RAP: Retrieval-Augmented Planning with Contextual Memory for Multimodal LLM Agents

Tomoyuki Kagaya*
Panasonic Connect Co., Ltd.

Thong Jing Yuan*
Panasonic R&D Center Singapore

Yuxuan Lou*
National University of Singapore

Jayashree Karlekar
Panasonic R&D Center Singapore

Sugiri Pranata
Panasonic R&D Center Singapore

Akira Kinose
Panasonic Connect Co., Ltd.

Koki Oguri
Panasonic Connect Co., Ltd.

Felix Wick
Panasonic R&D Center Germany

Yang You
National University of Singapore

Abstract

With recent advancements, Large Language Models (LLMs) can now be deployed as agents for increasingly complex decision-making applications in areas like robotics, gaming, and API integration. However, reflecting past experiences in current decision-making processes, an innate human behavior, continues to pose significant challenges. Addressing this, we propose Retrieval-Augmented Planning (RAP) framework, designed to dynamically leverage past experiences corresponding to the current situation and context, thereby enhancing agents' planning capabilities. RAP distinguishes itself by being versatile: it excels in both text-only and multimodal environments, making it suitable for a wide range of tasks. Empirical evaluations demonstrate RAP's effectiveness, where it achieves SOTA performance in textual scenarios and notably enhances multimodal LLM agents' performance for embodied tasks. These results highlight RAP's potential in advancing the functionality and applicability of LLM agents in complex, real-world applications.

1 Introduction

With impressive inferential abilities of Large Language Models (LLMs) [1, 2], LLMs can be employed as agents in various domains like decision-making and robotic control. Previous works like ReAct [3] have shown that LLMs can produce accurate actions by alternating between actions and reasoning.

Moreover, Retrieval-Augmented Generation (RAG) has emerged as a prevalent technique in enhancing LLMs' generation capabilities. This approach amalgamates external knowledge into the generation process, thus enriching the context of generated content. However, challenges remain in using external memory to enhance LLM planning, particularly in complex environments. Existing works like Reflexion [4] that analyzes failure cases, and ExpeL [5] that extracts insights for language agents,

*Equal contribution.

Preprint. Under review.

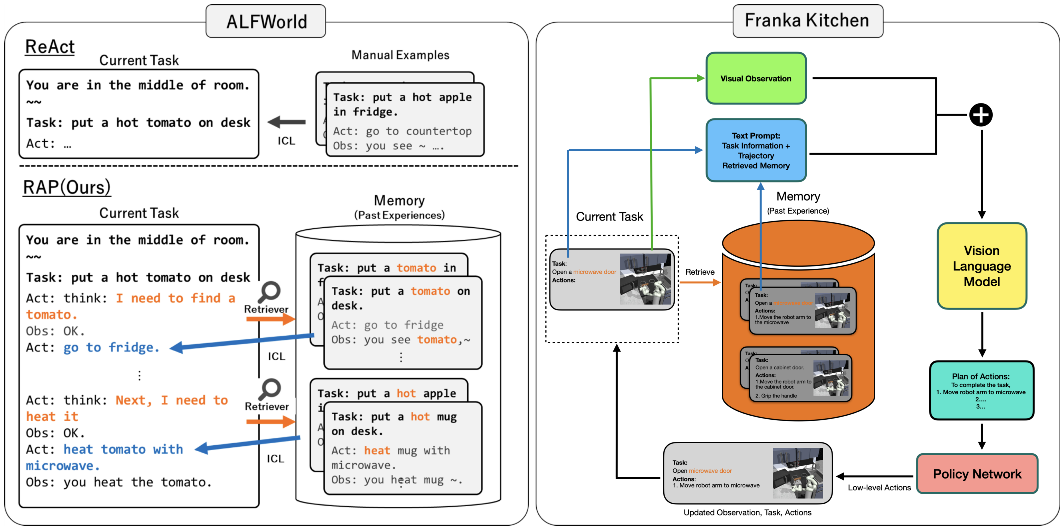

Figure 1: Overview of RAP. Our framework stores past experiences and retrieves them based on the current situation. **Left:** The evaluation process on ALFWorld. ICL stands for in-context learning. **Right:** The evaluation process on Franka Kitchen.

fall short in utilizing comprehensive past information in complex environments. This highlights a critical gap: the lack of a comprehensive framework for leveraging past experiences in LLM agent planning, limiting their adaptability in complex, real-world scenarios.

In this paper, we introduce a novel framework, Retrieval-Augmented Planning (RAP), which embodies a pivotal human ability – to leverage generalized past experiences for current tasks – for LLM agents. Our approach involves storing past experiences in memory, retrieving relevant ones based on the present context with may include multimodal information, and generating subsequent actions via in-context learning, thereby enhancing the decision-making capacity of language agents.

Central to this framework is the LLMs' ability to perform analogy-making from various abstracted patterns [6]. Leveraging this capability, our memory stores both context and action-observation trajectories for each experience. The approach effectively facilitates deriving correct actions from memory examples within task constraints. Furthermore, by storing multimodal information in memory and considering it when retrieving past experiences, our approach flexibly utilizes multimodal information with LLMs and Vision-Language Models (VLMs) separately for language agents.

Consequently, our approach proves to be effective for memory utilization by language agents in both decision-making and robotics tasks. Specifically, RAP achieves 33.6%, 13.0%, 18.2%, and 12.7% gain over ReAct on the ALFWorld, Webshop, Franka Kitchen, and Meta World benchmarks.

To summarize, our contributions are as follows:

- We propose RAP, a novel framework that enhances agents' planning capacity, that enriches decision-making via intelligent retrieval of past experiences based on the current context.
- RAP is versatile for both textual environments and multimodal embodied tasks, marking it as a pioneering effort in employing memory retrieval techniques for multimodal agents, a first in this domain to our knowledge.
- RAP shows significant gains compared to prior SOTA methods across various environments.

2 Related Work

2.1 Language Models and Vision-Language Models as Foundations

LLMs such as GPT [7] and LLaMA [8] have excelled in generating coherent textdue to their vast linguistic knowledge and reasoning abilities. Extending beyond LLMs, Vision-Language

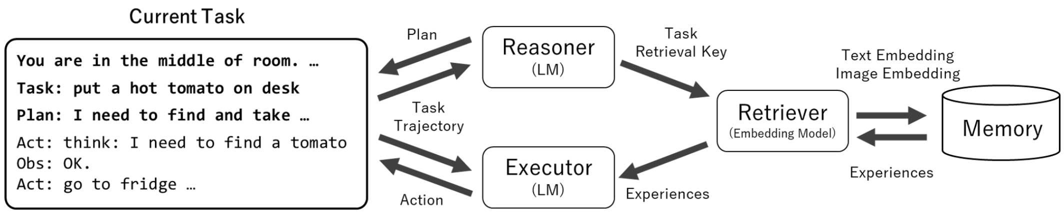

Figure 2: RAP Core Components

Models (VLMs)[9] such as LLaVA[10] and CogVLM[11], have attained a profound understanding of multimodal inputs, exemplified by LLaVA's training on image-caption pairs [12]. Our work utilizes these foundations to build agents for textual and embodied environments: text-based agents employing LLMs, and embodied agents integrating VLMs for visual perception and action planning. We focus on enhancing these agents' planning capabilities through memory retrieval techniques, enabling them to selectively access and utilize relevant memory for improved sequential decision-making.

2.2 Language Models as AI Agents

Recent works have leveraged LLM's anthropomorphic capabilities when building autonomous agents. These agents can be depicted into having 4 key aspects: Profile (agent characteristics), Memory (past information), Planning (future strategies) and Action (execution policies) [13]. A notable example is Chain-of-Thought (CoT) [14], where agents are encouraged to mirror human cognitive mechanisms by incorporating reasoning into intermediate steps for complex problem-solving tasks. With a dynamic reasoning process, ReAct [3] interleaves generated actions and environmental states, improving the reasoning ability through action-state synergy. Our work seeks to enhance the ReAct framework by allowing agents to identify specific objects within observations and additionally retrieve relevant aspects of past experiences based on the current context. By doing so, our agent can adaptably receive different experiences at different points in time that are most similar to the situation at hand.

