EMERGE-POLICY: A ROBOT MIND EMERGES BEYOND A SINGLE POLICY

Anonymous authors
Paper under double-blind review

ABSTRACT

A robot's effective "mind" need not reside in a single policy. It can emerge when specialized components perceive, reason, predict, act, verify, and remember within a shared orchestration process. EMERGE-Policy turns this perspective into a graph-structured agentic framework that coordinates both capability invocation and information exchange. A Main Agent retains task-level state within an active context window, while role-specific Sub Agents process perception, execution monitoring, verification, and memory consolidation in isolated contexts and return structured, task-relevant evidence. Role-specific contexts control information load by exposing only decision-relevant evidence to the Main Agent, while the functional Skill interface composes heterogeneous backends as Operational, Imagination, and Evaluation Skills. Criterion-grounded verification, textual failure diagnosis, and Branch Stack recovery provide localized correction, with token-aware external memory preserving task-relevant state. Together, their closed-loop interaction realizes the system-level policy captured by the name EMERGE-Policy. Without additional fine-tuning, we achieve outstanding performance on several public benchmarks that have had a wide-reaching impact, and conduct a series of real robot experiments. These system-level results suggest that through the division of different functional sub-tasks among multiple agents and their concurrent collaboration, as well as the technical paradigm where the model is regarded as a skill and called within the framework, EMERGE-Policy can extend the robust robot policies beyond isolated runs.

1 INTRODUCTION

A long-standing goal in embodied AI is constructing efficient, human-like intelligent systems. Intelligence is rarely confined to a single policy or reasoning model; instead, it arises from the coordinated interaction of perception, prediction, task reasoning, physical control, and external memory. In robot manipulation, this coordination is paramount: a robot must continuously ground instructions, execute behaviors, verify outcomes, and adjust to execution deviations. Because early-stage errors propagate through subsequent steps, system reliability depends not only on individual component strength, but on how control and information flow across the full pipeline. The central challenge is coordinating these specialized capabilities while managing the information load passed to the high-level planner.

Existing literature offers complementary building blocks toward this goal. Vision-Language-Action (VLA) models have evolved from pretrained vision-language representations (Kim et al., 2024; Brohan et al., 2023) to diffusion- and flow-based action generation (Li et al., 2024; Reuss et al., 2024; Black et al., 2024; Intelligence et al., 2025; Pertsch et al., 2025), as well as dual-system architectures combining deliberative reasoning with reactive control (Bjorck et al., 2025). Meanwhile, video-based pretraining incorporates human and physical priors (Cheang et al., 2024; Shen et al., 2026; Hu et al., 2024), and World-Action Models (WAMs) couple action generation with future-state prediction (Kim et al., 2026; Ye et al., 2026c; Yuan et al., 2026). In parallel, program-based abstraction represents behaviors as executable code over perception/control APIs (Liang et al., 2023; Mu et al., 2024; Fu et al., 2026), with frameworks converting validated repairs into reusable skills (Lu et al., 2026). Recent agentic systems further orchestrate learned policies, controllers, visual tools, and execution memory (Li et al., 2026c; Zhang et al., 2026e; Huang et al., 2026). However, current orchestration often passes unstructured, dense feedback directly to the high-level planner, risking

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

information overload. This motivates isolating intermediate perception, monitoring, and verification into structured, decision-relevant summaries.

We introduce **EMERGE-Policy**, a graph-structured agentic framework where coherent embodied behavior emerges from structured multi-agent collaboration rather than an isolated model. EMERGE-Policy explicitly defines functional roles, information boundaries, capability invocation, and recovery. A *Main Agent* tracks global task progress, decomposes instructions into criterion-grounded subgoals, and steers execution. Context-isolated *Sub Agents* handle specialized tasks (perception, monitoring, verification, and memory consolidation), returning structured evidence to the Main Agent to minimize context pollution while preserving crucial decision-level information.

Action execution in EMERGE-Policy centers on a shared subgoal specification—defining a target entity, intended state change, and completion criterion—serving as a contract across three Skill roles: *Operational Skills* (action execution), *Imagination Skills* (future outcome prediction), and *Evaluation Skills* (criterion and feasibility assessment). This functional abstraction enables modular composition of diverse policy and world-model backends within a single closed loop. Upon subgoal execution, verified success advances the plan; failures trigger a textual diagnosis and push a local recovery goal onto a *Branch Stack*. Global replanning is invoked only when local recovery fails. Token-aware external memory ensures long-horizon context retention. Together, these closed-loop mechanisms yield the system-level intelligence of EMERGE-Policy.

