From Exploration to Reuse: An Embodied Agent Framework for Manipulation Skill Learning

Abstract

Recent advances in Vision Language Models (VLMs) have enabled agentic robotic frameworks capable of interpreting natural language instructions and generating robot commands. However, existing approaches predominantly rely on either zero-shot execution with no modular reuse of learned skills or task-specific policy learning that must be retrained for each new behaviors. In this paper, we propose an embodied dual-agent hybrid architecture that allows agents to acquire new skills by exploring inputs and hyperparameter spaces of model-based controllers and reusing successfully learned, interpretable skills through a memory skill module in future tasks. The exploratory behavior is enabled through the interaction between VLM planning and monitoring agents, where the former determines a long-term plan and suggests the next skill and the latter explores skill changes that can lead to task success. Thus, we call this framework Model-based Acquisition and Retrieval of Skills (MARS). We showcase MARS across different embodiments in simulation and real-world through a variety of prehensile and non-prehensile, short and long-horizon tasks. Ablation studies on multiple embodiments in real-world and simulation show that the monitoring agent improves success rates from 25% to 77%, while decreasing the number of trials till task success from 6.25 to 1.5. While the skill memory module improves success rates from 39% to 70.5% and allows reusing strategies learned from previous interactions.

1 Introduction & Related Work

The goal of building robots that can understand and respond to free-form natural language instructions has been a long-standing challenge in robotics and AI research [Lynch *et al.*2022]. Rather than relying on rigid task identifiers or goal images to specify behaviors, natural language offers a flexible and intuitive interface that scales to a wide variety of tasks without requiring predefined vocabularies or grammars [Lynch *et al.*2022]. In this setting, a single agent must execute a diverse series of manipulation tasks each expressed in natural language, performing all tasks sequentially and in any order [Lynch *et al.*2022]. This requires the agent to simultaneously understand perception from raw sensory inputs, understand language, and master motor control as a single system. Despite recent progress in Embodied AI and VLM-based robotics, a persistent limitation across most systems is the inability to autonomously acquire new low-level skills, store them in a structured manner, and reuse them effectively in future tasks [Salimpour *et al.*2025].

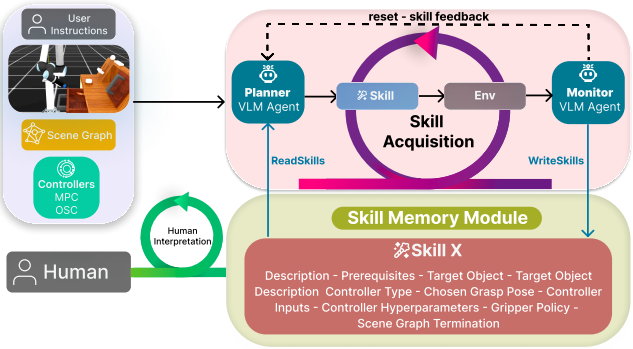

Figure 1: Overview of MARS. Its hierarchical design separates high-level planning from low-level model-based skill execution, while the symbolic skill structure enables persistent reuse across sessions and embodiments and interpretable skill execution and adaptation.

Many existing agentic robotic frameworks rely on a fixed set of predefined atomic or composite actions, which constrains their adaptability in dynamic, real-world environments [Qu *et al.*2026]. While these approaches enable flexible high-level planning, their action spaces are static, preventing the agent from extending its behavioral repertoire through experience. Other methods introduce new skills via imitation learning, requiring explicit human demonstrations for each new task [Mower *et al.*2024]. A separate line of work explores learning low-level behaviors through optimization guided by LLM-generated rewards [Yu *et al.*2023, Patel *et al.*2025], virtual potential fields [Huang *et al.*2023], or model-based controllers [Wei *et al.*2024]. While these methods demonstrate impressive zero-shot generalization, they typically execute skills in an open-loop manner and do not provide mechanisms for reusing the resulting policies. As a

result, newly acquired behaviors are discarded after execution, preventing cumulative skill acquisition and adaptation.

