, pages 394–406. PMLR.

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

Hanqing Wang, Wei Liang, Luc V Gool, and Wenguan Wang. 2022a. Towards versatile embodied navigation. *Advances in Neural Information Processing Systems*, 35:36858–36874.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. 2023. A survey on large language model based autonomous agents. *arXiv preprint arXiv:2308.11432*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022b. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

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

Yi Wu, Yuxin Wu, Georgia Gkioxari, and Yuandong Tian. 2018. Building generalizable agents with a realistic and rich 3d environment. *arXiv preprint arXiv:1801.02209*.

Fei Xia, William B Shen, Chengshu Li, Priya Kasimbeg, Micael Edmond Tchapmi, Alexander Toshev, Roberto Martín-Martín, and Silvio Savarese. 2020. Interactive gibson benchmark: A benchmark for interactive navigation in cluttered environments. *IEEE Robotics and Automation Letters*, 5(2):713–720.

Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. 2020. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11097–11107.

Brian Yamauchi. 1997. A frontier-based approach for autonomous exploration. In *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97.'Towards New Computational Principles for Robotics and Automation'*, pages 146–151. IEEE.

Claudia Yan, Dipendra Misra, Andrew Bennnett, Aaron Walsman, Yonatan Bisk, and Yoav Artzi. 2018. Chalet: Cornell house agent learning environment. *arXiv preprint arXiv:1801.07357*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Joel Ye, Dhruv Batra, Erik Wijmans, and Abhishek Das. 2021. Auxiliary tasks speed up learning point goal navigation. In *Conference on Robot Learning*, pages 498–516.

Bangguo Yu, Hamidreza Kasaei, and Ming Cao. 2023. Co-navgpt: Multi-robot cooperative visual semantic navigation using large language models. *arXiv preprint arXiv:2310.07937*.

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

Kaiwen Zhou, Kaizhi Zheng, Connor Pryor, Yilin Shen, Hongxia Jin, Lise Getoor, and Xin Eric Wang. 2023b. Esc: Exploration with soft commonsense constraints for zero-shot object navigation. *arXiv preprint arXiv:2301.13166*.

Fengda Zhu, Xiwen Liang, Yi Zhu, Qizhi Yu, Xiaojun Chang, and Xiaodan Liang. 2021. 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.

Wang Zhu, Hexiang Hu, Jiacheng Chen, Zhiwei Deng, Vihan Jain, Eugene Ie, and Fei Sha. 2020. Babywalk: Going farther in vision-and-language navigation by taking baby steps. *arXiv preprint arXiv:2005.04625*.

A More Method Details

In this section, we elaborate on the functionalities of the acting module, as depicted in Figure 3. In scenarios where the predicted action is move_to_object(o) or move_to_room(r), our approach incorporates the open-vocabulary model such as Grounding DINO for localizing the specified object o or room r. Subsequently, the localization of o or r in the simulation environment is achieved by converting pixel representations in 2D image into 3D voxel. This conversion process takes into account the agent's location, the camera's perspective, and depth information. In order to enhance the precision of object localization, we engage the SAM (Kirillov et al., 2023) for semantic segmentation in semantic localization. Following the acquisition of both the obstacle map and the designated goal location, the agent navigates towards the goal utilizing the fast marching technique as in Section 3.4.

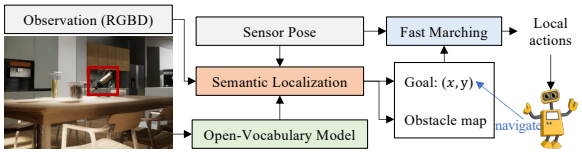

Figure 3: The illustration of the acting module in CorNav.

B Simulator

Scenes Our simulator provides four scenes, i.e., cafe, restaurant, nursing room, and home, which have a relatively large demand for robots. The overview of scenes is shown in Figure 4. These scenes are rendered based on real-world scenarios and contain more realistic details. The simulator is flexible and users can change the lighting and place more additional objects in the scene.