We systematically evaluate EMERGE-Policy across standard and perturbed LIBERO benchmarks, as well as multi-step real-robot manipulation tasks, demonstrating its adaptability across various VLA and WAM backends. Our contributions are threefold:

- We introduce EMERGE-Policy, a graph-structured agentic framework in which robot behavior emerges from the coordination of a Main Agent, role-specific Sub Agents, Skills, and external memory rather than from a single policy. By adopting this approach, we divided the task into several parts. Different agents can independently complete the sub-tasks or sub-functions, thereby solving the overall problem.
- We have expanded the definition of skills. EMERGE-Policy regards the currently widely-discussed models such as VLA, World model, and Verifier model as the skills of the agent. These skills are invoked by the agent at different stages of completing the task. Furthermore, we define explicit information and control interfaces: Sub Agents return only structured, task-relevant evidence, while functional Skills, completion criteria, Branch Stack recovery, and episodic memory connect planning, action, and feedback.
- We conduct thorough and extensive experiments on a series of widely adopted benchmarks. The experiment shows that our framework achieves a significant improvement compared to the most relevant models ($\pi_{0.5}$ and Cosmos Policy). It has increased by 2.0% and 0.7% on LIBERO, by 2.6% and 11.7% on LIBERO-plus, and significantly enhanced the performance of memory on Robodojo, and increased the success rate by 3.59% compared to the baseline, and we deploy it on a physical robot for a series of long-horizon tasks.

2 Related Work

Vision-Language-Action and World Models. VLAs map multi-modal observations to robot controls Brohan et al. (2023); O'Neill et al. (2024); Team et al. (2024); Kim et al. (2024), with recent flow-based models enhancing expressive generation Black et al. (2024); Intelligence et al. (2025). Meanwhile, world models predict latent future states or verify rollouts Zhou et al. (2024); Zhao et al. (2024); Liu et al. (2026c). Rather than retraining these models, EMERGE-Policy provides a unified functional interface that categorizes backends into Operational, Imagination, and Evaluation Skills within a closed-loop protocol.

Agent Engineering and Context Management. Language-based agents generate executable programs or maintain explicit state for long-horizon robotics Singh et al. (2022); Liang et al. (2023); Shinn et al. (2023); Mu et al. (2024); Tan et al. (2024); Yoneda et al. (2024); Fu et al. (2026); Lu et al. (2026). However, active context windows often suffer from redundant processing traces. EMERGE-Policy addresses this by distributing heterogeneous evidence across role-specific Sub Agents with isolated contexts, delivering only decision-critical evidence to the Main Agent.

