LLM-Empowered Embodied Agent for Memory-Augmented Task Planning in Household Robotics

Marc Glocker[1,2], Peter Hönig[1], Matthias Hirschmanner[1], and Markus Vincze[1]

***Abstract*— We present an embodied robotic system with an LLM-driven agent-orchestration architecture for autonomous household object management. The system integrates memory-augmented task planning, enabling robots to execute high-level user commands while tracking past actions. It employs three specialized agents: a routing agent, a task planning agent, and a knowledge base agent, each powered by task-specific LLMs. By leveraging in-context learning, our system avoids the need for explicit model training. RAG enables the system to retrieve context from past interactions, enhancing long-term object tracking. A combination of Grounded SAM and LLaMa3.2-Vision provides robust object detection, facilitating semantic scene understanding for task planning. Evaluation across three household scenarios demonstrates high task planning accuracy and an improvement in memory recall due to RAG. Specifically, Qwen2.5 yields best performance for specialized agents, while LLaMA3.1 excels in routing tasks. The source code is available at: `https://github.com/marc1198/chat-hsr`**

***Index Terms*— Embodied AI, Task Planning, Memory Retrieval**

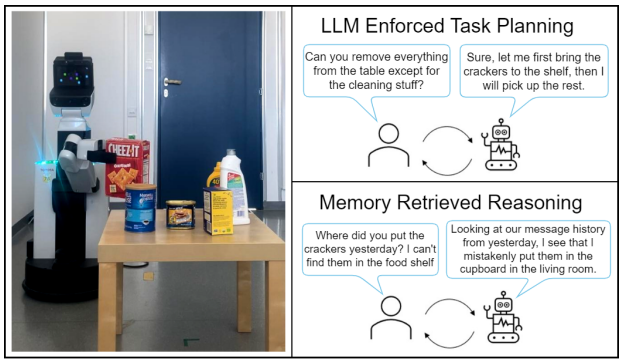


Fig. 1: Our LLM-driven robotic system autonomously plans tasks and retrieves past interactions to improve object handling, illustrated by LLM-enforced task planning and memory-retrieved reasoning in a household setting.

I. Introduction

Despite recent progress in robotics and artificial intelligence, robots still struggle to adapt flexibly to the diverse, dynamic situations of real-world environments, particularly in household settings [24]. While symbolic task planning with languages like the Planning Domain Definition Language (PDDL) [11] is effective in domains with fixed rules and predictable object categories, it lacks the adaptability required for open-ended household environments. In such settings, robots must deal with ambiguous user commands, detect novel or unstructured objects, and respond to constantly changing spatial configurations [24]. These limitations motivate our hypothesis that a modular LLM-driven system can enhance flexibility by leveraging natural language understanding, contextual reasoning, and memory-based adaptation. We provide a proof-of-concept implementation and assess its performance in real-world household tasks.

In this work, we present an embodied robotic system with an LLM-driven agent-orchestration architecture, where specialized software agents collaborate to address long-horizon household tasks. Recent advances in Large Language Models (LLMs) [13], [4], [15], [23], [5] have improved systems real-world understanding, enabling common-sense reasoning in human language and making them accessible to researchers. These advances combined with in-context learning [26] enable flexible embodied task planning by decomposing high-level commands, such as *"clear the dining table"*, into actionable steps based on detected objects [2], [7], [25], [9], [21]. By integrating Grounded Segment Anything Model (Grounded SAM) [17] and LLaMa3.2-Vision [4], our system creates grounded task plans. Unlike most other works, we address long-term operations by maintaining action and environment records, utilizing Retrieval-Augmented Generation (RAG) for efficient memory retrieval. Our approach enables the robot to autonomously organize and retrieve objects, interpret complex tasks, and provide updates on object locations, all while ensuring privacy through the use of offline LLMs and avoiding explicit model training. To illustrate the systems interaction, Fig. 1 shows an example of our system in action.

In summary, we present the following key contributions:

- A long-horizon task planner for household tasks leveraging in-context learning and offline LLMs.
- Use of RAG for efficient memory retrieval and object tracking.
- A modular agent-orchestration system that improves robustness and modularity.
- Evaluation of the system's performance in three real-world household scenarios.

This paper is structured as follows: Section II reviews related work in the areas of task planning and memory mechanisms. Section III details the proposed system architecture.