Agents Our simulator supports multiple agents with different practical uses. For example, the humanoid robot can navigate like a human and perform more actions such as turning the head or nodding to have a wider view, while the sweeping robot

Figure 4: Our simulator includes scenes of different difficulty, i.e., restaurant, cafe, nursing room, and home.

aims at cleaning the floor. We list all supported agents in Figure 5. Different agents have different movable joints. We mainly use the humanoid agent in our experiments.

Actions Our simulator supports continuous move or teleport actions. Users can define discrete actions such as rotating right by 30°. The humanoid agent has 21 movable joints that can make all human movements, including rotation of the head, neck, and waist.

Objects Our simulator was built with 2,165 categories in total. We choose 129 categories among them for interaction. Except for common objects in indoor environments, our simulator also includes some uncommon objects and more fine-grained categories, such as "soft drink" and "juice". Figure 6 shows some samples of objects in our simulator. Users can generate more objects in the scene by Python API.

B.1 Image Modalities

Different image modalities from different cameras are shown in Figure 7. There are three image modalities in the scene, including RGB, depth, and semantic segmentation. Some humanoid agents have three cameras on the head, chest, and waist, respectively.

B.2 3D Model

Our simulator is built based on real scenarios at a 1:1 ratio. We use 689 object models, classified into 534 categories, to build the cafe scene. For building the restaurant scene, we employ 2,023 object models classified into 782 categories. For the nursing house scene, we adopt 1,518 object models, which can be classified into 849 categories. For the home scene, we utilize 528 models.

Since our ultimate goal is to develop intelligent robots that can perform multiple tasks such as navigation and grasping in the future, we select some of these categories to build the benchmark. We have chosen 65 categories in the four scenes to evaluate navigation approaches. Among these scenes, the cafe contains 17 categories, the restaurant includes 14 categories, the nursing room has 32 categories, and the home scene contains 18 categories.

We also provide Python API for users to generate more objects in the scene. For example, put more stuff on the table or on the floor. In this way, they can change the layout of objects in the scene by themselves. There are a total of 129 categories for interaction.

B.3 Pedestrian

Our simulator supports walking and stationary pedestrians and provides Python API for users to generate and control them, aligning more closely with real-world navigation scenarios. This feature is not included in previous benchmarks and significant for developing navigation methods in complex scenarios. We present some examples in Figure 8.

C Dataset Details

Figure 9 (a) displays the length distribution of the collected instructions for three tasks. It shows that most instructions have 8 $\sim$ 12 words in the simple task, while most instructions have 15 $\sim$ 18 words in the reasoning task. For step-by-step instruction following, most instructions have 50 $\sim$ 70 words. We also compute the number of mentioned objects in the step-by-step instruction following and its distribution is presented in Figure 9 (b). It shows that 32% instructions mention 2 objects, 14% instruc-

Figure 5: Supported agents in our simulator. We include agents in a variety of application scenarios, such as humanoid agents, sweeping agents, and delivery agents.

Figure 6: Examples of objects in our simulator.

tions mention 9 objects, and around 20% instructions mention 13 $\sim$ 15 objects.

Figure 10 (a) presents the relative amount of words used in instructions in the form of the word cloud. It shows that GPT-4 prefers to generate 'find', 'nearby', and 'toward' for navigation. Most instructions involve 'something'. For step-by-step instruction following, most instructions involve 'door' and 'chair'.

D Prompt Details

D.1 Prompts for Data Collection

We have to generate instructions for three tasks: goal-conditioned navigation given a simple instruction, completing abstract instruction, and step-by-step instruction following. We detail prompts for different tasks during data collection in the following.