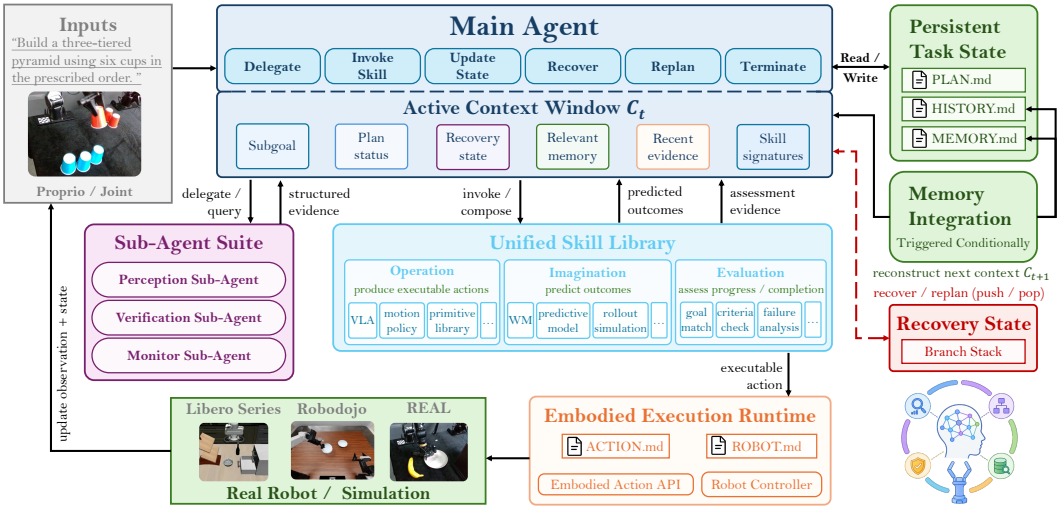


Figure 1: **EMERGE-Policy workflow from instruction to verified execution.** The Main Agent maintains active task context and coordinates role-specific Sub Agents with Operational, Imagination, and Evaluation Skills. Structured evidence and Skill outcomes update task state after each interaction: verified successes advance the plan, while failures enter localized Branch Stack recovery before global replanning. Episode memory preserves relevant task state throughout execution.

Closed-Loop Orchestration and Recovery. Prior hierarchical and closed-loop systems link high-level reasoning with affordances or low-level primitives Ahn et al. (2022); Huang et al. (2022; 2023); Shi et al. (2025); Yang et al. (2025). Recent robot harnesses emphasize runtime execution, spatial memory, and outcome verification Shah et al. (2025); Zhai et al. (2025); Tan et al. (2025a); Huang et al. (2026); Li et al. (2026c); Liu et al. (2026a); Zhang et al. (2026e). EMERGE-Policy complements these architectures by integrating capability invocation, evidence verification, and local recovery into an explicit agentic orchestration graph.

An extended review of related works is provided in Appendix A.1.

3 Method

3.1 System Overview

As illustrated in Figure 1, given a natural language instruction $\mathcal{T}$ and continuous visual observation $\mathcal{O}_t$, EMERGE-Policy organizes robot manipulation as closed-loop coordination among a Main Agent, role-specific Sub Agents, and a unified Skill library. The Main Agent maintains an active task context $\mathcal{C}_t$, decomposes instructions into subgoals with explicit completion criteria, and orchestrates overall execution.

To prevent context bloat caused by raw multimodal streams and intermediate reasoning traces, evidence-intensive operations (e.g., perception, verification, monitoring) are delegated to Sub Agents operating in isolated contexts. Furthermore, heterogeneous backends (e.g., VLAs, world models, motion primitives) are unified under a common functional Skill interface categorized into Operational, Imagination, and Evaluation Skills. The system-level policy emerges from the closed-loop interaction of these roles rather than any single backend.

3.2 Main Agent: Planning, Recovery, and Memory

The Main Agent coordinates task-level planning, capability invocation, outcome verification, and failure recovery. Given instruction $\mathcal{T}$ and evidence derived from observation $\mathcal{O}_t$, it maintains active context $\mathcal{C}_t$ and implements an LLM-based orchestration policy π_{M} over decision space $\mathcal{D}$. At interaction step t, the decision is given by:

$$d_t \sim \pi_{\mathrm{M}}\left(\cdot \mid \mathcal{C}_t\right), \qquad d_t \in \mathcal{D}, \tag{1}$$

where $\mathcal{D}$ comprises decisions to delegate processing to a Sub Agent, invoke a Skill, update task state, initiate recovery, revise plans, or terminate execution (Figure 2).



Figure 2: **Main Agent orchestration and feedback flow.** The Main Agent selects task-level decisions from an active context containing instructions, plan status, recovery state, memory, and compact Skill signatures. It delegates heavy evidence processing to Sub Agents, invokes Skills, and updates or revises the plan based on returned structured evidence.

Hierarchical Context Engineering & Memory Engine. The active context $\mathcal{C}_t$ contains the task instruction, active subgoal, plan status, top Branch Stack frame, task-relevant memory, and compact signatures of available Skills. Raw observations and raw execution histories remain strictly outside $\mathcal{C}_t$. State persistence is maintained across four complementary structures: working memory in $\mathcal{C}_t$, structured subgoal sequences in `PLAN.md`, chronological execution archives in `HISTORY.md`, and consolidated environment facts in `MEMORY.md`. When $\mathcal{C}_t$ reaches token capacity, a *Memory Integration Agent* compresses evicted context into `HISTORY.md` and updates `MEMORY.md` without modifying model weights. All structures are reinitialized per episode. Detailed memory integration algorithms and episode isolation protocols are described in Appendix A.2, Case study and performance in Robodojo-Sim (Chen et al., 2026) are shown in Appendix A.5.