[1] Automation and Control Institute, Faculty of Electrical Engineering, TU Wien, 1040 Vienna, Austria {hoenig, hirschmanner, vincze}@acin.ac.tuwien.at

[2] AIT Austrian Institute of Technology GmbH, Center for Vision, Automation and Control, 1210 Vienna, Austria marc.glocker@ait.ac.at

https://doi.org/10.34749/3061-0710.2025.9

Section IV describes the experimental setup and household scenarios. Section V presents the results. Finally, Section VI concludes the paper and outlines directions for future work.

II. Related Work

In this section we discuss related work for action and task planning, as well as memory and knowledge base.

A. Action and Task Planning

Recent advancements in prompt engineering have improved the problem-solving capabilities of LLMs [26], [28], enabling the generation of structured plans without fine-tuning. Consequently, modern agent architectures leverage LLMs to dynamically react to execution failures [27], [7] and expand their context by retrieval [8] or external tools [19], [18]. However, LLMs lack an inherent understanding of a robot's physical abilities and real-world constraints. *SayCan* [2] addresses this by integrating value functions of pre-trained robotic skills to ensure feasibility, whereas Huang et al. [6] leverage LLMs to match high-level plans with low-level actions through semantic mapping. Some works treat LLMs as programmers rather than direct decision-makers: *Code-as-Policies* [9] and *ProgPrompt* [21] allow LLMs to generate structured code for robotic executions, enhancing flexibility but adding an execution layer.

Pallagani et al. [14] found that LLMs perform better as translators of natural language into structured plans rather than generating plans from scratch. This ensures feasible actions based on predefined world models [20], [10]. These approaches are particularly effective in highly controlled environments, but present challenges when applied to open-ended, dynamic household settings. Our work, instead, embraces flexible, dynamic task planning with in-context learning like shown in [25]. The approaches named, while effective for short-horizon tasks, do not track object positions over time. For long-horizon tasks that involve real-world dynamic conditions, a combination of task planning and a memory mechanism is required.

B. Memory and Knowledge Base

Long-horizon tasks require robust memory mechanisms. While LLM context windows keep expanding [23], using excessively large contexts in robotics is computationally inefficient. Instead, long-term memory retrieval, accessed only when needed, is a more viable solution. RAG [8] provides an efficient mechanism for narrowing context by querying a vast dataset and retrieving only relevant information. Additionally, scene graphs, used in approaches like SayPlan [16] and DELTA [10], offer structured memory that improves action verification and contextual reasoning. However, in unstructured and constantly changing environments, maintaining these graphs becomes challenging due to the need for complex automatic mechanisms or manual curation.

Our work explores the feasibility of a lightweight, fully natural language-driven approach using RAG as a memory mechanism. Inspired by ReMEmbR [1], our system incorporates temporal elements into the retrieval process, ensuring the robot tracks long-term changes in its environment. While using language-based memory retrieval introduces potential for increased errors compared to structured models like scene graphs, we aim to evaluate how well purely language-based memory retrieval performs in practical, dynamic household scenarios. This approach offers flexibility, adaptability, and reduces the need for explicit world modelling, making it more suitable for real-world applications.

III. Methodology

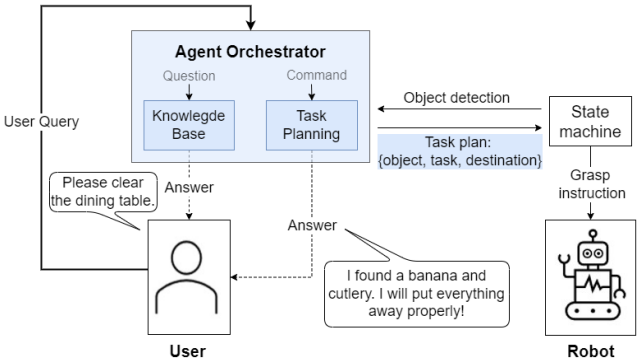


Fig. 2: The full pipeline, integrating long-horizon task planning. Newly introduced components are highlighted in blue.

Our system, coordinated by an agent-orchestration framework, combines task planning with RAG [8]. This chapter explains the individual components and their interaction.

Fig. 2 illustrates the overall pipeline. The focus of this work is the agent-orchestration system, which processes object detection and user requests to create a robot task plan. In the system, each agent uses an LLM with a specialized role. The task planning agent additionally is prompted with a chain-of-thought technique [26].