D.1.1 Goal-conditioned navigation given a simple instruction

> As the navigator of a walking robot, your task is to provide clear instructions for it to find specific objects.
> When I tell you the name of the item that the robot needs to go to, please output a command to the robot.
> For example:
> when I say "Bread", you should say "Get close to a loaf of bread", "Find a loaf of bread", "Go to a loaf of bread";
> when I say "Teacup", you should say "Please go to find a teacup", "Get to a teacup", "Robot, please find a teacup";
> when I say "Drinking Machine", you should say "Seek out a drinking machine in your vicinity", "Get near to a drinking machine nearby", "Navigate towards a drinking machine";
> when I say "Glass", you should say "Robot, help me to find a glass", "Make your way to the glass I referred to", "Navigate to a glass

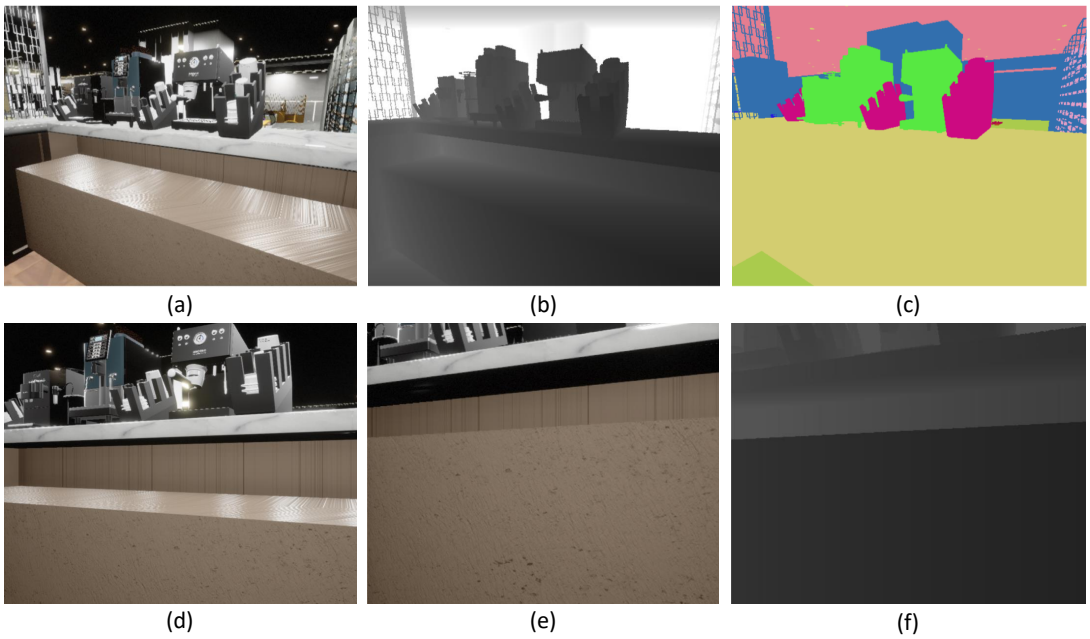

Figure 7: Examples of image modalities in our simulator: (a) RGB image from head camera; (b) depth image from head camera; (c) semantic segmentation mask from head camera; (d) RGB image from chest camera; (e) RGB image from waist camera; (f) depth image from waist camera.

Figure 8: Examples of stationary and walking pedestrians in our simulator.

nearby";
Please adhere to the following guidelines:
1. Avoid using phrases such as "for example", "I can say", "or", and "and";
2. Only provide clear instructions, and additional inquiries are not allowed;
3. Do not refer to any specific location, such as a kitchen or drawing room;
4. Only mention the provided items and avoid mentioning any other objects;
5. Avoid explicitly stating where the robot should retrieve an object from;
6. Use simple language when providing instructions.

Remember, when giving instructions, make sure they are clear enough for the robot to understand which object it needs to go to. The instructions should only require the robot to find and walk toward that specific object. Avoid mentioning any additional requirements beyond walking.
So, give the robot 10 instructions about {Object}:

D.1.2 Completing abstract instruction

As the navigator of a walking robot, your task is to provide clear instructions for it to find specific objects.