Hierarchical Task Planner and Branch Recovery. The Main Agent decomposes instruction $\mathcal{T}$ into an ordered sequence of subgoals $g_i = \langle \text{Target}_i, \text{Change}_i, \text{Criterion}_i \rangle$. Following physical execution, the observed outcome is evaluated against Criterion_i. If satisfied, execution advances; if a failure occurs, the Main Agent generates a textual diagnosis and pushes a localized recovery subgoal onto a dedicated **Branch Stack**. Stack-top subgoals are addressed first to resolve local discrepancies. If recovery fails repeatedly after a preset threshold, global replanning is triggered in `PLAN.md`. Formal failure diagnosis prompts and recovery stack are elaborated in Appendix A.2.2.

3.3 Sub Agents: Role-Specific Evidence Processing

Sub Agents perform evidence-intensive operations in role-specific, isolated contexts. For a role k, given raw observations $\mathcal{O}^{\text{raw}}_{0:t}$ and active subgoal g_i, the Sub Agent produces structured evidence:

$$\mathcal{Z}_t^{(k)} = f_k\left(\mathcal{O}^{\text{raw}}_{0:t};\, g_i\right), \tag{2}$$

where f_k represents role-specific processing. Sub Agents operate as non-blocking background workers and are categorized into three specializations:

- **Perception Sub Agent**: Extracts task-relevant scene evidence, 3D object poses, and spatial affordances via vision-language and spatial grounding tools. Details are shown in Appendix A.8.
- **Verification Sub Agent**: Evaluates execution preconditions and checks post-execution outcomes against Criterion_i.
- **Monitor Sub Agent**: Inspects live sensor and observation streams during physical interaction to detect state anomalies and execution drifts.

Upon failure detection, the responsible Sub Agent returns a structured report to trigger localized Branch Stack recovery. Detailed Sub-Agent prompting structures, failure report formats, and async execution protocols are provided in Appendix A.3.

3.4 Unified Skill and Tool Orchestration

EMERGE-Policy exposes heterogeneous capabilities through a unified functional Skill interface:

- **Operational Skills**: Produce executable robot actions, implemented by motion primitives, trajectory controllers, or VLA policies (e.g., $\pi_{0.5}$). The coordination between the Main Agent and Execution Skills and effect of task-specific prior knowledge can be seen in Appendix A.7.
- **Imagination Skills**: Predict candidate future states or candidate rollouts using world action models (WAMs) or action-conditional world models (AC-WMs) without physical execution.

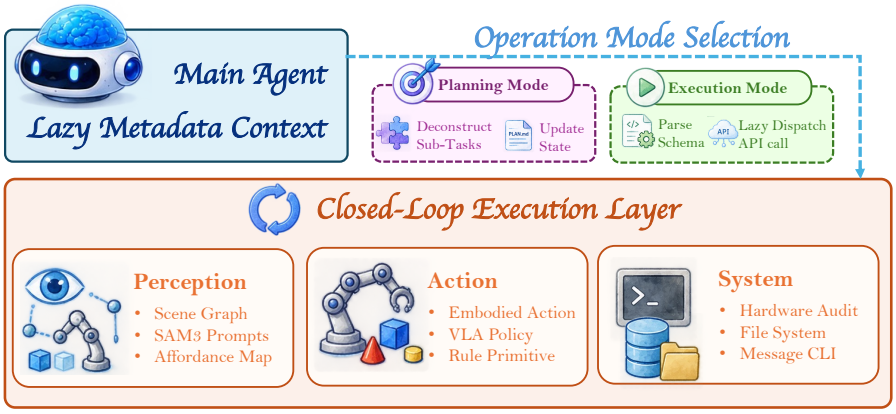


Figure 3: **Tool API hierarchy and standardized interface schema.** EMERGE-Policy organizes interfaces into Perception, Action, and System APIs. Each follows schema $\langle \mathcal{I}_{\text{meta}}, \Theta_{\text{param}}, \Phi_{\text{pre}}, \Psi_{\text{post}} \rangle$, enabling compact metadata retention in $\mathcal{C}_t$ and parameter retrieval upon invocation.

- **Evaluation Skills**: Assess state feasibility, progress, or completion conditions from multimodal evidence. Case study of imagination skill and evaluation skill is shown in Appendix A.6.