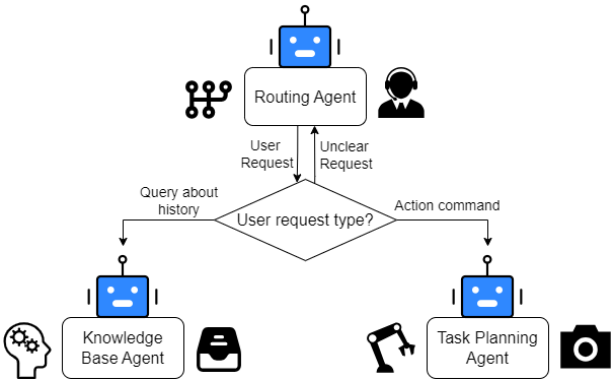


Fig. 3: The agent-orchestration architecture

The system architecture of the agent orchestrator, illustrated in Fig. 3, consists of:

1) A **routing agent**, responsible for analyzing incoming user requests.

https://doi.org/10.34749/3061-0710.2025.9

2) A **task planning agent**, handling commands that require the robot to perform actions.
3) A **knowledge base agent**, processing follow-up questions about previously handled objects.

When a user request arrives, the routing agent first analyzes it to determine its nature. The request is then categorized into one of three types:

1) **Action command:** If the robot is asked to perform an action, it is forwarded to the task planning agent.
2) **Query about history:** If it concerns previously handled objects, it is directed to the knowledge base agent.
3) **Unclear request:** If the request doesn't fit either category, clarification is requested before proceeding.

A. Task Planning Agent

The task planning agent receives frequent environmental updates via camera perception, encoded as a list of single objects. Grounded SAM [17] enables text-driven object detection and segmentation for the pipeline, while Vision Language Models (VLMs) generate natural language descriptions of the environment. Although VLMs alone can extract the object list for the LLM, Grounded SAM is essential for precise segmentation, which is critical for grasping tasks. Using the object list, the LLM processes the user request – which can be both expressed in high-level or low-level terms – and formulates tasks that best fulfill the command. The generated answer has to include a JSON string for an action following this structure:

1) **Objects involved** in the task.
2) **The destination** for placement tasks.

After the action is determined, the grasping process is initiated. We use the segmentation from Grounded SAM and the camera intrinsics to crop the depth image and project the depth crop to a 3D pointcloud of the respective object. To estimate a grasp approach vector, we feed the cropped object point cloud to Control-GraspNet [22], a pre-trained grasp estimator.

B. Knowledge Base Agent

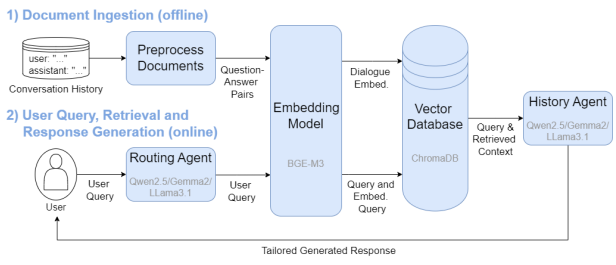


Fig. 4: RAG workflow for long-term question answering: Relevant past actions are retrieved from dialogue history, and the LLM generates responses based on the retrieved context.

The knowledge base agent is used for user inquiries regarding past robot actions, such as object locations or task completion status. These queries require access to long-term memory, for which RAG has proven most effective, as discussed in Section II. Fig 4 illustrates the RAG workflow, comprising two key steps:

1) **Document Ingestion:** Input data, such as conversation history, is preprocessed, split into smaller chunks (each representing a question-answer pair), and converted into high-dimensional vectors using an embedding model. These embeddings are then stored in a vector database for efficient retrieval.
2) **User Query, Retrieval, and Response Generation:** User queries are embedded using the same model and are matched against the stored vectors to retrieve the most relevant context. This context is then provided to the LLM, which generates a response tailored to the user's query.

To enable chronological reasoning, essential for tracking object movements over time, we augment RAG with a time stamp for each question-answer pair.

IV. Experiments

To evaluate our system, we conduct experiments addressing the three key challenges from Chapter I: (1) flexible task planning in dynamic household environments, (2) long-term memory usage, and (3) modular agent coordination. Specifically, we assess the system's ability to create grounded task plans, answer questions based on prior interactions, and route tasks to the appropriate agent.

