Robotic Action Planning Using Large Language Models

César Bastos da Silva
Advanced Robotics Laboratory
State University of Campinas
Campinas, Brazil
c264521@dac.unicamp.br

Juan Luis Barraza Ramírez
State University of Campinas
Campinas, Brazil
j272583@dac.unicamp.br

Niederauer Mastelari
State University of Campinas
Campinas, Brazil
niede@unicamp.br

Roberto Lotufo
NeuralMind.ai
Campinas-SP, Brazil
roberto@neuralmid.ai

Jayr Pereira
Universidade Federal do Cariri
Juazeiro do Norte-CE, Brazil
jayr.pereira@ufca.edu.br

Eric Rohmer
Advanced Robotics Laboratory
State University of Campinas
Campinas, Brazil
rohmer@unicamp.br

2024 Latin American Robotics Symposium (LARS) | 979-8-3315-0880-7/24/$31.00 ©2024 IEEE | DOI: 10.1109/LARS64411.2024.10786412

***Abstract*—This study[1] explores the integration of artificial intelligence (AI) and large language models (LLMs) in robotics, focusing on task planning and execution. We implemented a Reasoning and Acting (ReAct) system within a simulated environment, utilizing a humanoid robot equipped with various tools for searching, locomotion, vision, manipulation, and communication. The robot operates based on natural language prompts and utilizes the LangChain framework to facilitate interaction with the LLM. We conducted experiments to evaluate the robot's performance on tasks requiring short-term, medium-term, and long-term memory. Short-term memory tasks involved single-step actions, medium-term memory tasks needed the completion of two-step sequences, and long-term memory tasks involved three or more steps. The results demonstrated a high success rate for short-term tasks, while performance for medium and long-term tasks varied depending on the number of steps involved. Our findings highlight both the challenges and potential of using AI and LLMs in robotic task planning. The results demonstrate the promise of enhancing robotic capabilities to perform complex tasks through natural language instructions.**

***Index Terms*—ReAct, Task Planning, LLM, AI, Robotics Actions**

I. Introduction

With the advancement of artificial intelligence (AI) and large language models (LLMs), various applications have emerged across fields such as robotics, intelligent virtual assistants, market research, transcription, and more. AI's integration into robotics has opened new horizons and introduced new challenges, making its application in areas like navigation and manipulation increasingly essential. The advent of LLMs is ushering in a new era of decision-making and planning capabilities [1].

In recent years, numerous studies have focused on how AI and LLMs can assist in planning robotic actions for specific tasks due to their exceptional capability in context learning. These emerging studies aim to deploy robots to perform particular tasks, enhancing human capabilities in social service roles. Research presented by Wu et al. [1] explores how to perform multilevel decomposition for complex tasks requiring long-term memory, as these tasks often involve multiple steps for the robot to achieve the desired outcome. Companies like Microsoft have also delved into these investigations, where users can assign tasks to robots with diverse capabilities such as flight, manipulation, and mobility to achieve a common goal. For instance, works like in Vemprala et. al [2] demonstrate various applications developed by their Autonomous Systems and Robotics group. Additionally, the interaction between AI and robotics is detailed further, emphasizing the importance of obtaining the position and orientation of objects and programming the robots' locomotion. Finally, the robots communicate with ChatGPT to dialogue before executing the necessary actions.

Other works such as [3] and [4] continue in this direction, utilizing natural language for robotic tasks through planning representations and state abstractions. Task planning relies on the knowledge of the environment; for this, works such as Wang et. al [5] rely on the details in the use of structured data to enhance LLM capacities to predict state transitions. However, real environments have many dynamic factors, necessitating a robust and reasonable decision-maker. Each action transforms the environment state; thinking in this detail, Yao et. al [6] synergize reasoning with the action using LLMs, leading to superior performance with interpretative decision traces.