To address these limitations, we propose **MARS** (**M**odel-based **A**cquisition and **R**etrieval of **S**kills), an embodied dual-agent architecture in which two cooperating VLM agents jointly explore the parameter spaces of model-based controllers to acquire new manipulation skills and reuse successfully learned skills through a persistent skill memory module, as shown in Figure 1. A *planning agent* determines a long-horizon plan and proposes the next skill to execute, while a *monitoring agent* evaluates execution outcomes, explores corrective skill configurations, and commits successful skills to memory. In this work, a *skill* is defined as an atomic behavior that induces a targeted change in the environment, represented as a goal in the scene graph. Each skill is paired with a low-level model-based controller, whose hyperparameters and inputs determine the behavior and success of the skill. For each task, the system can explore multiple candidate skill configurations and retain the most reliable configuration based on execution feedback. We rely on a semantic 3D Scene Graph for both short and long-horizon planning, where nodes represent objects and edges represent spatial and physical relationships. The proposed formulation avoids voxel-based representations, static skill libraries, and imitation-learned policies, while remaining few-shot, reusable, and agnostic to specific environments and robots.

The main contributions of this paper are as follows:

- A hybrid planning architecture with symbolic skill formulations, in which dual VLM agents acquire new manipulation skills online through few-shot exploration of model-based controller parameter spaces.
- A persistent skill memory module that stores learned skills augmented with semantic and relational metadata, enabling interpretable adaptation and reuse across tasks and objects.

By comparing the full MARS framework to different baselines in an ablation study across different embodiments and custom short and long-horizon tasks, we show the dominant performance of dual-agent design and the skill memory retention module across CALVIN-style [Mees *et al.*2022] tasks and custom tabletop manipulation tasks.

2 Methodology

2.1 Problem Statement

Given a natural language instruction I, we consider the problem of generating a series of consecutive skills $\{\sigma_t\}$ that achieve the target task, where a skill σ_{t+1} is generated and executed immediately after σ_t terminates. A skill is defined as a high-level Bauplan of an environment-changing robotic behavior: **name**, **description**, **target object** and **target-object description**, **controller type**, **controller inputs**, **controller hyperparameters**, and **preconditions**. The name provides a symbolic identifier for referencing and reuse, while the description captures the strategy of the behavior. The target object specifies the entity to be manipulated, its description disambiguates that entity using visual attributes, and the chosen grasp pose describes the SE(3) pose selected for contacting the object. The controller type defines the low-level control family, controller inputs define the task-level goal variables passed to that controller, and controller hyperparameters define execution characteristics. The gripper policy meanwhile can be considered a separate controller structure with only binary conditions that either open or close the gripper. Constraints—including preconditions, timeout, and termination conditions—are integral parts of the skill; preconditions are critical for safe, context-correct execution, while termination conditions may be explicitly empty (`None`) when timeout- or state-based completion is sufficient for achieving the task. The problem inputs are augmented with the following: a library of pre-defined low-level controllers $\{C_1, C_2, ..C_n\}$, a Scene Graph $S_t =< V, E >$, where V is a set of detected object nodes and E is the set of inter-object relationships, and access to a library of pre-learned skills $\{\sigma_p\}$ if available. Thus, the system π

$$\pi(\sigma_t \mid \sigma_{\{0,\, t-1\}},\ \{C_1, C_2, ..C_n\},\ S_t,\ \{\sigma_p\})$$

samples a skill σ_t given the inputs and the history of previously applied skills.

2.2 MARS Dual-Agent Skill Acquisition

Let I denote the user instruction, x_t the RGB observation at step t, S_t the scene graph, and $\mathcal{C} = \{C_1, \ldots, C_n\}$ the controller library. Let $\mathcal{M}$ denote the skill memory.

The planning agent P proposes the next skill (or a stop token $\varnothing$), where h_t is the interaction history including the initial user message and agent intermediate outputs.

The monitoring agent M evaluates the execution given the skill and the before and after scene graphs and images, returning a structured outcome

$$(o_t, f_t, \tilde{\sigma}_t, r_t, m_t) = M(I, x_t, S_t, \sigma_t, x_{t+1}, S_{t+1}),$$

where $o_t \in \{\texttt{success}, \texttt{failure}\}$ is skill success, f_t is text feedback for re-planning, $\tilde{\sigma}_t$ is an optional refined skill variant, $r_t \in \{0, 1\}$ indicates whether to reset the environment, and $m_t \in \{0, 1\}$ indicates whether the (possibly refined) skill should be written to memory. The monitoring agent is instructed to reset if and only if the environment has been corrupted and hinders further learning or if too many skill trials have been explored, causing the history and context window to grow larger.