A. Experimental Setup

This study evaluates an agent-orchestration system for symbolic task planning and follow-up questions via a knowledge base. To ensure a thorough evaluation, we consider three distinct phases:

1) **Task Planning Performance** – The symbolic task planning output is assessed independently, measuring accuracy of object assignment to their destinations.
2) **Knowledge Base Reliability** – The system's ability to reason about past actions (with and without RAG) is tested by asking about the system's current status, such as locations of previously moved items.
3) **Routing Reliability** – Measures the accuracy of the routing agent in directing queries to the appropriate agent (Task Planning, History, or itself).

To isolate the performance of the specialized agents, agent handoff is not considered in the evaluation of 1) and 2).

B. Algorithmic Framework

The frameworks and models used are shown in gray in Fig. 4. To enable efficient collaboration among agents, we use *OpenAI Swarm* [12], a lightweight framework for agent orchestration and task delegation. We evaluate the performance of *Qwen2.5-32b* [15], *Gemma2-27b* [23], and *LLaMa3.1-8b* [4], selected for their open-source availability and ability to run locally on 16GB GPU RAM. For RAG, we employ *ChromaDB* [3], a vector database optimized for fast lookups, combined with the embedding model *BGE-M3*.

https://doi.org/10.34749/3061-0710.2025.9

C. Task Scenarios

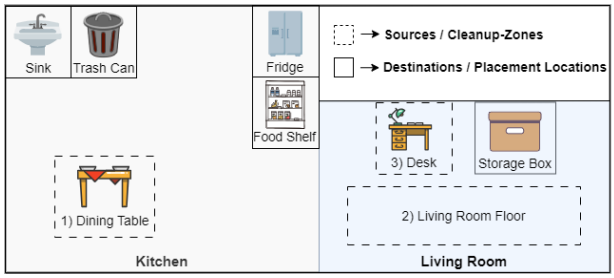


Fig. 5: The artificial household environment used in the experiment.

The experiment is conducted in an artificial household environment, where objects must be assigned to correct destinations based on high-level commands. To evaluate task planning, we define three scenarios (see Fig. 6) that share five predefined placement locations, while each uses a different cleanup zone. Fig. 5 shows a visual representation of the environment. These locations reflect common-sense knowledge typically understood by LLMs. To ensure clarity, the agent receives explicit definitions for each destination:

- **Sink** – For items that need washing.
- **Trash Can** – For disposable or inedible items.
- **Fridge** – For perishable food.
- **Food Shelf** – For non-perishable food items.
- **Storage Box** – For general storage.

Fig. 6 shows the object list extracted from a captured image of each task scenario using *LLaMa3.2-Vision* along with the user queries and the segmentation results from Grounded SAM.

After execution of all scenarios, the knowledge base agent is asked four distinct folow-up questions targeting different aspects of retrieval and reasoning:

- **Error Detection**: ”Where is the jacket that was in the living room? I thought you put it in the storage box, but I can’t find it there.”
- **Hallucination**: ”Where did you put the laptop? It’s not on the desk anymore.”
- **Food Availability**: ”I am hungry. Is there any food left from earlier?”
- **Trash Status**: ”How many objects are in the trash can?”

To better reflect real-world applications, we extend the conversation dialogue with additional question-answer pairs containing actions. Furthermore, deliberate errors are introduced into the task plans, where the agent provides the user a different location than the one forwarded to the state machine. This allows us to evaluate how well the knowledge base handles inaccuracies. Beyond evaluating the specialized agents in isolated setups, we assess how effectively the routing agent delegates tasks to the appropriate specialized agent. Specifically, we test:

- **Task Planning Queries**: The three high-level commands from the task planning scenarios (see Fig. 6) and an additional low-level request (”Can I have a banana?”)
- **Knowledge Base Queries**: The four follow-up questions from the knowledge base scenario.

D. Evaluation Methodology

The evaluation of the agent-orchestration system’s components is based on the task scenarios and follow-up questions defined in Section IV-C. Task planning performance is evaluated by testing each model on the three task scenarios, with each scenario executed five times per model. Accuracy is measured at the object level as the percentage of correctly assigned tasks. A task is deemed correct if it satisfies the following criteria:

- **Valid JSON format**
- **Correct destination assignment**
- **Stationary Object Exclusion** (ensuring no task is assigned to items that should remain in place)

The final accuracy score represents the percentage of objects for which tasks were correctly assigned, including the implicit ”no task” assignment for stationary objects (e.g., table).