Inspired by these studies, we propose an application wherein a robot, using natural language and specifically the ReAct framework, can reason and then act to fulfill the tasks requested by the user. The proposed React method was validated in a simulated environment with pick-and-place tasks to assess LLM capabilities in achieving success. The tasks varied in length, ranging from 1 to 3.

To summarize, the contributions of this study are as follows:

- React–Tools representation as the robot capabilities, such as vision, manipulation, and locomotion.

[1]The work is available on: https://github.com/cesarbds/LLM_Planner

- Multiple LLM utilization, one specific for decision-making and a second LLM to summarize key information, preventing overload in the decision-making LLM.
- Environment description with structured data, with metric and semantic data.

II. Related Work

A. Robotic task planning

Robotic task planning is a crucial area of research in robotics, focusing on developing algorithms and systems that enable robots to autonomously plan and execute complex tasks. One prominent method is Task and Motion Planning (TAMP), which integrates high-level task planning with low-level motion planning. Kaelbling and Lozano-Pérez [7] introduced the Hierarchical Planning in the Now (HPN) framework, interleaving task, and motion planning to handle uncertainty in dynamic environments. Their work laid the foundation for many subsequent TAMP methods, significantly improving efficiency and robustness in real-world applications. Srivastava et al. [8] further extended this approach by integrating symbolic planning for task-level decisions with motion planning for generating feasible trajectories, enabling robots to execute long-horizon tasks efficiently.

With the advent of machine learning, particularly deep learning, learning-based approaches to robotic task planning have gained prominence. Levine et al. [9] demonstrated the effectiveness of deep learning in end-to-end training of robotic manipulation tasks, showing that robots can learn to perform complex tasks such as grasping and object manipulation directly from raw sensory inputs.

B. Robotic task planning with Large Language Model

A Large Language Model (LLM) is a neural network equipped with deep language comprehension, reasoning, human-like text generation capabilities, and strong analytical abilities, rendering it indispensable across various fields [10]. In robotics, there is growing interest in integrating LLMs into robot system policies. Pre-trained models, as demonstrated by Singh et al. [11] with GPT-3, introduce novel prompting schemes that go beyond natural language conditioning by incorporating programming language structures to define tasks based on specified actions. Li et al. [12] propose an adaptive GPT-based system capable of adjusting plans in response to new user requests and task guidelines.

However, ensuring the feasibility and correctness of generated plans remains a challenge. Zhou et al. [13] address this issue with a three-step framework, including a plan refinement stage with a validator that verifies action plan correctness and provides feedback to the LLM planner.

Closed-source models are often favored for their superior reasoning capacity compared to open-source LLMs. To bridge this gap, efforts are focused on refining small-scale open-source LLMs using task planning data to enhance their planning capabilities. Wu et al. [14] propose the Task Planning Agent (TaPA), which generates executable plans aligned with visual perception models and scene objects, achieving higher success rates than state-of-the-art LLMs like GPT-3.5. Wu et al. [1] introduce the Multi-Level Decomposition Task Planning (MLDT) method, which breaks down tasks recursively to enhance LLM planning abilities, particularly for tasks requiring manageable steps. Li et al. [15] utilize word embeddings to represent goals and observations, employing a pre-trained LLM as a policy network fine-tuned with active data gathering to predict actions effectively.

The present work integrates open-source pre-trained models, harnessing their inherent reasoning abilities through the ReAct framework for task completion.

III. Methodology

The application presented in this work was implemented using a robot similar to Pepper[2] within a simulation 3D environment created in CoppeliaSim [16]. This simulated environment represents a house with various rooms such as a kitchen, living room, bedroom, bathroom, dining room, patio, hallway, and entrance ramp. This study aims to assign tasks to the robot and ensure it can complete them. To achieve this, we proposed using a Reasoning and Acting (ReAct) system, wherein the agent determines which tool to use for the given task. The tools utilized include Search, Locomotion, Vision, Manipulation with Pick and Place, and Communication.