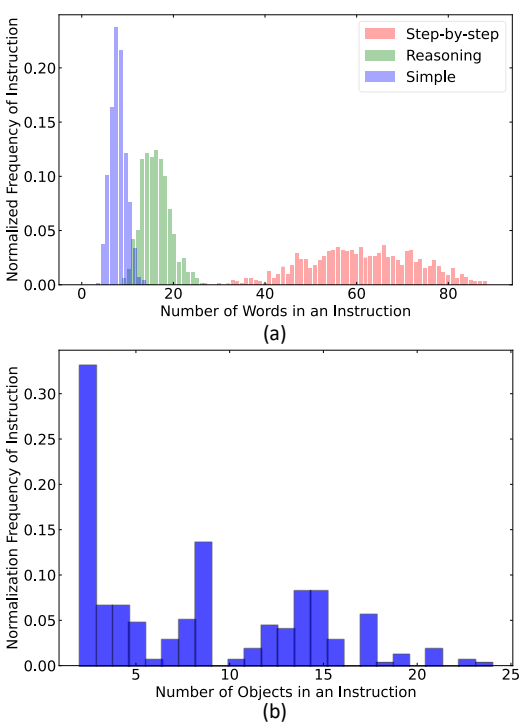

Figure 9: The distribution of the number of words (a) and objects in Step-by-step (b) in each instruction.

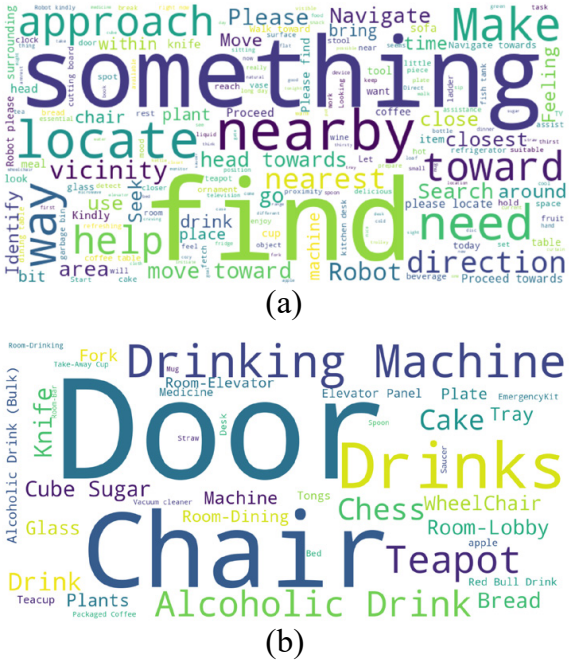

Figure 10: Word cloud of all instructions (a) and target objects in Step-by-step (b) in our dataset. The bigger the font, the more percentage it occupies.

When I tell you the name of the item that the robot needs to go to, please output a command to the robot.
For example:
when I say "bread", you should say: "I'm hungry and want something to eat.", "I just got off work and haven't eaten yet. Find me something to eat".
Please adhere to the following guidelines:
1. Avoid using phrases such as "for example", "I can say", "or", and "and";
2. Only provide instructions; additional inquiries are not allowed;
3. Do not refer to any specific location, such as a kitchen or drawing room;
4. Only mention the provided items and avoid mentioning any other objects.
5. Avoid explicitly stating where the robot should retrieve an object from.
6. Use less direct expressions
Remember, when giving instructions, make sure they never mention the name of the given object. The instructions should only require the robot to find and walk toward that specific object. Avoid mentioning any additional requirements beyond walking.

And NEVER DIRECTLY reference to the name I give.
For example:
I say "Banana", You can say: "Feeling a bit low on energy. Could you fetch something to pick me up?", "Find me something I can eat on the go, without needing a plate or utensils", "Just got back from my run. I want to use a quick natural energy booster", "Find me a snack that's a good source of natural sugars";
I say "Blood pressure monitor", You can say: "I've been feeling a bit light-headed lately. I need to check something important", "I think it's time for a quick health check, but nothing too invasive", "I've been feeling a bit odd lately, and want to use something to check my wellness".
So, give the robot 1 instruction about {Object}:

D.1.3 Step-by-step instruction following

Imagine that you are someone who is good at expressing needs. Now, you need to command a robot to follow a certain path and reach a specific object. Remember the fol-

lowing rules:
1. Use everyday expressions as much as possible;
2. Keep your words as simple as possible, don't get too complicated, and focus on the key objects;
3. Do not use expressions like "position 0" to distinguish;
4. Only provide clear instructions; additional inquiries are not allowed;
5. Only mention the items that have been provided and avoid mentioning any other objects;
6. Remember, you can output a command with up to 80 words;
7. Don't concatenate landmarks with verbs that contain location information. Only use verbs like "go to" and "find".
This is the path that you need to command the robot to walk through, recording the name of the nearest object at intervals:
Target name: {Target}
Things around every point of the trajectory: {Object1}, {Object2}, {Object3}.
Now, based on the information above, you need to clearly describe the trajectory given to a navigation robot so that it can follow the path after understanding your description.
Describe the important information and summarize it into a paragraph of 80 words or less.
Now, give your instructions directly.

For this task, we define two types of instructions: instructions providing only landmarks and goals, and instructions providing additional movement sequences. When generating instructions providing additional movement sequences, we provide the movement sequences in the prompt as follows: Actions that the robot should take at each point: {Action1}, {Action2}, {Action3}, $\cdots$.

D.2 Self-refinement

- Bring me a little touch of nature that can thrive under indoor care and brighten up the space. (BEFORE: plants, plant, fruit, cloth; AFTER: plants, plant)
- It's been a tough day, could you find me something to support me while moving around? (BEFORE: chair, wheelchair, armrest, table; AFTER: chair, wheelchair, armrest)
- Help me find something to neatly pour hot water. (BEFORE: pot, table, teapot; AFTER: pot, teapot)
- After a long day, it's time to wind down and find a place to rest my eyes. (BEFORE: curtain, chair, sofa, books, bed; AFTER: curtain, chair, sofa, bed)
- I'm in the mood to choose an outfit; find me an area where I can store clothes neatly. (BEFORE: closet, cloth; AFTER: closet)

Using LLMs to assess the viability of their own predictions is becoming an increasingly important procedure in problem-solving (Shinn et al., 2023; Madaan et al., 2023; Paul et al., 2023; Yao et al., 2023). In the task of completing abstract instruction, GPT-4 (OpenAI, 2023) outputs possible objects for each instruction. Then we use GPT-4 to self-refine its output. In the above examples, the content inside the brackets represents the desired objects. BEFORE indicates outputs without self-refinement, while AFTER represents outputs with self-refinement. We can conclude that self-refinement improves the initial generation.

D.3 Prompts for Instruction Parsing

We design different prompts for decoding instructions as follows.

D.3.1 Goal-conditioned navigation given a simple instruction

Imagine that you are a very intelligent service robot.
You receive an instruction from the user: {Instruction}
You need to figure out what object the user needs, and then output it. Remember, answer in a short statement, because you can only choose one item.
Your output is:

D.3.2 Completing abstract instruction

> Imagine that you are a very intelligent service robot.
> You will receive an instruction from the user, and the only things you can provide to the user are listed as below {Option1, Option2, · · · }.
> The instruction you have just received is {Instruction}.
> You need to choose the item you can provide that best fits the user's needs. Remember, answer in a short statement, because you can only choose one item.

D.3.3 Step-by-step instruction following

> Given an instruction, you need to extract the landmarks in the instruction and sort them in the order in which they appear in the realistic navigation (not in the order they appear in the instruction).
> Requirement 1: Extract all landmarks in the instruction.
> Requirement 2: Do not generate landmarks that are not in the instruction.
> Requirement 3: Print the landmarks in sequence.
> Requirement 4: Don't put anything other than a landmark on the landmark line.
> For example:
> Instruction: First, start at the Curtain, then walk along until you see the Plants, and continue heading straight. When you reach the Fridge, take a slight turn towards the right, and just a bit beyond it, you should see the Monitor.
> Landmarks: 1. Curtain; 2. Plants; 3. Fridge; 4. Monitor.
> Now, you are given an Instruction: {Instruction}
> Landmarks:

E Instruction Samples

We provide some generated instructions for different tasks.