Standardized Schema & Execution Lifecycle. As shown in Figure 3, each registered interface K follows schema $K = \langle \mathcal{I}_{\text{meta}}, \Theta_{\text{param}}, \Phi_{\text{pre}}, \Psi_{\text{post}} \rangle$. The Main Agent retains only compact metadata $\mathcal{I}_{\text{meta}}$ in $\mathcal{C}_t$, lazy-loading runtime parameters Θ_{param} upon selection. Skill execution follows a four-phase dual-mode lifecycle: (1) **Planning Phase**, (2) **Perceptual Grounding Phase**, (3) **Physical Dispatch Phase**, and (4) **Outcome Verification Phase**. Complete API specifications, parameter schemas, and hardware dispatch pipelines are detailed in Appendix A.4.

4 Experiments

4.1 Experimental Setup

We evaluate EMERGE-Policy across four benchmarks: LIBERO, LIBERO-Plus, and LIBERO-Pro for tabletop manipulation, and RoboDojo for bimanual manipulation.

Planner and Primitive Configurations. Across all benchmarks, EMERGE-Policy employs the closed-source Codex (powered by `codex-5.6-sol`) as the unified high-level planner and shares a fixed set of action primitives. Environment-specific interaction logic is provided via a benchmark-tailored `AGENTS.md` file without altering the core planner or primitives. For low-level execution in the LIBERO series, we deploy two zero-shot, off-the-shelf policies without additional fine-tuning: a World Action Model (WAM) instantiated via `Cosmos-Policy-LIBERO-Predict2-2B`[1], and a VLA policy instantiated via `pi05_LIBERO`[2]. Functionally, Cosmos-Policy acts as both an Operational Skill (generating action chunks) and an Imagination Skill (predicting future frames); candidate rollouts are scored by an Evaluation Skill, dispatching the top-ranked action greedily.

Perceptual and Horizon Modifications. For the LIBERO series, we add five fixed camera views exclusively to assist high-level perception and spatial reasoning. WAM and VLA retain their original observation interfaces and input spaces without using these extra views. Additionally, to accommodate the multi-stage reasoning overhead of EMERGE-Policy, we moderately extend the per-episode step limit across benchmarks to prevent premature timeout terminations.

4.2 Results on Benchmarks

4.2.1 Baselines

We compare our approach with representative works of three types of methods: End-to-End VLAs and World-Action Models, Code-based and Harness-based Methods, on some mainstream benchmarks.

[1] `https://huggingface.co/nvidia/Cosmos-Policy-LIBERO-Predict2-2B`
[2] `https://github.com/Physical-Intelligence/openpi`

Table 1: **Standard LIBERO results across four task suites.** Success rates (%) are reported for LIBERO-Goal, LIBERO-Spatial, LIBERO-Object, and LIBERO-Long, with their mean. **Bold** and underlined values denote the best and second-best results, respectively. Here, WM denotes the world-model pathway: *w/o WM* uses $\pi_{0.5}$ as the execution backend without this pathway, whereas *w/ WM* enables the Cosmos Policy world-model pathway. For EMERGE-Policy variants, red values in parentheses report absolute gains over the corresponding execution backend.

Method	Goal	Spatial	Object	Long	Average
End-to-end policies and world-action models					
Diffusion Policy (Chi et al., 2025)	68.3	78.3	92.5	50.5	72.4
Octo (Team et al., 2024)	84.6	78.9	85.7	51.1	75.1
MDT (Reuss et al., 2024)	73.5	78.5	87.5	64.8	76.1
OpenVLA (Kim et al., 2024)	79.2	84.7	88.4	53.7	76.5
π_0-FAST (Pertsch et al., 2025)	88.6	96.4	96.8	60.2	85.5
π_0 (Black et al., 2024)	95.8	96.8	98.8	85.2	94.2
OpenVLA-OFT (Kim et al., 2025)	97.9	97.6	98.4	94.5	97.1
$\pi_{0.5}$ (Intelligence et al., 2025)	98.0	98.8	98.2	92.4	96.8
Fast-WAM (Yuan et al., 2026)	97.0	98.2	**100**	95.2	97.6
StarVLA-α (Ye et al., 2026b)	<u>98.6</u>	98.7	99.7	94.2	97.8
AtomicVLA (Zhang et al., 2026c)	97.2	98.8	<u>99.8</u>	96.2	97.8
Cosmos Policy (Kim et al., 2026)	98.2	98.1	**100**	97.6	98.5
Harness-based frameworks					
HarnessVLA (Zhang et al., 2026e)	94.0	97.0	**100**	93.0	96.0
RoboHarness (Huang et al., 2026)	–	–	–	–	98.7
EMERGE-Policy					
EMERGE-Policy (w/o WM)	98.2 **(+0.2)**	**99.2** **(+0.4)**	**100** **(+1.8)**	<u>97.8</u> **(+5.4)**	<u>98.8</u> **(+2.0)**
EMERGE-Policy (w/ WM)	**99.2** **(+1.0)**	<u>99.0</u> **(+0.9)**	**100** **(+0.0)**	**98.6** **(+1.0)**	**99.2** **(+0.7)**