A. Environment

As mentioned, the environment developed in CoppeliaSim includes ten rooms through which the robot and various users can move. This scenario involves 41 objects, each with properties such as *Pickable* and *Placeable*, indicating whether the robot can manipulate the object. Additionally, objects have the *Surface* property, which informs the robot that they are surfaces where other objects can be placed but cannot be manipulated themselves. The environment includes four agents: three users and the robot. Figure 1 shows the overall environment used.

All necessary information was provided to the robot to perform the requested tasks. The file organization uses the concept of a multi-agent system (MAS) [17], initially almost just with one agent, but ready to evolve. The environment description follows the concept brought by [5]. This information includes the positions and orientations of rooms, objects, and agents, as well as certain object properties needed for the robot to carry out tasks using the data from three different JSON files:

- *rooms.json*, describe physically the environment;
- *objects.json*, describes object localization and properties;
- *agents.json*, describe robot and human agents localization and properties.

[2]https://www.aldebaran.com/en/pepper

Fig. 1: Environment simulating a house, with kitchen, living room, dining room, bathroom and bedroom

Additionally, after any action is performed in the environment, these JSON files are automatically updated. For instance, when an object is picked up and placed in a different room, its localization is updated.

B. Agent

The robot utilized is a type of humanoid robot with wheels, enabling movement through two motors at its base, operating via a differential platform. This robot, similar to the Pepper robot, requires velocity commands for navigation and torque commands for its robotic arms to perform Pick & Place activities. It is also equipped with a vision sensor to detect nearby objects and confirm if the requested object has been grasped. For simplification purposes, the robot has been provided with a pre-mapped environment, analogous to giving the robot a map to avoid the simultaneous localization and mapping (SLAM) process, typically required in robotics [18].

C. Reasoning and Acting (ReAct)

ReAct framework expands the agent's range of actions to include both traditional actions A and elements from a language space L. In this setup, actions $\hat{a}_t$ from the language space, referred to as thoughts or reasoning traces, do not directly affect the environment or result in observable feedback. Instead, they aim to gather useful information by reasoning over the current context c_t. This updated context $c_{t+1} = (c_t, \hat{a}_t)$ helps the agent plan future actions or refine its understanding. The thoughts can serve various functions, like breaking down task goals, devising action plans, incorporating relevant knowledge, extracting key details from observations, monitoring progress, and adapting plans for unexpected situations.

In this context, the tools represent the space of actions, with the language model independently governing the asynchronous initiation of its thoughts and actions. The decision-making and reasoning, as previously noted, are integrated into a LLM with ReAct, aiming for flexibility, robustness, and controllability. However, its performance heavily relies on prompt quality.

The *ReAct_Prompt*, describes to the LLM its purpose and also contains comprehensive information about the tools available to the robot. The prompt emphasizes the importance of correctly using the tools based on the properties of the objects.

D. Tools

In this Section, we explain in detail the tools provided to the robot to perform the ReAct process.

1) Search: This tool allows the robot to search through the respective JSON files to determine the location of the objects requested by the user and identify the room where the object is located. The search is powered by an LLM, which is responsible for verifying the Planner request and then providing summarized information from the files, to attenuate the length of information passed to the planner. The input gathers the structured data description, with all the information from the JSON files, and its command. This ensures that the search tool can retrieve comprehensive information about the tools and the environment, working as a summarizing LLM.

2) Locomotion: This tool dislocates the robot to a desired localization, room, or static object. It directly communicates with the robot's wheels to facilitate movement. As a differential drive robot, the information sent involves the speed of each wheel. A topological map was constructed using the center of each room and points located in front of static objects, providing specific targets for navigation.

3) Vision: This tool is employed once the robot is close to the requested object. It enables the agent to identify objects within its field of vision, allowing the subsequent use of the manipulation tool. It simulates an object recognition technique at first.

4) Manipulation: This concept includes two tools: Pick and Place. The Pick tool allows the robot to grasp objects within its field of action, while the Place tool enables it to place the grasped object at a desired location. This tool is essential for delivering the requested object to the user.