2.3 Retrieval-Augmented Generation with Memory

Among works [15, 16, 17] that seek to derive better answers from LLMs via memory, RAG [18] is a notable method that combines retrieval-based mechanisms with generative models. Responses from memory are selected and passed into LLMs as additional context to deliver outputs that are creative and contextually-grounded. Building on RAG, Reflexion [4] requires LLMs to self-reflect on unsuccessful tasks for self-improvement in solving tasks over time. ADaPT [19] further decomposes into sub-tasks and re-executes where necessary. Yet, these works only reflect on trajectories within the task. Hence, these insights are often restricted to each task. Building on Reflexion, ExpeL [5] passes all generated experiences into the LLM to reflect in a text-based manner. In contrast, our work adopts a different approach by implicitly drawing from a diverse range of experiences from memory without explicitly requiring an additional step of re-tasking the LLM to extract insights. With this approach, our agent can not only efficiently generalize experiences from other successful tasks to solve the current task, but also be flexible enough to extract relevant components from experiences, enhancing the agent's ability to expand its memory from textual to multimodal contexts.

3 RAP: Retrieval-Augmented Planning

We developed Retrieval-Augmented Planning (RAP), a framework that leverages past experiences to facilitate decision-making according to the current context. Fig. 2 provides an overview of the framework, which consists of four core components: Memory, Reasoner, Retriever, and Executor. The specific details of each module will be discussed in sections 3.2 to 3.5.

3.1 Preliminaries

In this work, we consider an agent operating in a particular environment and assigned with completing some task T. The agent forms an overall plan p, then interacts with the environment over a finite horizon of H timesteps. The overall plan is the first "think" statement, that details the agent's strategy to solve the overarching task. At each timestep $t \in \{1, 2, \ldots H\}$, the agent forms an action plan p'_t, selects an action α_t from the action space A and receives an observation o_t from the observation space O. Here, action plans are all subsequent "think" statements that agent creates when interacting with environment. The trajectory $\tau = \{\vec{p'_t}, \vec{\alpha_t}, \vec{o_t}\}$ of the agent up to time t consists of the sequences of plans $\vec{p'_t} = (p'_1, ...p'_t)$, actions $\vec{\alpha_t} = (\alpha_1, ...\alpha_t)$ and observations $\vec{o_t} = (o_1, ...o_t)$. Hence, *action plans* are reasoning steps created when interacting with environment, whereas the *overall plan* is generated only at the first execution step. Also, the Reasoner generates *action plans* for agent to form its own reasoning in environment, while the Executor generates *actions* to interact with environment. Using the example on Page 22 to illustrate,

- **Overall plan**: "To solve the task, I need to find and take a cloth, then clean it with sinkbasin, then put it in cabinet."
- **Action plan**: Act 1 "think: First I need to find a cloth", Act 7 "think: Now I take a cloth (1). Next, I need to go to sinkbasin (1) and clean it"
- **Actions**: Act 2 "go to cabinet 1", Act 3 "open cabinet 1"

3.2 Memory

To enable retrieval-augmented planning, memory databases, that contain logs of prior successful task executions, are first constructed. For each log L_i completing a task T_i in H_i steps, we record the task information T_i, overall plan p_i , and agent's trajectory τ_{L_i} including plans, actions, and observations.

$$L_i = \{T_i, p_i, \tau_{L_i}\} \tag{1}$$

$$\tau_{L_i} = \{\vec{p'_{L_i}}, \vec{\alpha_{L_i}}, \vec{o_{L_i}}\} \tag{2}$$

For textual environments, the observations are textual descriptions of the world state. For multimodal environments, the observations are images from a fixed viewpoint camera after each agent action.

The logs are collected by having agents attempt the tasks and saving streams of successful episodes. The episodic logs capture the steps needed to complete the tasks. Storing these examples allows the agents to leverage prior experience when planning for new instances of the tasks.

During interactions with the environment, the agents can selectively retrieve relevant memory samples to make more informed action decisions. For textual tasks, the logs provide crucial context. For multimodal tasks, prior visual observations reveal outcomes of actions. By retrieving prototypical executions, the agents can plan smarter policies while avoiding past failures. The memory augmentation thus equips the models with vital environmental knowledge for sequential decision-making.

3.3 Reasoner

The Reasoner generates overall plans, action plans, and retrieval keys based on the agent's current situation and action trajectory, using LLMs. Initially, the Reasoner produces the overall plan from the task information. Based on the task and the overall plan, the Reasoner generates an action plan. Also, in accordance with ReAct, an action or action plan is dynamically generated by LLMs, considering the current task status. If an action plan is generated, a retrieval key is created based on the generated action plan. For instance, in ALFWorld [20], if an action plan like "I need to find the watch" is generated, the retrieval key would be "search watch". Hence, the Reasoner enables agents to take into consideration the current situation and context.

3.4 Retriever

The Retriever is designed to extract the most relevant memory logs to guide the agent's subsequent actions to complete the current task. This process is shown in Fig. 3 (Left). Here, the similarity score, comparing the current state S_0 with log L_i, is calculated as a weighted average of the task similarity, overall plan alignment, and retrieval key congruence.

Let the current agent task be T_0, the overall plan be p_0, and the retrieval key generated by the Reasoner based on current action plan p' be k_0. The similarity score between current state S_0 with log L_i is calculated as a weighted average of the similarity score for task, overall plan, and retrieval key.

$$Score(S_0, L_i) = w_t \cdot sim(T_0, T_{Li}) + w_p \cdot sim(p_0, p_{L_i}) + w_k \cdot sim(k_0, \tau_{L_i}) \tag{3}$$

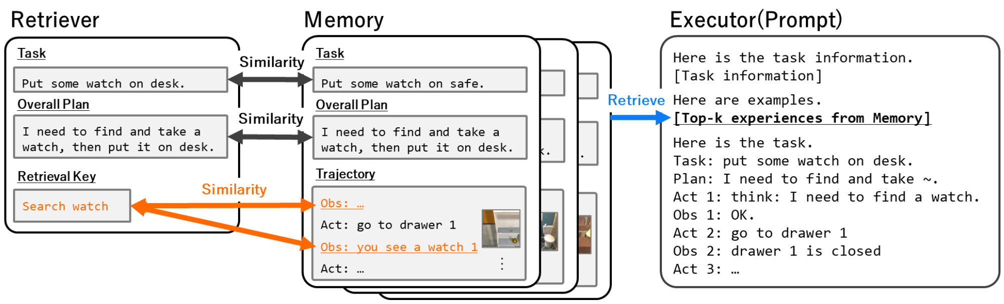

Figure 3: Memory-Retrieval in RAP. **Left to Middle:** The Retriever, calculating similarities with Memory, dynamically switches between action or observation based on the Retrieval Key. **Right:** Executor receives related experiences from memory and utilizes them in the prompt.

Each component's similarity score is determined using cosine similarity of their feature representations. For text data, the representations are derived using sentence-transformers [21]. For images, the representations are generated with a CLIP-based Vision Transformer.

The similarity score between the retrieval key and the log trajectory is adaptive based on environment type and retrieval key type. In multimodal environments, the retrieval key corresponds to agents' current visual observation. Thus, the retrieval-key similarity score is the score between current and logged visual trajectory observations as in equation (4).

$$sim(k_0, \tau_{L_i}) = max(cos\_sim(k_0^{visual}, o_j)), \text{ for } o_j \in \vec{o_{L_i}} \tag{4}$$

In textual environments, the retrieval-key similarity score is adaptive based on key type. In scenarios involving the retrieval key for searching or locating objects, the similarity score is calculated between the retrieval key and the logged textual trajectory observations, as in equation (5).

$$sim(k_0, \tau_{L_i}) = max(cos\_sim(k_0^{text}, o_j)), \text{ for } o_j \in \vec{o_{L_i}} \tag{5}$$

When the retrieval-key influences the action plan, it is an action similarity score, as in equation (6).

$$sim(k_0, \tau_{L_i}) = max(cos\_sim(k_0^{text}, \alpha_j)), \text{ for } \alpha_j \in \vec{\alpha_{L_i}} \tag{6}$$

Furthermore, component weights are adaptively calibrated based on the environment. In environments with a constrained task space, task similarity is assigned a higher weight. For example, in Franka Kitchen environment which has only 5 tasks, we only retrieve logs with same task type.