E.1 Goal-conditioned navigation given a simple instruction

- Locate the nearest monitor.
- Robot, move towards the knife.
- Approach the chess set nearby.
- Walk towards a closet in your vicinity.
- Seek out any visible book in the area.
- Please locate and walk to the desired medicine container.
- Make your way to the nearest table.
- Seek out an emergency kit in your proximity.
- Look for a plant and make your way towards it.
- Proceed to the nearest bed in the area.

E.2 Completing abstract instruction

- I'm thinking about doing some screen-based work, find me a device that will help me with that.
- I'm about to make a sandwich and need a helpful utensial to get the job done efficiently.
- Feeling inspired to make a cozy, warm meal for tonight. Could you fetch me the key vessel we'll be using to cook it in?
- In the mood for some strategic fun, can you locate a board game with elegant pieces engraved with squares on it?
- Find me a place where I can enjoy my meal more comfortably.
- Locate a snack that contains vitamins and is both sweet and refreshing.
- It's getting a bit chilly, could you find me something cozy to wear?
- I'm in need of something designed to hold a warm beverage for brewing.
- I've been on my feet all day, and now I ought to locate a cozy spot to recharge.

- Feeling a tad warm, could you look for something that keeps food and drinks chilled?

E.3 Step-by-step instruction following

- Robot, please start by heading towards the teapot. As you continue, you will pass by a drinking machine multiple times. After that, you'll see an alcoholic drink in the area. Keep going until you find a cake. Make sure you follow the cakes - there will be a few. Your final destination is the tongs. Keep following this path, and you'll reach the tongs. Remember, the keynote of our journey is teapot, drinking machines, alcoholic drink, cakes, and then tongs. Good luck!
- Robot, I need you to follow this path to reach the Tongs. Begin by moving forward towards the Teapot, continue going straight, passing the Drinking Machine and Alcoholic Drink. Keep moving forward until you reach the Cake and then take a slight left. Proceed forward next to more Cakes, then take a right turn as you keep forward. After that final turn, you will see the Tongs in front of you. Please stop upon reaching the Tongs.
- Robot, please move forward along this trajectory. As you proceed, you will first notice a monitor nearby. Keep going straight and focus on your path. Then, after approaching the plants, immediately stop. That is your final destination, which we referred to as Plants. Remember to take actions such as Forward and STOP only when necessary. Good luck on your journey!
- Hey robot, first, I need you to move forward and head towards the door. Make sure they're on your side. Continue along the path, and you'll eventually reach our desired target - the drinks. Keep focused, nice, and simple - locate the door, then finally the drinks.

F Human Scoring

We sample 60 instructions generated by different LLMs and ask 100 people to rate their plausibility. We have developed detailed scoring guidelines for raters to reference. The maximum score an instruction can receive is 10 points.

Scoring criteria for the Simple task:

- Clarity and fluency of expression: 1 point
- No mention of specific locations (e.g., "Please go to the kitchen and get a plate" is incorrect): 2 points
- Mention of provided object names is allowed: 2 points
- Adherence to human expression habits and grammatical correctness: 2 points
- The command aligns with your vision of using a robot in the future: 3 points

Scoring criteria for the Abstract task:

- The target object should not be directly mentioned: 2 points
- Objects not listed in the given set should not be mentioned: 1 point
- Specific locations of objects should not be mentioned (e.g., "I'm hungry, please go to the kitchen" is incorrect): 1 point
- Instructions should not be overly simple: 1 point
- Adherence to human expression habits and grammatical correctness: 2 points
- The command aligns with your vision of using a robot in the future: 3 points

Scoring criteria for the Step-by-step task:

- Correct inclusion of all required elements:
  - For tasks without actions: Correct inclusion of all landmarks: 2 points; No inclusion of unfounded actions: 2 points
  - For tasks with actions: Correct inclusion of all actions and landmarks: 4 points (2 points each for actions and landmarks)
- Brevity: The description is within 80 words: 1 point

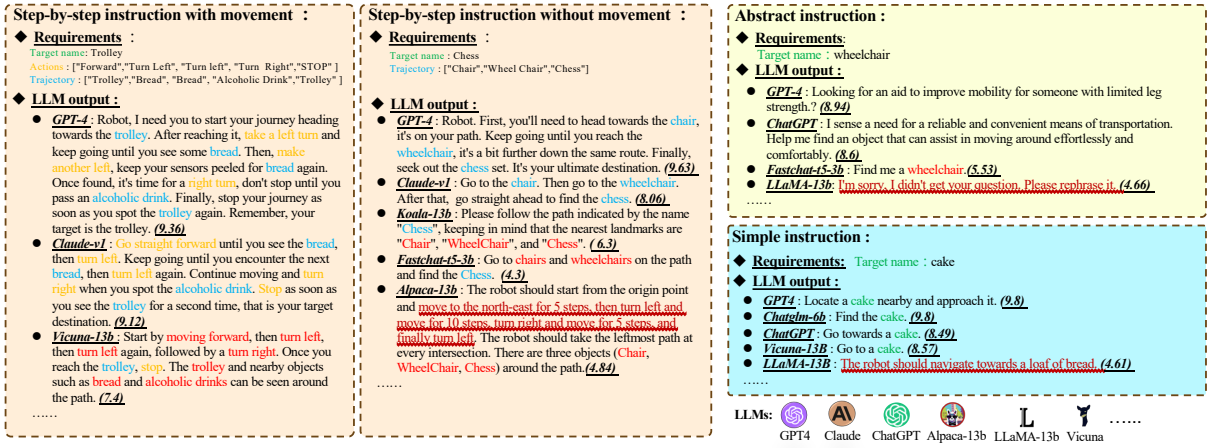

Figure 11: Examples of instruction generated by different LLMs under three different tasks.

Table 8: Humans score LLMs on a scale of 0∼10. M represents movement sequences. AVG indicates the average score.

	Simple	Reasoning	Step-by-step w/o M	Step-by-step w/ M	AVG
GPT-4 (OpenAI, 2023)	**9.80**	**8.94**	**9.63**	**9.36**	**9.43**
Claude-v1 (Bai et al., 2022)	8.26	8.39	8.06	9.12	8.46
ChatGPT (Ouyang et al., 2022)	8.49	8.60	8.00	8.08	8.29
Vicuna-13b (Chiang et al., 2023)	8.57	7.42	6.80	7.40	7.55
Vicuna-7b (Chiang et al., 2023)	8.26	7.10	7.89	6.60	7.46
Koala-13b (Geng et al., 2023)	8.57	7.30	6.30	6.44	7.15
Chatglm-6b (Du et al., 2022)	**9.80**	6.10	5.50	5.85	6.81
Fschat-t5-3b (Chiang et al., 2023)	9.10	5.53	4.30	5.33	6.07
Alpaca-13b (Taori et al., 2023)	8.50	4.84	4.84	5.84	6.01
LLaMA-13b (Touvron et al., 2023)	4.61	4.66	2.71	5.77	4.44
AVG	8.40	6.89	6.40	6.98	-

- Adherence to human expression habits and grammatical correctness: 2 points

- Alignment with future use vision: The command aligns with your vision of using a robot in the future: 3 points

These guidelines are designed to ensure a uniform and fair assessment of the instructions' suitability and effectiveness for guiding the robot's actions in various task scenarios. As shown in Table 8, GPT-4 (OpenAI, 2023) performs best overall. For the simple task, Chatglm-6b (Du et al., 2022) performs as well as GPT-4. The evaluations of the quality of instructions are collected from 100 people, who are undergraduate and graduate students from the university and ages from 18 to 30. They are able to evaluate the dataset accurately. Examples of some of the higher-scoring and lower-scoring cases are presented in Figure 11.