The knowledge base is evaluated using four follow-up questions, each tested five times per model. Unlike the task planning agent, the knowledge base agent does not require a strict output format. It is assessed based on factual correctness, measured as the percentage of correct answers. For queries expecting multiple objects as an answer (e.g., ”Which objects are in the trash?”), accuracy is based on the percentage of correctly identified objects.

The routing agent’s ability to correctly assign tasks is evaluated by processing queries from the task planning scenarios and history-based questions, along with one additional query, five times per model. The final metric is quantified as the percentage of correctly assigned tasks. Gemma2, which does not support tool calling, is excluded from this test.

V. Results and Discussion

This section presents the experimental results for task planning, knowledge base and agent routing.

A. Task Planning

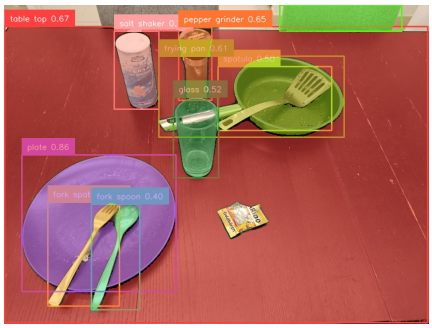
We introduce a *lenient* evaluation metric (cf. Table I), where reasonable alternative placements based on user preferences are counted as correct. The strictly correct placements, following the intended plan as prompted to the LLM, are presented under the *strict* metric in Table I.

Table I shows that *Qwen* consistently outperforms the other models in nearly all scenarios. *LLaMA* performs notably worse in the living room scenario, with the lowest strict accuracy (40.0%). *Gemma2* falls between the two, showing higher accuracy than *LLaMA* but lower than *Qwen*.

B. Knowledge Base

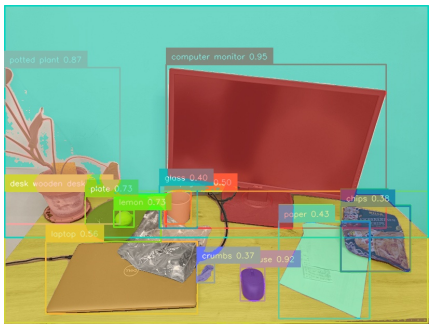
The integration of RAG notably enhances the accuracy of the knowledge base’s responses, even in medium-term interactions consisting of 21 question-answer pairs with approximately 4000 tokens. *Qwen* achieves the highest validity (91.3%) with RAG (cf. Table II), highlighting the potential of retrieval-augmented approaches for maintaining consistency over longer interactions.

https://doi.org/10.34749/3061-0710.2025.9



(a) **Scenario 1:** Dining Table Cleanup **Object list from VLM:** *Plate, Fork, Spoon, Salt shaker, Glass, Frying pan, Spatula, Chair, Table top, Pepper grinder.* **Command:** *I just finished dinner, please clear the dining table.*

(b) **Scenario 2:** Living Room Cleanup **Object list from VLM:** *A table, A couch, A brush, Scissors, Pen, Book, Salt packet, Jacket, Markers.* **Command:** *Please hand me the brush and tidy up the rest of the living room.*



(c) **Scenario 3:** Desk Organization **Object list from VLM:** *Desk, Computer Monitor, Laptop, Mouse, Plate, Crumbs, Lemon, Cup, Glass of water, Bag of chips, Piece of paper, Potted plant, Cord, Wooden desk, White wall.* **Command:** *Please clear my desk, leaving only the essentials for work.*

Fig. 6: The three scenarios used for task planning. For each scenario we have extracted an object list using the Vision-Language Model *LLaMa3.2-Vision*. This list is used as input for Grounded SAM [17] to perform segmentation.