5) Communication: This tool provides feedback to the user regarding the completion of the task. It delivers a summary of the robot's performance during the task.

E. LangChain

To facilitate communication between all these tools and a large language model, we used the LangChain framework[3]. LangChain was chosen for its capabilities in supporting the ReAct process in conjunction with Groq API, an inference engine that enabled us to use LlaMA3-70B as the LLM, in the react and the tool. LangChain's integration of the ReAct framework allows the robot to reason and act according to the user's task instructions through the configuration of function parameters.

[3]LangChain is an open source framework for developing applications using large language models. See: https://www.langchain.com/

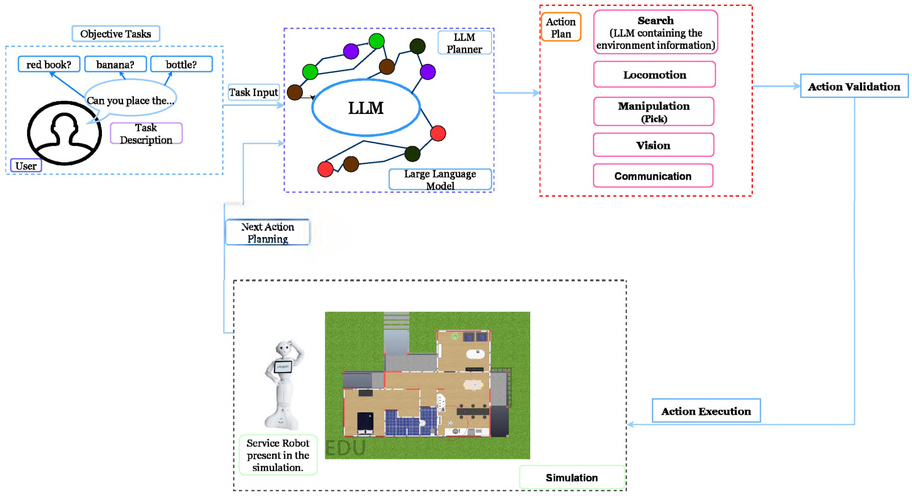


Fig. 2: Proposed Architecture, where the robot interacts with the environment and controls the usage of tools.

F. Expected behavior

Bringing everything together, it is expected that the robot act as follows:

- The user provides a task to the robot.
- The robot uses the Search tool to locate the requested object.
- If the robot is not in the same room as the object, the Locomotion tool is used to navigate to the correct room.
- Once nearby, the Vision tool identifies the object.
- The Pick tool grasps the object.
- If the robot is not close to the desired place, the Locomotion tool is used to navigate to the correct room.
- The Place tool positions the object as instructed by the user.

This can be seen graphically in Figure 2.

IV. Experiments

For the experiments, we decided to categorize the queries based on the number of tasks required for the agent to complete, as an increased number of tasks demands the agent to be capable of remembering each point of the requested tasks. Therefore, we divided the experiments into three categories: short-term memory (one task at a time), medium-term memory (two tasks), and long-term memory (three or more tasks). The initial environment condition is the same for all tests and each task is randomly selected from the objects and places present in the environment. For the latter, we conducted a test with five tasks to analyze the robot's performance.

A. Short-term memory tasks

For the short-term memory tasks, where only one task was required, we conducted 50 different simulations, each with the robot performing a single task. An example of such a task is, *"Can you put the laptop on the countertop?"* In this task, the ReAct process was as follows:

Action 1
Thought: Do I need to use a tool? Yes
Reasoning: I need to use the Search tool to get information about the environment and locate the Countertop and laptop.
Action: Search
Action Input: Where is the Countertop and laptop?

Tool output: The Countertop is in the Kitchen, and the laptop is on top of the small table in the living room.