G Qualitative Results

G.1 Compare with previous simulators

We present qualitative comparisons between different simulators in Figure 12. These visual comparisons underscore the superior realism and intricacy of our simulator.

G.2 Environment details

We provide more figures of the environment details in different scenes in our simulator as in Figure 13. We can see that the illumination, reflections, and shadows are close to the physical world. Besides, our simulator also contains dynamic steam and water, following the principles of the physical world.

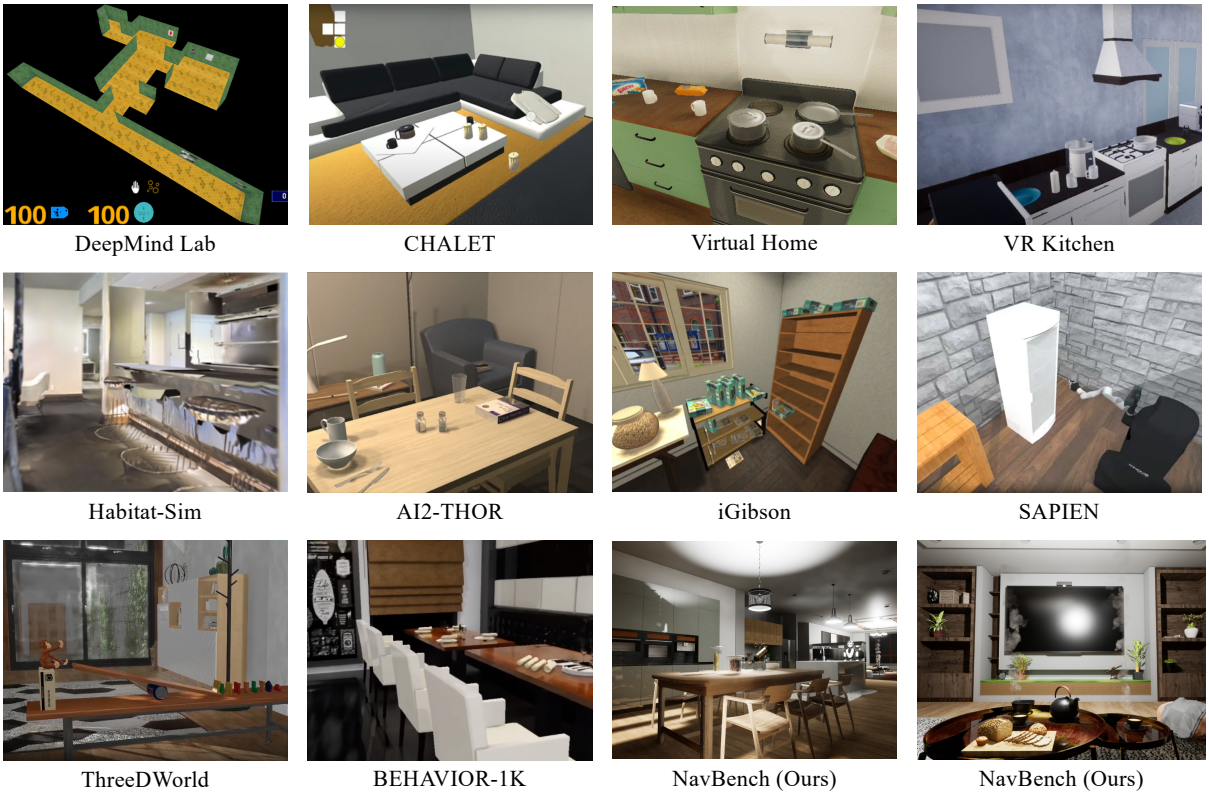

Figure 12: Compare with existing simulators. The last two pictures are from our simulator. Our environment is more elaborate, and the lighting in our simulator is more realistic.

Figure 13: Elaborate environment details in different scenes in our simulator.