Model	Dining Table		Living Room		Desk Organization		Total Accuracy (%)	
	Strict (%)	Lenient (%)	Strict (%)	Lenient (%)	Strict (%)	Lenient (%)	Strict (%)	Lenient (%)
LLaMa3.1-8B	**68.0**	78.0	40.0	40.0	61.3	65.3	56.4	61.1
Gemma2-27B	58.0	68.0	68.9	68.9	68.0	69.3	65.0	68.7
Qwen2.5-32B	64.0	**80.0**	**88.9**	**88.9**	**78.7**	**84.0**	**77.2**	**84.3**

TABLE I: Task Planning Accuracy Across Different LLMs. **Strict (%)**: Percentage of objects correctly placed according to the intended plan. **Lenient (%)**: Percentage of objects placed differently than expected, but with reasonable alternative placements based on user preferences.

Method	Model	Response Validity (%)				Total Validity (%)
		Err. Detection	Hallucination	Food Avail.	Trash Status	
Without RAG (Ablation Study)	LLaMa3.1-8B	20.0	80.0	70.0	65.0	58.8
	Gemma2-27B	0.0	80.0	10.0	60.0	37.5
	Qwen2.5-32B	0.0	80.0	60.0	**75.0**	53.75
With RAG	LLaMa3.1-8B	40.0	**100.0**	**90.0**	55.0	71.25
	Gemma2-27B	80.0	**100.0**	40.0	60.0	70.0
	Qwen2.5-32B	**100.0**	**100.0**	**90.0**	**75.0**	**91.3**

TABLE II: Knowledge Base Response Accuracy Across Different LLMs. **Used Embedding Model for RAG**: *BGE-M3*. **No. of question-answer pairs retrieved by RAG**: *5*

C. Agent Routing

In task delegation, *LLaMA* exhibits the highest routing accuracy (92.5%), despite its weaker reasoning abilities (cf. Table III). Its structured approach to tool-calling ensures stable performance. In contrast, *Qwen*, while superior in contextual understanding, occasionally produces incorrect structured outputs, leading to execution failures.

D. Summary

Our findings highlight the potential of lightweight, open-source LLMs for memory-augmented long-horizon task planning. A combination of *LLaMA* (routing) and *Qwen* (specialized agents) achieves the best balance between structured execution and high-level reasoning.

Evaluating task execution remains challenging due to subjective human preferences, emphasizing the need for user studies. Furthermore, integrating Vision-Language Models (VLMs) into the agent orchestrator – rather than only using them for object lists – could enhance robustness. Embedding contextual information into the latent space reduces command dependency and improves autonomy.

RAG improves factual consistency in knowledge retrieval but struggles with repeated object interactions and long histories, making full-history queries impractical. Scene graphs, as proposed by Liu et al. [10], present a promising alternative for efficient and robust knowledge integration.

https://doi.org/10.34749/3061-0710.2025.9

Model	Task Planning Queries (%)	Knowledge Base Queries (%)	Total Success Rate (%)
LLaMa3.1-8B	85.0	**100.0**	**92.5**
Qwen2.5-32B	**95.0**	85.0	90.0

TABLE III: Routing Success Rate Across Different LLMs

While task delegation via the routing agent was mostly successful, certain models occasionally produced invalid structured outputs, leading to execution failures. To increase robustness, future work should explore schema validation and adaptive retry mechanisms that can automatically mitigate such issues.

In summary, open-source LLMs prove viable for long-horizon task planning. However, addressing key challenges – refining evaluation metrics, improving long-term robustness, and integrating multimodal perception – remains essential for achieving reliable household robotics.

VI. Conclusion

This work presents a prototype of an agent-orchestration system for household robots, utilizing local, lightweight open-source LLMs to translate high-level user commands into structured task plans for tidy-up scenarios. Memory-augmented task planning enables follow-up queries about past actions, improving user interaction and assisting in locating misplaced objects. Our evaluation shows strong task planning, routing, and knowledge retrieval. with Qwen2.5 excelling in reasoning-heavy tasks and LLaMA3.1 providing a more efficient routing solution. However, RAG-based retrieval for general tasks remains a challenge, particularly for implicit queries where relevant information is not always found. Addressing these limitations is key to improving long-term reasoning and knowledge access.

Future work will focus on robust storage solutions, improved knowledge representations, broader user studies with structured datasets for evaluating and benchmarking existing approaches. Enhancing communication and tool usage in agent-orchestration will be crucial for greater adaptability and autonomy in household robotics.

ACKNOWLEDGMENT

This research is supported by the EU program EC Horizon 2020 for Research and Innovation under grant agreement No. 101017089, project TraceBot, and the Austrian Science Fund (FWF), under project No. I 6114, iChores.