Action 2
Thought: I need to use the Locomotion tool to move to the Small Table.
Action: Locomotion
Action Input: Small Table

Tool output: Now you are in front of the Small Table

Action 3
Thought: I need to use the Vision tool to see the objects around me.
Action: Vision
Action Input:

Tool output: You have the following objects: Small Table, Walls, Sofa, TV_Stand, TV, laptop, Indoor Plant

Action 4
Thought: I need to use the Manipulation_Pick tool to pick up the laptop.
Action: Manipulation_Pick
Action Input: laptop

Tool output: Now you have the laptop object with you

Action 5
Thought: I need to use the Locomotion tool to move to the Countertop.
Action: Locomotion
Action Input: Countertop

Tool output: Now you are in front of the countertop

Action 6
Thought: I need to use the Manipulation_Place tool to place the laptop on the Countertop.
Action: Manipulation_Place
Action Input: laptop, Counter top

Tool output: You just placed the laptop on top of the counter top

Action 7
Thought: Do I need to use a tool? No
Final Answer: The laptop is on Countertop.

As shown in the ReAct process, the expected outcome was achieved. To evaluate our reasoning and acting process, we scored each successful task completion as 1, and only one failure to complete a task was analysed. The robot had a 98% of success rate as shown in Figure 3. Most of the experiments had 5 to 10 iterations, the peak is when it reached the maximum of iterations.

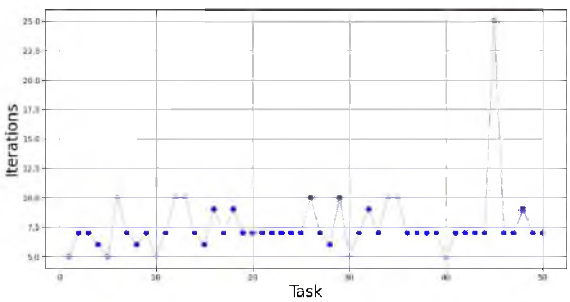


Fig. 3: Number of Iterations for short-term memory tasks

B. Medium-term memory tasks

For the medium-term memory tasks, involving two tasks, we conducted also 50 different simulations, each requiring the robot to complete two tasks. An example of such a task is: *"Can you put the bowl on the bed, then pick the red book and put it on the small table?"*. Using the same procedure as before, we observed a performance rate of 88%. This result is logical, as the robot tends to forget some tasks as more tasks are added in a single prompt, and finishes the query before completing it. Figure 4 shows a plot of the robot's performance.

C. Long-term memory tasks

For the long-term memory tasks, we conducted two different sets of experiments, each of which will be discussed in the following subchapters.

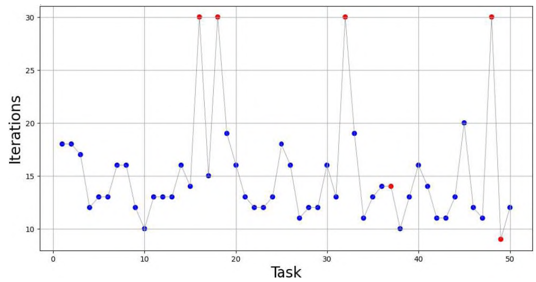
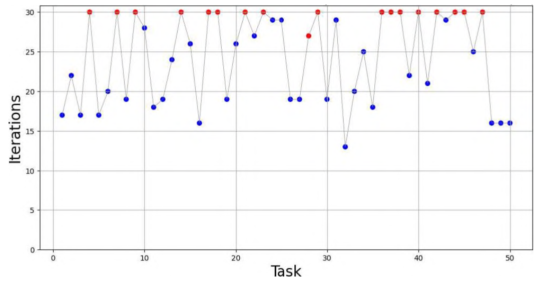


Fig. 4: Number of Iterations for medium-term memory tasks. In blue completed Task, in red incomplete task.



(a) Number of Iterations for long-term memory tasks. In blue completed Task, in red incomplete task.

Fig. 5: Long–term memory results.