We save the skills on desk as JSON documents and add the growing set of skills to both agents prompts as execution continues. In simulation, MARS uses two low-level controllers in IsaacLab depending on the skill: a model predictive controller (MPC) [Grüne and Pannek2011] for goal-directed end-effector motion and an operational space controller (OSC) [Khatib1987] for task-space wrench force controller. We implement MPC as a nonlinear program in a receding horizon manner [Grüne and Pannek2011], producing SE(3) way-points that are converted to joint-space commands through the IsaacLab IK controller, while OSC directly tracks target pose–wrench commands with axis-wise stiffness and force gains [Khatib1987]. In the real world Igus Rebel,real-world skills are executed as asynchronous end-effector pose commands with velocity and timeout constraints rather than MPC rollouts. Each skill also includes a gripper policy and a

termination rule: the gripper state is governed either by an always open/close behavior or by pose-triggered binary conditions of the form $d_t < \tau$ or $d_t > \tau$, while termination is evaluated from scene-graph relations as $\beta_\sigma(S_t) = \texttt{TERMINATE}$ if and only if all required relations hold and all forbidden relations are absent. We build the scene graph using a lightweight manipulation-oriented pipeline built on ConceptGraphs [Gu *et al.*2023]. From RGB-D observations, we detect and segment objects, lift masked pixels into 3D point clouds, fit object-level 3D bounding boxes, infer semantic relations, and output a structured graph whose nodes store object identity, captions, 3D geometry, and heuristic candidate grasp poses (top, left, right), while edges encode manipulation-relevant relations including `on top of`, `inside`, `next to`, and `gripping object`. This scene graph serves as the shared state for both agents.

3 Experimental Design

We evaluate MARS in simulation and the real world across multiple embodiments. In IsaacLab, we use a custom tabletop scene with a cup, basket, and box, and a CALVIN C desk scene [Mees *et al.*2022] with a drawer, slider, button, and cube. Simulation experiments use a UR10 with a vacuum gripper and a UR10E with a Robotiq 2F-85 gripper, on tasks including drawer opening/closing, slider movement, button pressing, cube pick-and-place, and cup placement into a basket. Real-world experiments use a 6DOF Igus Rebel with a vacuum gripper in a tabletop scene, where the tasks are placing a cup or a concave box into a basket; the latter requires side grasps. For planning, monitoring, and scene-graph captioning, we use GPT-4o, GPT-5o, and GPT-5-mini, respectively. In simulation, both IsaacLab and the scene-graph pipeline run on a single NVIDIA RTX 6000 machine. Across embodiments, skills are kept fixed except for a lightweight adapter that transforms target end-effector quaternions into the tool frame. We first evaluate short-horizon skills through two ablation studies isolating the effects of the monitoring agent and the skill memory module. Specifically, we compare the full system against a planning-only baseline, a planning-plus-memory baseline, and a dual-agent variant without skill memory. Performance is measured by success rate S_R and trials to first success T_S over $T \in \{5, 7, 10\}$ trials, with $T_S = T$ indicating failure and $T_S = 0$ indicating zero-shot skill success.

4 Results

In the first ablation study in Table 1, MARS shows a large performance gap relative to the baselines on both metrics, achieving substantially lower T_S and markedly higher S_R across almost all tasks, achieving a mean S_R of 77% and T_S of 1.5, compared to only 25% S_R and 6.25 T_S. In the second ablation study in Table 2, the baseline without skill memory achieves the same mean T_S as MARS, with both variants reaching 1.5. However, the difference in success rate remains large: the version without skill memory reaches only 39%, whereas MARS achieves 70.5%. This gap indicates that skill memory improves consistency because without it the agent must repeatedly relearn successful skill configurations across trials instead of reusing them.

Table 1: MARS ablation with and without monitor agent.