For each retrieved experience, only a window of trajectory centered around the most similar action is passed to the agent. This allows agents to focus on actions most similar to current task, rather than full trajectories that may instead create additional noise to the agent.

In summary, our meticulously-crafted retrieval method efficiently identifies the most pertinent logs by calculating a weighted similarity score that takes into account various aspects including task information, overall planning, and retrieval key. This process ensures that the most relevant and contextually appropriate logs are selected from a vast repository of memory logs. Once these logs are retrieved, they serve as an invaluable resource for the large language model serving as Executor.

3.5 Executor

The Executor receives past experiences from the Retriever and generates the next action by utilizing these experiences through in-context learning. This process is illustrated in Fig. 3 (Right). By presenting the past experience aligned with the current context as a prompt, it enables accurate decision-making for the next action, mirroring the process humans leverage past experiences for future actions. Additionally, the length of the current task trajectory is used in the same way as past experiences, utilizing only a constant number of new trajectories. This encourages effective analogy-making from experiences through in-context learning in LLMs.

Algorithm 1 Retrieval-Augmented Planning

Initialize: Trajectory τ, Reasoner$_{LLM}$, Retriever, Executor$_{LLM}$, Memory
Input: Task T
p_o = Reasoner(T) // Generate overall plan
e = Retriever(Memory, T) // Retrieve experiences
while reward is not Success and $t <$ max steps **do**
    // Generate action or action plan
    a_t = Executor(T, p_o, e, τ)
    if action plan **then**
        k = Reasoner(a_t) // Generate retrieval key
        e = Retriever(Memory, T, k) // Retrieve experiences
    else
        reward, o_t = Env(a_t) // Input action to environment
    end if
    $\tau \leftarrow a_t, o_t$ // Update trajectory
    $t = t + 1$ // Increment
end while

4 Experiments

To validate the effectiveness of our framework in various environments, we performed evaluations on four benchmarks. These include the text-based multi-step tasks in ALFWorld [20] and Webshop [22], and robotics tasks, FrankaKitchen [23] and Meta-World [24], which are multimodal environments. A more detailed analysis of ablation studies performed can be found in Appendix C.

4.1 Textual Environments

4.1.1 ALFWorld

ALFWorld [20] is a synthetic text-based game that challenges an agent to solve multi-step tasks in a variety of interactive environments based on TextWorld [25]. Following ReAct, we evaluated an agent in 134 unseen games, including six types of tasks: Pick, Clean, Heat, Cool, Look, and Pick2. In this environment, agents are required to accomplish complex tasks using text-based actions in a simulated household providing textual and image feedback. Following previous works [3, 4, 19], RAP runs recursively until it reaches a depth (trial) of 3.

In our evaluation, due to the lack of prior experiences, the initial trial in RAP was conducted using ReAct. We built our memory from the tasks that succeeded. For subsequent trials, we utilized the memory from the previous trial. In RAP$_{train}$, the memory was constructed using 1000 tasks from the provided training set, and run recursively with memory from successful tasks both from the training set and previous trials. Additionally, we use task information including the task category for the Retriever. During action-based similarity calculation, it extracts four experiences and ten actions from both before and after the most similar action. Meanwhile, during observation-based similarity calculation, it uses eight experiences and five actions from both before and after the most similar action.

We conducted evaluations using three models: GPT-3.5 [26], GPT-4 [7], and Llama2-13b [8]. In Table 1, our experiments with GPT-3.5 show that RAP(85.8%) and RAP$_{train}$(91.0%) achieve a significantly higher success rate compared to previous works such as ReAct(52.2%), Reflexion(74.6%), and ADaPT(71.6%). Further, RAP is effective for both high-performance models like GPT-4, and locally-operated models like Llama2-13b, thus illustrating the efficacy of RAP across various LLMs.

4.1.2 WebShop

WebShop [22] is a web application that simulates online shopping, where agents are required to select products for purchase based on a given user instruction. WebShop contains a total of 1.18M real-world products featured on Amazon, and comprises a wide variety of structured and unstructured texts. Following Reflexion [4] and ADaPT [19], we evaluated an agent across 100 instructions. For each instruction, agents are required to reason and select a desired product that is most aligned to

Table 1: ALFWorld task-specific success rate(%).

Method(d_{max}=3)	Model	Pick	Clean	Heat	Cool	Look	Pick2	All
Act	GPT-3.5	66.7	51.6	73.9	61.9	38.9	17.6	53.7
ReAct	GPT-3.5	50.0	41.9	73.9	66.7	55.6	23.5	52.2
Reflexion	GPT-3.5	75.0	77.4	65.2	76.2	83.3	70.6	74.6
ADaPT*	GPT-3.5	87.5	80.6	60.8	76.2	61.1	52.9	71.6
RAP(Ours)	GPT-3.5	**95.8**	87.1	78.3	**90.5**	88.9	70.6	85.8
RAP$_{train}$(Ours)	GPT-3.5	**95.8**	**100.0**	**82.6**	85.7	**100.0**	**76.5**	**91.0**
ReAct	GPT-4	83.3	71.0	95.7	81.0	**100.0**	**94.1**	85.8
RAP(Ours)	GPT-4	**95.8**	**90.3**	**100.0**	**95.2**	**100.0**	88.2	**94.8**
ReAct	Llama2-13b	29.2	41.9	34.8	52.4	38.9	**17.6**	36.6
RAP(Ours)	Llama2-13b	**62.5**	**61.3**	**56.5**	**61.9**	**44.4**	**17.6**	**53.0**

* We use the performance reported by [19]

the given instruction based on observations returned by the web application, and perform additional precise interactions with the portal to navigate through the web application such as searching or clicking buttons. Such interactions are performed in a text-based manner where the agent issues a textual command into the web application. Following previous studies, we allow the agent to run recursively until it reaches a depth (trial) of 3.

During our evaluation, we ran the initial trial with a ReAct agent and formulated the memory database based on successful tasks. The memory database would be further expanded for subsequent trials based on successful tasks in the preceding trials. Here, successful tasks are counted as those with a reward of 1. In addition, during retrieval of actions in memory, the Retriever extracts three experiences and five actions from before and after the most similar action.

Moreover, unlike other environments where objects are generalizable across different tasks, WebShop has an additional unique feature where actions in each task are highly dependent on the scenario outlined in that task. As such, apart from the correlation between the current reasoning and trajectories in memory, our agent also considers the relationship between the action of each task in memory that is most similar to the current action and its corresponding scenario for that task. This builds on the concept of "A is to B as C is to D", where the generated action depends not only on similar trajectories in memory, but also how these trajectories relate back to their scenario, and how the current trajectory should be related to the current scenario. By incorporating intra-task relationships, this allows the agent to better reason how the actions in memory are correlated with their own scenarios, and thereafter generate an action that is also aligned to the current scenario at hand.

We performed evaluations using two different models: GPT-3.5 [26] and Llama2-13b [8]. In Table 2, experiments with GPT-3.5 demonstrate that our method (48.0%) achieves a higher success rate compared to previous studies such as ReAct (35.0%), Reflexion (35.0%), and ADaPT (43.0%). Furthermore, our method is able to achieve a higher overall reward score (76.1%) as compared to ReAct (61.8%), Reflexion (61.8%) and ADaPT (64.0%).

Table 2: WebShop Score (%) and Success Rate(%).