The VLA and WAM baselines are: Baseline methods include Diffusion Policy (Chi et al., 2025), Octo (Team et al., 2024), MDT (Reuss et al., 2024), OpenVLA (Kim et al., 2024), π_0-FAST (Pertsch et al., 2025), π_0 (Black et al., 2024), OpenVLA-OFT (Kim et al., 2025), $\pi_{0.5}$ (Intelligence et al., 2025), Fast-WAM (Yuan et al., 2026), StarVLA-α (Ye et al., 2026b), AtomicVLA (Zhang et al., 2026c) and Cosmos Policy (Kim et al., 2026). For Code-based methods, We mainly compare CaP-X (Fu et al., 2026), RATS (Zhang et al., 2026b) and ASPIRE (Lu et al., 2026) because their performance has been published on the widely adopted benchmark. When it comes to the Harness-based robot policy, we mainly compare it with HarnessVLA (Zhang et al., 2026e) and RoboHarness (Huang et al., 2026).

4.2.2 Results

In this section, we will present the results of our method on the benchmarks. The results in Robodojo-Sim can be seen in Appendix A.5.

On standard LIBERO (Table 1), EMERGE-Policy with the world-model pathway obtains a 99.2% average success rate, compared with 98.5% for its Cosmos Policy execution backend; the largest suite-level difference is on LIBERO-10 (98.6% versus 97.6%). The variant using $\pi_{0.5}$ obtains 98.8%, compared with 96.8% for the underlying policy. Because performance is near the benchmark ceiling, these differences should be interpreted together with trial counts and uncertainty estimates rather than as evidence of a broad breakthrough.

On LIBERO-Plus (Table 2), the world-model variant obtains 93.9%, compared with 82.2% for Cosmos Policy and 93.2% for the reported RoboHarness result. These numbers indicate that the complete EMERGE-Policy system is robust under the evaluated perturbations. They do not, by themselves, identify which architectural component causes the gain, and comparisons with published baselines are controlled only when observation, camera, and interaction budgets match.

On LIBERO-Pro (Table 3), EMERGE-Policy improves substantially over $\pi_{0.5}$ on several implicit-instruction suites, including Spat-S (95.7% versus 20.0%) and L10-S (59.1% versus 8.0%). This result is consistent with the intended benefits of high-level re-grounding and closed-loop orchestration. However, a comparison between the complete system and an end-to-end execution policy does not isolate the contribution of asynchronous cognitive decomposition. A matched ablation must compare

Table 2: **LIBERO-Plus robustness under seven perturbation types.** Success rates (%) are reported for each perturbation, with the mean in Avg. across all perturbations. **Bold** and underlined values denote the best and second-best results, respectively. For each EMERGE-Policy variant, colored values report the absolute change from its corresponding execution backend, $\pi_{0.5}$ or Cosmos Policy: **red** indicates an improvement, and **green** indicates a decrease.