Task	**Robot**	**Metric**	**Planner-Only**	**Planner-Memory**	**MARS**
Open Drawer	UR10E (sim)	T_S	10	10	2
		S_R	0%	0%	80%
Close Drawer	UR10 (sim)	T_S	10	10	1
		S_R	0%	0%	70%
Move Slider	UR10E (sim)	T_S	10	10	2
		S_R	0%	0%	70%
Press Button	UR10E UR10 (sim)	T_S	5	5	1
		S_R	0%	0%	80%
Pick and Place Cube	UR10 (sim)	T_S	0	0	0
		S_R	100%	100%	100%
Pick and Place Cup in Basket	UR10 (sim)	T_S	10	10	4
		S_R	0%	0%	60%
Pick and Place Concave Box	Igus Rebel (real)	T_S	2	3	1
		S_R	29%	43%	71%
Pick and Place Cup	Igus Rebel (real)	T_S	2	2	1
		S_R	29%	57%	86%
Mean		T_S	6.13	6.25	**1.5**
		S_R	20%	25%	**77%**

Table 2: MARS ablation with and without memory.

Task	**Metric**	**MARS-NoMemory**	**MARS**
Close Drawer (sim)	T_S	2	2
	S_R	20%	70%
Pick and Place Concave Box (real)	T_S	1	1
	S_R	57%	71%
Mean	T_S	1.5	**1.5**
	S_R	39%	**70.5%**

T_S denotes the number of trials until the first successful execution, where lower values indicate faster skill acquisition. S_R denotes the success rate across all trials, reported as a percentage, where higher values indicate more consistent reuse of successful skill configurations. MARS shows dominant performance in S_R against all baselines and T_S against baselines without monitor agent.

Figure 2 shows sample runs, with the top figure showing the monitoring agent searching for the correct skill configuration by generating insightful feedback and skill suggestions. The bottom figure shows how an existing skill is used to infer the correct strategy for grasping a new object with the same topology. The primary sources of errors were: wrong grasp and placement poses, wrong force directions, pose detection errors, and vacuum gripper release.

5 Conclusion and Future Work

We presented MARS, a dual-agent framework for online manipulation skill acquisition and reuse. By combining symbolic planning with closed-loop monitoring, MARS explores skill configurations, stores successful ones, and reuses them in later tasks. The monitoring agent is the key contributor: without it, the planning-only system reaches only 25% mean success, while full MARS achieves 77% and finds a working configuration in an average of 1.5 trials. Skill memory further improves reliability, increasing success from 39% to 70.5% by reusing validated configurations instead of relearning them. Limitations include the separation of skill acquisition and long-horizon execution, usage of the external scene graph with substantial time latency and static candidate grasp

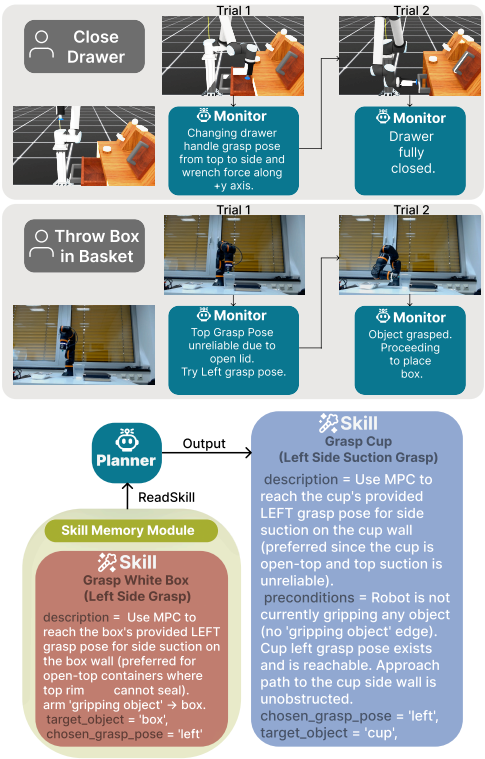

Figure 2: Top: example runs with $T_S = 1$ showing monitor skill exploration. Bottom: real-world example showing zero-shot generalization to a new object (cup) based on the skill learned from interacting with a similar object (open-lid box). Only a summary of the learned skills are shown.

poses, and lack of safety mechanisms. An interesting interpretation is to view MARS through the lens of Hierarchical Reinforcement Learning (HRL) [Sutton and Precup1998]. In this view, MARS presents a Semi-MDP where skill preconditions define initiation sets, the controller defines the intra-option policy, and scene graph checks define termination, making skill generation analogous to an HRL option-level policy. We plan to address the limitations by investigating this connection and other directions such as safe exploration.