Method(d_{max}=3)	Model	Score	Success Rate
ReAct	GPT-3.5	61.8	35.0
Reflexion	GPT-3.5	61.8	35.0
ADaPT	GPT-3.5	64.0	43.0
RAP(Ours)	GPT-3.5	**76.1**	**48.0**
ReAct	Llama2-13b	64.6	31.0
RAP(Ours)	Llama2-13b	**71.1**	**36.0**

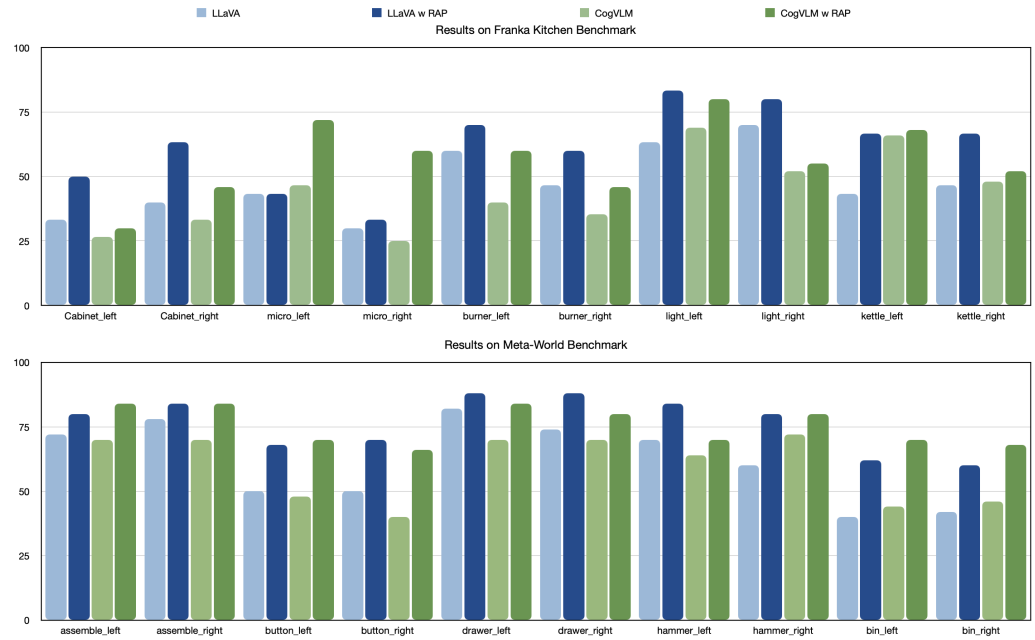

Figure 4: Evaluation on Franka-Kitchen and Meta-World Benchmark. We evaluate with LLaVA and CogVLM both with and without our proposed RAP method. The results demonstrate that our method notably enhances the performance of multimodal LLM agents in executing embodied tasks.

4.2 Multimodal Environments

We evaluated our proposed technique in embodied multimodal agents on two benchmark environments: Franka Kitchen and Meta-World. These simulations offer a diverse set of household and robotic manipulation tasks requiring visual perception and physical interaction.

We constructed embodied agents using 2 VLM foundations - LLaVA and CogView. For each VLM, we compared task performance of the base model to a RAP-enhanced agent utilizing our memory retrieval system.

The Franka Kitchen benchmark consists of compound tasks like arranging objects and preparing meals, while Meta-World provides a suite of 50 distinct robotic skills focused on fine manipulation. Here, the agent is required to plan actions based on visual observations in an interactive 3D environment. We report quantitative results on success rates with and without RAP. Our method allows VLM Agents to selectively reference prior successful executions during planning. This provides vital visual context and demonstrates the benefits of memory-augmented reasoning for embodied agents.

To map the high-level plans of agents to executable environment actions, we train a policy network on 25 demonstrations for each task. We evaluate on 5 subtasks with 2 different camera views. For each (task, view) combination, we run 50 trials with different random seeds and report success rates.

Table 3 and Figure 4 shows that RAP can significantly enhance embodied multimodal agents planning on both benchmarks. The results offer insights into how memory can aid these models for sequential decision making and embodied tasks requiring interactive visual perception.

Table 3: Average success rates on Franka Kitchen and Meta World of VLM Agents(%)

Method	Franka Kitchen	Meta World
LLaVA	43.4	65.4
LLaVA with RAP (Ours)	**61.6**	**79.2**
CogVLM	44.2	61.5
CogVLM with RAP (Ours)	**56.9**	**68.8**

5 Conclusion and Limitations

We propose Retrieval-Augmented Planning (RAP), a framework that leverages past experiences from multimodal sources to guide language agents' actions. RAP demonstrated superior performance across various LLMs and agent/robotics benchmarks, enhancing decision-making capabilities by mimicking human-like experience utilization. While promising, our study reveals important future directions. These include addressing the organization of accumulating memory in lifelong learning scenarios and expanding beyond text and image inputs to include video and audio. These challenges are crucial for real-world applications and more adaptive AI systems. In conclusion, RAP advances experience-driven language agents while highlighting key areas for future research to fully realize their potential in complex, dynamic environments.

Acknowledgments and Disclosure of Funding

Use unnumbered first level headings for the acknowledgments. All acknowledgments go at the end of the paper before the list of references. Moreover, you are required to declare funding (financial activities supporting the submitted work) and competing interests (related financial activities outside the submitted work). More information about this disclosure can be found at: `https://neurips.cc/Conferences/2024/PaperInformation/FundingDisclosure`.

Do **not** include this section in the anonymized submission, only in the final paper. You can use the `ack` environment provided in the style file to automatically hide this section in the anonymized submission.

References

[1] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Ji-Rong Wen. A Survey on Large Language Model based Autonomous Agents. In *arXiv*, 2023.

[2] Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, Rui Zheng, Xiaoran Fan, Xiao Wang, Limao Xiong, Yuhao Zhou, Weiran Wang, Changhao Jiang, Yicheng Zou, Xiangyang Liu, Zhangyue Yin, Shihan Dou, Rongxiang Weng, Wensen Cheng, Qi Zhang, Wenjuan Qin, Yongyan Zheng, Xipeng Qiu, Xuanjing Huang, and Tao Gui. The rise and potential of large language model based agents: A survey. In *arXiv*, 2023.

[3] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

[4] Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Neural Information Processing Systems (NeurIPS)*, 2023.

[5] Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM Agents Are Experiential Learners. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2023.

[6] Suvir Mirchandani, Fei Xia, Pete Florence, Brian Ichter, Danny Driess, Montserrat Gonzalez Arenas, Kanishka Rao, Dorsa Sadigh, and Andy Zeng. Large language models as general pattern machines. In *Conference on Robot Learning (CoRL)*, 2023.

[7] OpenAI. Gpt-4 technical report. In *arXiv*, 2023.

[8] Meta GenAI. Llama 2: Open foundation and fine-tuned chat models. In *arXiv*, 2023.

[9] Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. A survey on multimodal large language models. In *arXiv*, 2023.

[10] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *ArXiv*, volume abs/2304.08485, 2023.

[11] Weihan Wang, Qingsong Lv, Wenmeng Yu, Wenyi Hong, Ji Qi, Yan Wang, Junhui Ji, Zhuoyi Yang, Lei Zhao, Xixuan Song, Jiazheng Xu, Bin Xu, Juanzi Li, Yuxiao Dong, Ming Ding, and Jie Tang. CogVLM: Visual Expert for Pretrained Language Models. In *ArXiv*, volume abs/2311.03079, 2023.

[12] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[13] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Ji-Rong Wen. A survey on large language model based autonomous agents. In *arXiv*, 2023.

[14] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc., 2022.

[15] Aman Madaan, Niket Tandon, Peter Clark, and Yiming Yang. Memory-assisted prompt editing to improve gpt-3 after deployment. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2022.

[16] Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. What makes good in-context examples for gpt-3? In *arXiv*, 2021.

[17] Hongjin Su, Jungo Kasai, Chen Henry Wu, Weijia Shi, Tianlu Wang, Jiayi Xin, Rui Zhang, Mari Ostendorf, Luke Zettlemoyer, Noah A. Smith, and Tao Yu. Selective annotation makes language models better few-shot learners. In *arXiv*, 2022.

[18] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc.

[19] Archiki Prasad, Alexander Koller, Mareike Hartmann, Peter Clark, Ashish Sabharwal, Mohit Bansal, and Tushar Khot. Adapt: As-needed decomposition and planning with language models. In *arXiv*, 2023.

[20] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations (ICLR)*, 2021.

[21] Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *arXiv*, 2019.

[22] Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. In *arXiv*, 2022.

[23] Abhishek Gupta, Vikash Kumar, Corey Lynch, Sergey Levine, and Karol Hausman. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2019.

[24] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Avnish Narayan, Hayden Shively, Adithya Bellathur, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning, 2021.