1) Long-term memory tasks: 3 tasks: For these tasks, we used the same procedure as mentioned previously, but with three tasks given simultaneously in a single prompt. We conducted five different simulations and achieved a performance rate of 64%. We believe the difference in performance compared to the medium-term memory tasks is due to the bad use in some point of the search tool and also fully related to the memory problem. Figure 5 shows the robot's performance, where it is possible to analyze one query that was not finished, but the LLM configured as completed. Another significant finding is that some individuals fail to complete tasks; the LLM itself might also fail to complete a task if it gets stuck in a loop.

2) Long-term memory tasks: 5 tasks: In this case, we conducted a single simulation to test how it will perform with more tasks. The following prompt was given: *"Can you put the bowl on the countertop, then pick the red book and put it in the kitchen drawer, then pick the banana and put it on the bed, then pick the plate and put it on the rack, then pick the laptop and put it on the small table?"*. Table I illustrates the process followed by the agent to complete the five tasks. As shown, the robot completed the first task without any issues. It started the second task but got lost while attempting to simultaneously begin the third task. Consequently, it completed the third task but not the second. The robot managed to complete the fourth task without much trouble, and then it considered the job done, without attempting the fifth task. Therefore, with this single simulation, the robot was able to perform 60% of the query.

TABLE I: Reasoning for long-term memory tasks

Search	
Locomotion	Countertop
Pick	Bowl
Vision	
Place	Bowl, Countertop
Locomotion	Rack
Vision	
Pick	Red book, kitchen drawer
Locomotion	Kitchen drawer
Place	Red book, kitchen drawer
Locomotion	Kitchen drawer

Pick	Banana
Place	Banana, bed
Locomotion	Bed
Place	Banana, bed
Locomotion	Rack
Vision	
Locomotion	Kitchen
Vision	
Pick	Plate
Place	Plate, rack
Locomotion	Rack
Place	Plate, rack

D. Qualitative Evaluation

Even though the system had a high success rate, the task sequence was not optimal. To further analyze this, the system was described to ChatGPT4o and asked what could be enhanced, as a larger LLM its reasoning and sequence conception is better. Here is an example:

Sequence 1: The robot is supposed to move to the living room to pick up the red book, it should have realized it was already in the living room after placing the green book on the Small table.

V. Conclusion

This work proposes a robotic action planning system based on Large Language Models (LLMs) utilizing the ReAct concept. By leveraging prompt engineering, the LLM is informed of its functionality and the capabilities of the tools at its disposal. The tasks involved in this study were randomly selected pick-and-place operations. Remarkably, for single tasks, the system achieved a highsuccess rate. However, the performance dropped when multiple tasks were introduced, indicating a limitation in the current system's ability to handle compound tasks.

This finding underscores the significant potential of LLMs in the robotics industry, paving the way for novel research fields and enhancing reasoning capabilities in task execution. Additionally, the approach of providing detailed environmental descriptions shows promise, as it enables the LLM to interact more effectively with its surroundings. Overall, the system demonstrated successful task execution and achieved an expected success rate, comparable to more complex and state-of-the-art approaches such as MLDT.

VI. Future Work

This work paves the way for new avenues of research. A critical next step is to implement this method in a more dynamic experimental environment, requiring advanced object detection and robust robot motion algorithms. Additionally, evaluating the experimental setup with parallel algorithms is essential, as minimizing the time taken for each step is crucial to avoid falling into the "canny valley."

Future enhancements, particularly in memory management, could significantly improve the system's performance and its ability to handle more complex sequences of tasks. Integrating this system with cognitive frameworks that manage both long-term and working memory will enable it to effectively execute prolonged and intricate tasks.

Acknowledgment

This project was supported by the Ministry of Science, Technology and Innovation, with resources from Law No. 8,248, of October 23, 1991, within the scope of PPI-SOFTEX, coordinated by Softex and published in Cognitive Architecture (Phase 3), DOU 01245.003479/2024-10.