References

[Grüne and Pannek, 2011] Lars Grüne and Jürgen Pannek. *Nonlinear Model Predictive Control*, pages 43–66. 04 2011.

[Gu *et al.*, 2023] Qiao Gu, Ali Kuwajerwala, Sacha Morin, Krishna Murthy Jatavallabhula, Bipasha Sen, Aditya Agarwal, Corban Rivera, William Paul, Kirsty Ellis, Ramalingam Chellappa, Chuang Gan, C. M. D. Melo, J. B. Tenenbaum, Antonio Torralba, F. Shkurti, and L. Paull. Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning. *IEEE International Conference on Robotics and Automation*, 2023.

[Huang *et al.*, 2023] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *Conference on Robot Learning*, 2023.

[Khatib, 1987] O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53, 1987.

[Lynch *et al.*, 2022] Corey Lynch, Ayzaan Wahid, Jonathan Tompson, Tianli Ding, James Betker, Robert Baruch, Travis Armstrong, and Pete Florence. Interactive language: Talking to robots in real time, 2022.

[Mees *et al.*, 2022] Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters (RA-L)*, 7(3):7327–7334, 2022.

[Mower *et al.*, 2024] Christopher E. Mower, Yuhui Wan, Hongzhan Yu, Antoine Grosnit, Jonas Gonzalez-Billandon, Matthieu Zimmer, Jinlong Wang, Xinyu Zhang, Yao Zhao, Anbang Zhai, Puze Liu, Davide Tateo, Cesar Cadena, Marco Hutter, Jan Peters, Guangjian Tian, Yuzheng Zhuang, Kun Shao, Xingyue Quan, Jianye Hao, Jun Wang, and Haitham Bou-Ammar. Ros-llm: A ros framework for embodied ai with task feedback and structured reasoning. *arXiv.org*, 2024.

[Patel *et al.*, 2025] Shivansh Patel, Xi Yin, Wenlong Huang, Shubham Garg, H. Nayyeri, Fei-Fei Li, S. Lazebnik, and Yunzhu Li. A real-to-sim-to-real approach to robotic manipulation with vlm-generated iterative keypoint rewards. *arXiv.org*, 2025.

[Qu *et al.*, 2026] Kaixian Qu, Guowei Lan, René Zurbrügg, Changan Chen, Christopher E. Mower, Haitham Bou-Ammar, and Marco Hutter. A pragmatist robot: Learning to plan tasks by experiencing the real world, 2026.

[Salimpour *et al.*, 2025] Sahar Salimpour, Lei Fu, Kajetan Rachwał, Pascal Bertrand, Kevin O'Sullivan, Robert Jakob, Farhad Keramat, Leonardo Militano, Giovanni Toffetti, Harry Edelman, and Jorge Peña Queralta. Towards embodied agentic ai: Review and classification of llm- and vlm-driven robot autonomy and interaction, 2025.

[Sutton and Precup, 1998] Richard Sutton and Doina Precup. Between mdps and semi-mdps: Learning, planning, and representing knowledge at multiple temporal scales. 08 1998.

[Wei *et al.*, 2024] Tianhao Wei, Liqian Ma, Rui Chen, Weiye Zhao, and Changliu Liu. Meta-control: Automatic model-based control synthesis for heterogeneous robot skills. *arXiv.org*, 2024.

[Yu *et al.*, 2023] Wenhao Yu, Nimrod Gileadi, Chuyuan Fu, Sean Kirmani, Kuang-Huei Lee, Montse Gonzalez Arenas, Hao-Tien Lewis Chiang, Tom Erez, Leonard Hasenclever, Jan Humplik, Brian Ichter, Ted Xiao, Peng Xu, Andy Zeng, Tingnan Zhang, Nicolas Heess, Dorsa Sadigh, Jie Tan, Yuval Tassa, and Fei Xia. Language to rewards for robotic skill synthesis, 2023.