[25] Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Ruo Yu Tao, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, Wendy Tay, and Adam Trischler. Textworld: A learning environment for text-based games. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2018.

[26] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Neural Information Processing Systems (NeurIPS)*, 2022.

[27] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 18–24 Jul 2021.

A Parameters

In Webshop, all experiments were performed with a temperature setting of 0.0, due to the additional requirement of precise commands for interactions with the web application.

Table 4: Parameters

Experiments on ALFWorld	
Text Embedding Model	sentence-transformers/all-MiniLM-L6-v2
VLM	openai/CLIP, ViT-B/32
Max steps	50
GPT-3.5 version	gpt-3.5-turbo-instruct
GPT-3.5 temperature	0
GPT-4 version	gpt-4-0613
GPT-4 temperature	0.5
Llama2-13b version	meta-llama/Llama-2-13b-chat-hf
Llama2 temperature	default(1.0)
Experiments on Webshop	
Text Embedding Model	sentence-transformers/all-MiniLM-L6-v2
Max steps	15
GPT-3.5 temperature	0
Llama2-13b version	meta-llama/Llama-2-13b-chat-hf
Llama2 temperature	0

B Multimodal Environments and Agents Details

B.1 Task Selection

To evaluate RAP in multimodal environments, we focus on two benchmarks: Franka Kitchen and Meta-World. These benchmarks offer a diverse range of tasks, simulating real-world scenarios in a controlled environment, making them ideal for evaluating the performance of multimodal agents.

Franka Kitchen Benchmark simulates a kitchen environment, where the agent interacts with various kitchen appliances and items. We have selected five specific tasks that test the agent's ability to manipulate objects and controls within this environment. These tasks are listed in table 5

Task No.	**Task Description**
1. cabinet	Open a hinge cabinet door
2. micro	Open the microwave door
3. burner	Twist knob for bottom left burner
4. light	Move lever to turn on light
5. kettle	Move kettle from bottom to top burner

Table 5: Selected Tasks in Franka Kitchen Benchmark

The Meta-World benchmark is designed to evaluate an agent's skill in more generalized object manipulation tasks. We have selected five tasks that represent a broad range of actions.

Task No.	**Meta-World Tasks**
1. assemble	Pick up a nut and place it onto a peg
2.buttion	Press a button
3. drawer	Open a drawer
4. hammer	Hammer a screw on the wall
5. bin	Grasp the puck from one bin and place it into another bin

Table 6: Selected Tasks in Meta-World Benchmark

B.2 Model Specification

In this subsection, we detail the model specifications for our agent, which is built upon two VLMs: LLaVA and CogVLM. These models allows for a more comprehensive understanding and interactions.

LLaVA Model: Our agent utilizes the LLaVA-v1.5 13B model, which incorporates Vicuna-v1.5 13B as its underlying LLM to tap on its robust linguistic processing capabilities.

CogVLM Model: The CogVLM-17B model contains a significant number of parameters – 10 billion are dedicated to visual understanding, and 7 billion are focused on language processing.

In our experimental setup for the language generation of these models, we set the temperature to 0.0. This setting is chosen to prioritize precision and determinism in the generated outputs, which is crucial for the consistency and reliability of the agent's planning responses in our multimodal tasks.

B.3 Policy Network details

The policy network is a crucial component of our framework, designed to translate high-level action plans generated by the VLMs into precise, low-level control actions suitable for the specific action space of the environment. Our approach utilizes a Multi-Layer Perceptron (MLP) neural network for this purpose. In both environments, the learning of the policy network is facilitated through a few-shot learning approach, leveraging a limited but highly informative set of demonstration data.

For each environment, we provide 25 expert demonstrations sourced from the D4RL dataset. These demonstrations consist of trajectories that include both observations and actions, showcasing expert-level performance in the respective tasks.

In the Franka Kitchen tasks, each demonstration is composed of 50 state-action pairs, reflecting the sequence and specifics of actions required to complete the task. Meanwhile, for the Meta-World tasks, each demonstration sample comprises 500 state-action pairs.

C Ablation Study

C.1 Improvement of Success Rate through Recursive Evaluation

The following figure shows the progression of the improvement in success rate through recursive evaluation in ALFWorld and WebShop.

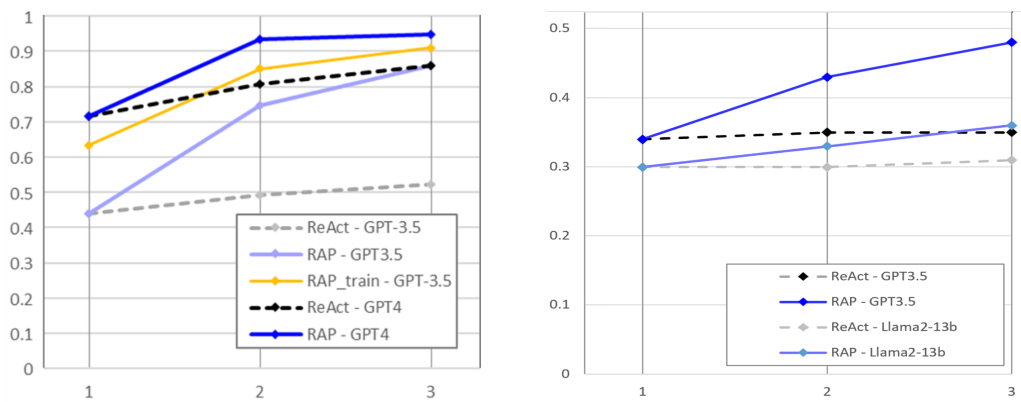

Figure 5: Success Rate over 3 trials, **Left**: ALFWorld, **Right**: WebShop

RAP outperform ReAct and it indicate the effective utility of successful experiences from other tasks in memory. In addition, RAP with GPT-3.5 achieves performance equivalent to ReAct with GPT-4. Also, RAP with memory built from the training set via GPT-3.5 surpasses ReAct with GPT-4 in ALFWorld.

C.2 Design of Retrieval Process

C.2.1 Evaluation across various Retrievers

We perform evaluation on ALFWorld with GPT-3.5 by varying the Retriever, as shown in Table 7. The results of RAP$_{act}$ and RAP$_{obs}$ illustrate the effectiveness of switching the information used for retrieval depending on the situation.

Furthermore, we utilize visual observation provided by ALFWorld instead of textual observation, and perform an evaluation using similarity between textual retrieval key and image observation with CLIP [27]. As a result, slightly better performance is demonstrated than when using text observation information. This suggests that employing direct image data, rather than information converted into text, could enable more effective retrieval.

Table 7: ALFWorld success rate(%) with different retrievers. RAP$_{act}$ uses only action information, while RAP$_{obs}$ utilizes only observation information for retrieval. RAP$_{clip}$ refers to evaluations using images, rather than texts, as observations with CLIP.

Method(d$_{max}$=3)	Success Rate
ReAct	52.2
RAP$_{act}$	82.1
RAP$_{obs}$	84.3
RAP$_{clip}$	**86.6**
RAP	<u>85.8</u>

Additionally we illustrate the effect of varying the components extracted from each experience by the Retriever in Webshop. Here, all evaluations are performed on Llama2-13b.

As shown in Table 8, through RAP$_{obs}$, the agent is able to retrieve trajectories from memory based on either actions or observations, depending on the current stage of solving the task. With the incorporation of intra-task similarity in RAP$_{intra}$, the agent is able to align the relationship between task information and the corresponding trajectories of each experience when projecting to the current task. By retrieving based on product category in RAP$_{cat}$, the agent is able to retrieve experiences that are more related to the current task. Overall, RAP takes into account these components, resulting in an overall boost of 6.5% and 5.0% for overall reward and success rate respectively on Llama2-13b. With these, RAP also demonstrates a boost of 14.3% and 13.0% for overall reward and success rate respectively on GPT-3.5 in Table 2, showcasing RAP's generalizability across different models.

Table 8: Webshop overall score (%) and success rate(%) with different retrievers on Llama2-13b. RAP$_{obs}$ uses additional retrieval by observations on top of action-based retrieval in RAP$_{act}$. Also, RAP$_{intra}$ and RAP$_{cat}$ uses intra-task retrieval and product-category retrieval. RAP indicates combination of RAP$_{obs}$, RAP$_{intra}$ and RAP$_{cat}$.