References

[1] Y. Wu, J. Zhang, N. Hu, L. Tang, G. Qi, J. Shao, J. Ren, and W. Song, "Mldt: Multi-level decomposition for complex long-horizon robotic task planning with open-source large language model," *arXiv preprint arXiv:2403.18760*, 2024.

[2] S. H. Vemprala, R. Bonatti, A. Bucker, and A. Kapoor, "Chatgpt for robotics: Design principles and model abilities," *IEEE Access*, 2024.

[3] L. Wong, J. Mao, P. Sharma, Z. S. Siegel, J. Feng, N. Korneev, J. B. Tenenbaum, and J. Andreas, "Learning adaptive planning representations with natural language guidance," *arXiv preprint arXiv:2312.08566*, 2023.

[4] A. Peng, I. Sucholutsky, B. Z. Li, T. R. Sumers, T. L. Griffiths, J. Andreas, and J. A. Shah, "Learning with language-guided state abstractions," *arXiv preprint arXiv:2402.18759*, 2024.

[5] R. Wang, G. Todd, Z. Xiao, X. Yuan, M.-A. Côté, P. Clark, and P. Jansen, "Can language models serve as text-based world simulators?" *arXiv preprint arXiv:2406.06485*, 2024.

[6] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "React: Synergizing reasoning and acting in language models," *arXiv preprint arXiv:2210.03629*, 2022.

[7] L. P. Kaelbling and T. Lozano-Párez, "Hierarchical task and motion planning in the now," *IEEE International Conference on Robotics and Automation*, 2011.

[8] S. Srivastava, E. Fang, L. Riano, R. Chitnis, S. Russell, and P. Abbeel, "Combined task and motion planning through an extensible planner-independent interface layer," *International Conference on Robotics and Automation*, 2014.

[9] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *International Journal of Robotics Research*, 2016.

[10] Y. Yao, J. Duan, K. Xu, Y. Cai, Z. Sun, and Y. Zhang, "A survey on large language model (llm) security and privacy: The good, the bad, and the ugly," *High-Confidence Computing*, p. 100211, 2024.

[11] I. Singh, V. Blukis, A. Mousavian, A. Goyal, D. Xu, J. Tremblay, D. Fox, J. Thomason, and A. Garg, "Progprompt: Generating situated robot task plans using large language models," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 11 523–11 530.

[12] B. Li, P. Wu, P. Abbeel, and J. Malik, "Interactive task planning with language models," *arXiv preprint arXiv:2310.10645*, 2023.

[13] Z. Zhou, J. Song, K. Yao, Z. Shu, and L. Ma, "Isr-llm: Iterative self-refined large language model for long-horizon sequential task planning," *arXiv preprint arXiv:2308.13724*, 2023.

[14] Z. Wu, Z. Wang, X. Xu, J. Lu, and H. Yan, "Embodied task planning with large language models," *arXiv preprint arXiv:2307.01848*, 2023.

[15] S. Li, X. Puig, C. Paxton, Y. Du, C. Wang, L. Fan, T. Chen, D.-A. Huang, E. Akyürek, A. Anandkumar *et al.*, "Pre-trained language models for interactive decision-making," *Advances in Neural Information Processing Systems*, vol. 35, pp. 31 199–31 212, 2022.

[16] E. Rohmer, S. P. Singh, and M. Freese, "V-rep: A versatile and scalable robot simulation framework," in *2013 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2013, pp. 1321–1326.

[17] A. A. A. Rasheed, M. N. Abdullah, and A. S. Al-Araji, "A review of multi-agent mobile robot systems applications," *International Journal of Electrical and Computer Engineering*, vol. 12, no. 4, pp. 3517–3529, 2022.

[18] N. Hughes, Y. Chang, and L. Carlone, "Hydra: A real-time spatial perception system for 3d scene graph construction and optimization," *arXiv preprint arXiv:2201.13360*, 2022.