References

[1] A. Anwar, J. Welsh, J. Biswas, S. Pouya, and Y. Chang, "Remembr: Building and reasoning over long-horizon spatio-temporal memory for robot navigation," *arXiv preprint arXiv:2409.13682*, 2024.

[2] A. Brohan, *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," in *in Proceedings of the Conference on Robot Learning (CoRL)*, 2023, pp. 287–318.

[3] Chroma, "Chromadb," open-source vector database for AI applications. [Online]. Available: https://www.trychroma.com/

[4] A. Dubey, *et al.*, "The llama 3 herd of models," *arXiv preprint arXiv:2407.21783*, 2024.

[5] D. Guo, *et al.*, "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," *arXiv preprint arXiv:2501.12948*, 2025.

[6] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch, "Language models as zero-shot planners: Extracting actionable knowledge for embodied agents," in *in Proceedings of the International Conference on Machine Learning (ICML)*, 2022, pp. 9118–9147.

[7] W. Huang, *et al.*, "Inner monologue: Embodied reasoning through planning with language models," *arXiv preprint arXiv:2207.05608*, 2022.

[8] P. Lewis, *et al.*, "Retrieval-augmented generation for knowledge-intensive nlp tasks," *Advances in neural information processing systems*, vol. 33, pp. 9459–9474, 2020.

[9] J. Liang, *et al.*, "Code as policies: Language model programs for embodied control," in *in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 9493–9500.

[10] Y. Liu, L. Palmieri, S. Koch, I. Georgievski, and M. Aiello, "Delta: Decomposed efficient long-term robot task planning using large language models," *arXiv preprint arXiv:2404.03275*, 2024.

[11] D. M. McDermott, "The 1998 AI planning systems competition," *AI Magazine*, vol. 21, no. 2, p. 35, 2000.

[12] OpenAI, "Swarm: Educational framework for multi-agent systems." [Online]. Available: https://github.com/openai/swarm

[13] OpenAI, *et al.*, "GPT-4 technical report," 2024. [Online]. Available: http://arxiv.org/abs/2303.08774

[14] V. Pallagani, *et al.*, "On the prospects of incorporating large language models (llms) in automated planning and scheduling (aps)," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 34, 2024, pp. 432–444.

[15] Qwen, *et al.*, "Qwen2.5 technical report," 2025. [Online]. Available: http://arxiv.org/abs/2412.15115

[16] K. Rana, *et al.*, "Sayplan: Grounding large language models using 3d scene graphs for scalable robot task planning," *arXiv preprint arXiv:2307.06135*, 2023.

[17] T. Ren, *et al.*, "Grounded sam: Assembling open-world models for diverse visual tasks," *arXiv preprint arXiv:2401.14159*, 2024.

[18] J. Ruan, *et al.*, "Tptu: Task planning and tool usage of large language model-based ai agents," in *NeurIPS 2023 Foundation Models for Decision Making Workshop*, 2023.

[19] T. Schick, *et al.*, "Toolformer: Language models can teach themselves to use tools," *Advances in Neural Information Processing Systems*, vol. 36, pp. 68 539–68 551, 2023.

[20] T. Silver, *et al.*, "Generalized planning in pddl domains with pretrained large language models," in *in Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, 2024, pp. 20 256–20 264.

[21] I. Singh, *et al.*, "Progprompt: Generating situated robot task plans using large language models," in *in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 11 523–11 530.

[22] M. Sundermeyer, A. Mousavian, R. Triebel, and D. Fox, "Contact-graspnet: Efficient 6-dof grasp generation in cluttered scenes," in *in Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 13 438–13 444.

[23] G. Team, *et al.*, "Gemma 2: Improving open language models at a practical size," *arXiv preprint arXiv:2408.00118*, 2024.

[24] S. Tellex, N. Gopalan, H. Kress-Gazit, and C. Matuszek, "Robots that use language," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, no. 1, pp. 25–55, 2020.

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

[26] J. Wei, *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in neural information processing systems*, vol. 35, pp. 24 824–24 837, 2022.

[27] S. Yao, *et al.*, "React: Synergizing reasoning and acting in language models," in *in Proceedings of the International Conference on Learning Representations (ICLR)*, 2023.

[28] D. Zhou, *et al.*, "Least-to-most prompting enables complex reasoning in large language models," *arXiv preprint arXiv:2205.10625*, 2022.