Method(d$_{max}$=3)	Success Rate	Overall Score
ReAct	31.0	64.6
RAP$_{act}$	33.0	68.6
RAP$_{obs}$	33.0	69.0
RAP$_{intra}$	34.0	69.3
RAP$_{cat}$	35.0	69.9
RAP	**36.0**	**71.1**

C.2.2 Weights of components in Retriever

An ablation study was performed on the effectiveness of each component in Equation 3 for the Retriever. Here, a modified-condition/decision coverage (MC/DC) test scenario was adopted to analyze the effect of each variable.

Following Equation 3, the retrieval process for ALFWorld is based on similarity with memory logs, and is calculated as:

1. $w_t * sim(T_0, T_{Li})$ has 2 parts: (a) similarity of task category with log L_i, (b) similarity of task description with log L_i
2. $w_p * sim(p_0, p_{Li})$: similarity of overall plan with scenario of log L_i
3. $w_k * sim(k_0, \tau_{Li})$: similarity of action/observation of current task with trajectory of log L_i

Additionally, to validate the effectiveness of similarity, we conducted random retrievals as RAP_{random} for verification.

Table 9: Weights used in 3 for Retriever, on ALFWorld, with GPT-3.5

Config	wt_{cat}	wt_{des}	w_p	w_k	Success (%)
ReAct	0	0	0	0	52.2
RAP_{random}	-	-	-	-	65.7
RAP_t	1	1	0	0	80.6
RAP_{tp}	1	1	1	0	84.3
RAP	1	1	1	1	**85.8**

The results of the ablation study were tabulated in Table 9. Here, each component was added incrementally, beginning with the addition of task similarity (Row 3), followed by similarity of overall plan (Row 4), and lastly action-observation similarity (Row 5). From these results, it can be confirmed that retrieval based on similarity positively impacts performance and is effective.

Furthermore, following Equation 3, the retrieval process for WebShop is based on similarity with memory logs, and is calculated as:

1. $w_t * sim(T_0, T_{Li})$: similarity of product category of current task with category of log L_i
2. $w_p * sim(p_0, p_{Li})$: similarity of overall plan with scenario of log L_i
3. $w_k * sim(k_0, tau_{Li})$ has 2 parts: (a) similarity of retrieval key of task with trajectory of log L_i, (b) similarity of trajectory of log L_i with its own task scenario

By contrasting against results in row 2 of Table 10, there was an increase in reward and success rate for each variable of observation (row 3), intra-task relationship (row 4), and category (row 5). As these components are independent, the combination in row 6 showcases synergy among them.

Table 10: Weights used in 3 for Retriever, on WebShop, with LLama2-13b

Config	w_t	w_p	w_{intra}	$w_{act,obs}$	Reward (%)	Success (%)
ReAct	0	0	0	0	64.6	31.0
RAP_{act}	1	0	0	0	68.6	33.0
RAP_{obs}	1	0	0	1	69.0	33.0
RAP_{intra}	1	0	1	0	69.3	34.0
RAP_{cat}	1	1	0	0	69.9	35.0
RAP	1	1	1	1	**71.7**	**36.0**

C.2.3 Number of experiences retrieved

Apart from the comparisons of similarity of memory logs with the current task, another crucial aspect of the proposed algorithm is selecting the right number of experiences to be passed to the agent.

As such, an ablation study was performed on the ideal number of experiences retrieved from the memory by the Retriever. From Table 11, it was confirmed that the performance improves as the number of experiences utilized increases.

Table 11: Number of retrieved experiences, on ALFWorld, with GPT-3.5

Num examples in action retrieval	Num examples in observation retrieval	Success (%)
2	2	81.3
4(default)	8(default)	**85.8**

Table 12: Number of retrieved experiences, on WebShop, with Llama2-13b

Num examples	Reward (%)	Success (%)
1	66.3	32.0
2	68.9	34.0
3(default)	**71.1**	**36.0**

From Table 12, it can be observed that the highest success rate and reward was observed when 3 experiences were retrieved from the memory. Due to prompt length limitation of the agent, together with the long observations returned by WebShop, the number of experiences were limited to 3.

C.3 Transfer Learning via Memory

RAP is capable of utilizing past experiences that are stored in memory. Since the experience of solving tasks is independent of the model, the model used for evaluation does not need to match the one used for memory construction. Here, we illustrate a verification of transfer learning between models by using memory constructed via different models for the evaluation model. From Section 4.1.1, we use 1000 samples from training data, but no recursive trial is conducted ($d_{max} = 1$) to simply verify the effect of transfer learning.

Table 13 shows results of transfer learning, which indicate memory generated with GPT-3.5 is also effective in Llama2-13b. Thus, RAP allows sharing experiences across models.

Table 13: ALFWorld success rate(%) with Memory and d_{max}=1. Model_{Memory} indicates the language model used to construct memory from the training data.

Model	Model_{Memory}	Success Rate
GPT-3.5	-	44.0
GPT-3.5	GPT-3.5	63.4
Llama2-13b	-	20.9
Llama2-13b	Llama2-13b	26.9
Llama2-13b	GPT-3.5	27.6

D Prompts

D.1 ALFWorld Prompt

D.1.1 Prompt for overall plan in Reasoner

Here are examples.
{Your task is to: put some vase on safe.
$>$ To solve the task, I need to find and take a vase, then put it on safe.**} examples x 3**

Here is the task. Please make a plan from the examples.
Your task is to: put some watch on safe.
$>$ think: To solve the task,

D.1.2 Prompt for retrieval key in Reasoner

Here are examples.
think: First I need to find a spraybottle. A spraybottle is more likely to appear in cabinet (1-4), countertop (1), toilet (1), sinkbasin (1-2), garbagecan (1). I can check one by one, starting with cabinet 1.
> search: spraybottle
think: Now I put the first creditcard in dresser. Next, I need to find the second creditcard. I can directly go to countertop 1.
> search: creditcard
think: Now I take a pen (2). Next, I need to find a desklamp. A desklamp is more likely to appear in dresser (1), shelf (1-9), bed (1), garbagecan (1), drawer (1-10). I can check one by one, starting with dresser 1.
> search: desklamp
think: Now I find a lettuce (1). Next, I need to take it.
> action: take
think: Now I find a pan (1). Next, I need to take it.
> action: take
think: Now I find the second saltshaker (2). Next, I need to take it.
> action: take
think: Now I heat an egg (2). Next, I need to put it in/on diningtable 1.
> action: put
think: Now I take a spraybottle (2). Next, I need to put it in/on toilet 1.
> action: put
think: Now I take an apple (1). Next, I need to go to a microwave (1) and heat it.
> action: heat
think: Now I take a bread (1). Next, I need to go to a microwave (1) and heat it.
> action: heat
think: Now I take a mug (3). Next, I need to go to a fridge (1) and cool it.
> action: cool
think: Now I take a potato (2). Next, I need to go to a fridge (1) and cool it.
> action: cool
think: Now I find a desklamp (1). Next, I need to use it.
> action: use
think: Now I find a desklamp (3). Next, I need to use it.
> action: use

Here is the task. Please make a plan from the examples.
think: First I need to find a watch. A watch is more likely to appear in/on cabinet (1-4), drawer (1-5). I can check one by one, starting with cabinet 1
>

D.1.3 Prompt for action plan in Reasoner and for action in Executor

Here is the task information.
You are in the middle of a room. Looking quickly around you, you see a armchair 1, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a drawer 5, a drawer 4, a drawer 3, a drawer 2, a drawer 1, a dresser 1, a garbagecan 1, a safe 1, a shelf 12, a shelf 11, a shelf 10, a shelf 9, a shelf 8, a shelf 7, a shelf 6, a shelf 5, a shelf 4, a shelf 3, a shelf 2, a shelf 1, a sidetable 1, and a sofa 1.

Here are examples.
{Task: put a vase in safe.
Plan: To solve the task, I need to find and take a vase, then put it in the safe.
> think: First I need to find a vase.
OK.
> go to shelf 1
On the shelf 1, you see nothing.
...
> go to shelf 6
On the shelf 6, you see a vase 2.
> think: Now I find a vase (2). Next, I need to take it
OK.
> take vase 2 from shelf 6
You pick up the vase 2 from the shelf 6.**} examples x n**

Here is the task. Please make an action from the examples.
Task: put some watch on safe.
Plan: To solve the task, I need to find and take a watch, then put it on the safe.
>

E Memory

E.1 ALFWorld memory example

```
{
    "Task": "examine the pencil with the desklamp.",
    "Category": "examine",
    "Overall Plan": "To solve the task, I need to find and take a pencil, then find and use a desklamp.",
    "Trajectory": [
        "> think: First I need to find a pencil",
        "OK.",
        "> go to shelf 2",
        "On the shelf 2, you see a alarmclock 1, a pen 2, and a pencil 1.",
        "> think: Now I find a pencil (1). Next, I need to take it",
        "OK.",
        "> take pencil 1 from shelf 2",
        "You pick up the pencil 1 from the shelf 2.",
        "> think: Now I take a pencil (1). Next, I need to find and use a desklamp",
        "OK.",
        "> go to desk 1",
        "On the desk 1, you see a alarmclock 2, a cellphone 2, a desklamp 1, a keychain 2, a mug 1, and a pencil 3.",
        "> think: Now I find a desklamp (1). Next, I need to use it",
        "OK.",
        "> use desklamp 1",
        "You turn on the desklamp 1."
    ]
}
```

E.2 Webshop memory example

```
{
    "Category": "beauty",
    "Query": "hair extensions, wigs & accessories",
    "Overall Plan": "i need a long clip-in hair extension which is natural looking, and price lower than 40.00 dollars",
    "Trajectory": [
        "Action: search[long clip-in hair extension natural looking]",
        "Observation: \n[Back to Search] \nPage 1 (Total results: 50) \n[Next >] \n[B09QQLDJ93] \nLroveb Fashion Long Curly Wig Head Dyed Black Gradient Golden Long Straight Wig African Long Curly Hair Wigs Ladies Gradient Color Chemical Fiber Silk Corn Perm Wig for Women \n$34.89 \n[B09QQKPZJR] \nLroveb Beanie Hat Knit with Long Wavy Curly Hair Extensions Wig Winter Warm Fashion Long Curly Hair Hat Wig Wool Curl Corn Perm Curl Knitted Cap Attached Synthetic Hairpiece \n$36.06 \n[B08H5DCD65] \nCorn Wave Ponytail Extension Clip in - 22 Inch Long Wavy Curly Wrap Around Pony Tail Heat Resistant Synthetic Hairpiece for Women (Natural Black #1B) \n$9.98 ",
        "Action: think[B09QQLDJ93 and B09QQKPZJR are long clip-in hair extensions that are natural looking and less than 40 dollars. I can check B09QQLDJ93 first.]",
```

```
        "Observation: OK.",
        "Action: click[B09QQLDJ93]",
        "Observation: \n[Back to Search] \n[< Prev] \nLroveb Fashion Long
        ↪  Curly Wig Head Dyed Black Gradient Golden Long Straight Wig
        ↪  African Long Curly Hair Wigs Ladies Gradient Color Chemical
        ↪  Fiber Silk Corn Perm Wig for Women \nPrice: $34.89 \nRating:
        ↪  N.A. \n[Description] \n[Features] \n[Reviews] \n[Attributes]
        ↪  \n[Buy Now] ",
        "Action: think[For long clip-in hair extension that is natural
        ↪  looking, the item has options 'long curly wig' and 'dyed black
        ↪  gradient golden' and seems good to buy.]",
        "Observation: OK.",
        "Action: click[Buy Now]"
    ],
}
```

F Trajectories

F.1 ALFWorld trajectories

Task Description

You are in the middle of a room. Looking quickly around you, you see a bed 1, a desk 2, a desk 1, a drawer 6, a drawer 5, a drawer 4, a drawer 3, a drawer 2, a drawer 1, a garbagecan 1, a laundryhamper 1, a safe 1, a shelf 6, a shelf 5, a shelf 4, a shelf 3, a shelf 2, and a shelf 1.
Your task is to: put a mug in desk.

ReAct

Act 1: think: To solve the task, I need to find and take a mug, then put it in desk
Obs 1: OK.
Act 2: think: First I need to find a mug. A mug is more likely to appear in drawer (1-6), shelf (1-6), desk (1-2), garbagecan (1), laundryhamper (1), safe (1). I can check one by one, starting with drawer 1
Obs 2: OK.
Act 3: go to drawer 1
Obs 3: The drawer 1 is closed.
Act 4: open drawer 1
Obs 4: You open the drawer 1. The drawer 1 is open. In it, you see a pen 1.
...
Act 23: think: Hmm, it seems like the mug is not in any of the drawers. Let me check the other possible locations
Obs 23: OK.
...
Fail

RAP

Plan: To solve the task, I need to find and take a mug, then put it in desk.
Act 1: think: First I need to find a mug
Obs 1: OK.
Retrieval key: search mug
Act 2: go to shelf 1
Obs 2: On the shelf 1, you see a mug 2.
Act 3: take mug 2 from shelf 1
Obs 3: You pick up the mug 2 from the shelf 1.
Act 4: think: Now I find a mug (2). Next, I need to put it in/on desk 1
Obs 4: OK.
Retrieval key: action put
Act 5: go to desk 1
Obs 5: On the desk 1, you see a bowl 1, a cd 2, a creditcard 2, a desklamp 1, a keychain 1, a laptop 2, a pen 2, and a pencil 2.
Act 6: put mug 2 in/on desk 1
Obs 6: You put the mug 2 in/on the desk 1.
Success

Task Description

You are in the middle of a room. Looking quickly around you, you see a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a countertop 1, a garbagecan 1, a handtowelholder 2, a handtowelholder 1, a sinkbasin 2, a sinkbasin 1, a toilet 1, a toiletpaperhanger 1, and a towelholder 1.
Your task is to: clean some cloth and put it in cabinet.

ReAct

Act 1: think: To solve the task, I need to find and take a cloth, then clean it with sinkbasin, then put it in a cabinet
Obs 1: OK.
Act 2: think: First I need to find a cloth. A cloth is more likely to appear in a cabinet (1-4), countertop (1), sinkbasin (1-2), handtowelholder (1-2), towelholder (1), toiletpaperhanger (1), garbagecan (1). I can check one by one, starting with cabinet 1
Obs 2: OK.
...
Act 6: open cabinet 2
Obs 6: You open the cabinet 2. The cabinet 2 is open. In it, you see a cloth 1.
Act 7: take cloth 1 from cabinet 2
Obs 7: You pick up the cloth 1 from the cabinet 2.
Act 8: think: Now I take a cloth (1). Next, I need to go to sinkbasin (1-2) and clean it
Obs 8: OK.
Act 9: go to sinkbasin 1
Obs 9: On the sinkbasin 1, you see nothing.
Act 10: go to sinkbasin 2
Obs 10: On the sinkbasin 2, you see nothing.
Act 11: think: I can't clean the cloth without water. I need to find a source of water
Obs 11: OK.
Act 12: go to toilet 1
...
Fail

RAP

Plan: To solve the task, I need to find and take a cloth, then clean it with sinkbasin, then put it in cabinet.
Act 1: think: First I need to find a cloth
Obs 1: OK.
Retrieval key: search cloth
Act 2: go to cabinet 1
Obs 2: The cabinet 1 is closed.
Act 3: open cabinet 1
Obs 3: You open the cabinet 1. The cabinet 1 is open. In it, you see a toiletpaper 1.
Act 4: go to cabinet 2
Obs 4: The cabinet 2 is closed.
Act 5: open cabinet 2
Obs 5: You open the cabinet 2. The cabinet 2 is open. In it, you see a cloth 1.
Act 6: take cloth 1 from cabinet 2
Obs 6: You pick up the cloth 1 from the cabinet 2.
Act 7: think: Now I take a cloth (1). Next, I need to go to sinkbasin (1) and clean it
Obs 7: OK.
Retrieval key: action clean
Act 8: go to sinkbasin 1
Obs 8: On the sinkbasin 1, you see nothing.
Act 9: clean cloth 1 with sinkbasin 1
Obs 9: You clean the cloth 1 using the sinkbasin 1.
Act 10: think: Now I clean a cloth (1). Next, I need to put it in/on cabinet (1-6)
Obs 10: OK.
Retrieval key: action put
Act 11: go to cabinet 3
Obs 11: The cabinet 3 is closed.
Act 12: open cabinet 3
Obs 12: You open the cabinet 3. The cabinet 3 is open. In it, you see a soapbottle 1.
Act 13: put cloth 1 in/on cabinet 3
Obs 13: You put the cloth 1 in/on the cabinet 3.
Success

F.2 Webshop trajectories

Task Description

WebShop
Instruction:
i am looking for mn4 color foundation for my sensitive skin, and price lower than 40.00 dollars
[Search]

ReAct

Action: search[mn4 color foundation sensitive skin]
Observation: [Back to Search] Page 1 (Total results: 50) [Next >] [B077PR9TL4] Physicians Formula The Healthy Foundation with SPF 20, LW2, 1 Fl Oz $7.59 [B07VLKYJNH] PÜR 4-in-1 Love Your Selfie Long-wear Foundation & Concealer - Full Coverage Foundation & Concealer Makeup For Dark Spots, Blemishes and Imperfections - Long-Lasting Liquid Makeup & Skincare $36.0 [B09J1X9M1P] Neutrogena Healthy Skin Sensitive Skin Serum Foundation with Pro-Vitamin B5, Color Correcting & Pore Minimizing Liquid Foundation & Face Serum, Buildable Coverage, Medium 02, 1 oz $14.39

Action: think[B077PR9TL4 and B09J1X9M1P are mn4 color foundation for sensitive skin and price lower than 40 dollars.]
Observation: OK.

Action: click[B077PR9TL4]
Observation: [Back to Search] [< Prev] color [1][dc1][dn3][dn4][lc1][ln3][ln4][lw2] [mc1][mn3][mn4][mw2] size [1 fl oz (pack of 1)][1 ounce (pack of 1)][1 fluid ounce] Physicians Formula The Healthy Foundation with SPF 20, LW2, 1 Fl Oz Price: $7.59 Rating: N.A. [Description] [Features] [Reviews] [Attributes] [Buy Now]

Act 11: think[For mn4 color foundation for sensitive skin, the item has options 'lw2' and '1 fluid ounce' and seems good to buy.]
Observation: OK.

Action: click[lw2]
Observation: You have clicked lw2.

Action: click[Buy Now]
Observation: Your score (min 0.0, max 1.0): 0.6666666666666666

Fail

RAP

Action: search[mn4 color foundation sensitive skin]
Observation: [Back to Search] Page 1 (Total results: 50) [Next >] [B077PR9TL4] Physicians Formula The Healthy Foundation with SPF 20, LW2, 1 Fl Oz $7.59 [B07VLKYJNH] PÜR 4-in-1 Love Your Selfie Long-wear Foundation & Concealer - Full Coverage Foundation & Concealer Makeup For Dark Spots, Blemishes and Imperfections - Long-Lasting Liquid Makeup & Skincare $36.0 [B09J1X9M1P] Neutrogena Healthy Skin Sensitive Skin Serum Foundation with Pro-Vitamin B5, Color Correcting & Pore Minimizing Liquid Foundation & Face Serum, Buildable Coverage, Medium 02, 1 oz $14.39

Action: think[B077PR9TL4, B07VLKYJNH, and B09J1X9M1P are mn4 color foundation for sensitive skin and price lower than 40.00 dollars.]
Observation: OK.

Action: click[B077PR9TL4]
Observation: [Back to Search] [< Prev] color [1][dc1][dn3][dn4][lc1][ln3][ln4][lw2] [mc1][mn3][mn4][mw2] size [1 fl oz (pack of 1)][1 ounce (pack of 1)][1 fluid ounce] Physicians Formula The Healthy Foundation with SPF 20, LW2, 1 Fl Oz Price: $7.59 Rating: N.A. [Description] [Features] [Reviews] [Attributes] [Buy Now]

Action: think[For mn4 color foundation for sensitive skin, the item has options '1', 'dc1', 'dn3', 'dn4', 'lc1', 'ln3', 'ln4', 'lw2','mc1','mn3','mn4', and'mw2' and seems good to buy.]
Observation: OK.

Action: click[mn4]
Observation: You have clicked mn4.

Action: click[Buy Now]
Observation: Your score (min 0.0, max 1.0): 1.0

Success

Task Description

WebShop
Instruction:
i want capri sun pacific cooler mixed fruit naturally flavored juice drinks, and price lower than 20.00 dollars
[Search]

ReAct

Action: search[capri sun pacific cooler mixed fruit naturally flavored juice drinks]
Observation: [Back to Search] Page 1 (Total results: 50) [Next >] [B000R30X2A] Capri Sun Pacific Cooler Mixed Fruit Naturally Flavored Juice Drink Blend (Pack of 10) (Packaging May Vary) $2.55 [B075CSWGJR] Capri Sun Pacific Cooler Ready-to-Drink Juice, 6 Fl Oz (Pack of 30) $100.0 [B0083GG2SC] Capri Sun Coolers Variety Pack Ready-to-Drink Juice (40 Pouches, 4 Boxes of 10) $100.0

Action: think[B000R30X2A is capri sun pacific cooler mixed fruit naturally flavored juice drinks and price lower than 20.00 dollars.]
Observation: OK.

Action: click[B000R30X2A]
Observation: [Back to Search] [< Prev] flavor name [assorted][pacific cooler][strawberry kiwi][variety pack] size [6 fl oz (pack of 10)][6 fl oz (pack of 30)] Capri Sun Pacific Cooler Mixed Fruit Naturally Flavored Juice Drink Blend (Pack of 10) (Packaging May Vary) Price: $2.55 Rating: N.A. [Description] [Features] [Reviews] [Attributes] [Buy Now]

Action: think[For capri sun pacific cooler mixed fruit naturally flavored juice drinks, the item has options 'assorted', 'pacific cooler', and '6 fl oz (pack of 10)']
Observation: OK.

Action: click[assorted]
Observation: You have clicked assorted.

Action: click[Buy Now]
Observation: Your score (min 0.0, max 1.0): 0.6666666666666666

Fail

RAP

Action: search[capri sun pacific cooler mixed fruit naturally flavored juice drinks]
Observation: [Back to Search] Page 1 (Total results: 50) [Next >] [B000R30X2A] Capri Sun Pacific Cooler Mixed Fruit Naturally Flavored Juice Drink Blend (Pack of 10) (Packaging May Vary) $2.55 [B075CSWGJR] Capri Sun Pacific Cooler Ready-to-Drink Juice, 6 Fl Oz (Pack of 30) $100.0 [B0083GG2SC] Capri Sun Coolers Variety Pack Ready-to-Drink Juice (40 Pouches, 4 Boxes of 10) $100.0

Action: think[B000R30X2A is capri sun pacific cooler mixed fruit naturally flavored juice drinks and seems good to buy.]
Observation: OK.

Action: click[B000R30X2A]
Observation: [Back to Search] [< Prev] flavor name [assorted][pacific cooler][strawberry kiwi][variety pack] size [6 fl oz (pack of 10)][6 fl oz (pack of 30)] Capri Sun Pacific Cooler Mixed Fruit Naturally Flavored Juice Drink Blend (Pack of 10) (Packaging May Vary) Price: $2.55 Rating: N.A. [Description] [Features] [Reviews] [Attributes] [Buy Now]

Action: think[For capri sun pacific cooler mixed fruit naturally flavored juice drinks, the item has options 'assorted', 'pacific cooler', and '6 fl oz (pack of 10)' and seems good to buy.]
Observation: OK.

Action: click[assorted]
Observation: You have clicked assorted.

Action: click[pacific cooler]
Observation: You have clicked pacific cooler.

Action: click[6 fl oz (pack of 10)]
Observation: You have clicked 6 fl oz (pack of 10).

Action: click[Buy Now]
Observation: Your score (min 0.0, max 1.0): 1.0

Success