Method	State	Language	Layout	Background	Sensor	Camera	Light	Average
π_0 (Black et al., 2024)	61.0	63.5	76.4	79.0	80.1	61.0	85.0	53.6
$\pi_{0.5}$ (Intelligence et al., 2025)	75.4	85.6	85.7	94.6	89.7	75.4	96.9	85.7
UniVLA (Bu et al., 2025)	46.2	77.6	31.9	81.0	21.2	1.8	69.0	42.9
OpenVLA (Kim et al., 2024)	3.5	23.0	28.5	34.8	15.2	0.8	8.1	15.6
WorldVLA (Cen et al., 2025)	27.9	41.6	38.0	17.1	10.9	0.1	43.7	25.0
RIPT-VLA (Tan et al., 2025b)	31.2	77.6	74.2	91.6	73.5	55.2	88.4	68.4
DreamVLA (Zhang et al., 2026d)	17.6	67.0	43.5	71.5	53.6	26.2	77.5	48.9
ABot-M0 (Yang et al., 2026c)	67.9	86.4	82.6	91.6	86.4	60.4	96.2	80.5
Being-H0.7 (Luo et al., 2026)	–	–	–	–	–	–	–	82.1
OpenVLA-OFT (Kim et al., 2025)	21.7	81.0	68.7	91.0	78.6	55.6	92.7	67.9
X-VLA (Zheng et al., 2026)	89.7	75.7	71.8	96.0	62.7	23.4	88.2	71.4
Cosmos-Policy (Kim et al., 2026)	63.3	81.7	82.2	88.9	92.7	75.8	96.5	82.2
Fast-WAM (Yuan et al., 2026)	44.5	68.9	60.7	53.7	37.7	16.4	78.2	51.5
LingBot-VA (Li et al., 2026b)	83.0	86.4	76.2	53.1	64.4	40.9	82.3	69.5
RoboHarness (Huang et al., 2026)	90.4	97.0	86.8	**97.1**	90.3	**87.6**	97.4	93.2
EMERGE-Policy								
EMERGE-Policy (w/o WM)	87.6 (+12.2)	87.5 (+1.9)	**91.6** (+5.9)	89.7 (-4.9)	84.2 (-5.5)	84.9 (+9.5)	92.4 (-4.5)	88.3 (+2.6)
EMERGE-Policy (w/ WM)	**95.0** (+31.7)	**98.0** (+16.3)	89.0 (+6.8)	97.0 (+8.1)	**94.0** (+1.3)	85.0 (+9.2)	**100** (+3.5)	**93.9** (+11.7)

Table 3: **LIBERO-Pro instruction robustness under explicit and implicit instructions.** Success rates (%) on four task suites under explicit (E) and implicit (S) instructions. **Bold** and underlined values denote the best and second-best results, respectively. Colored values report absolute changes from each method's stated execution baseline, π_{RLinf} for HarnessVLA and $\pi_{0.5}$ for EMERGE-Policy: **red** indicates an improvement, and **green** indicates a decrease.

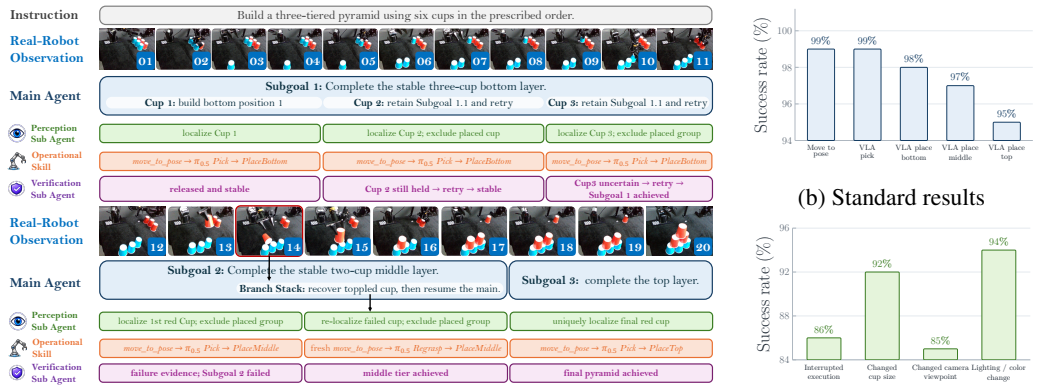
Method	Spat-E	Spat-S	Obj-E	Obj-S	Goal-E	Goal-S	L10-E	L10-S	Total (E/S)
End-to-End VLAs									
$\pi_{0.5}$ (Intelligence et al., 2025)	46.0	20.0	73.0	17.0	46.0	38.0	46.0	8.0	52.8 / 20.8
OpenVLA (Kim et al., 2024)	**89.0**	0.0	0.0	0.0	**98.0**	0.0	**85.0**	0.0	68.0 / 0.0
X-VLA (Zheng et al., 2026)	–	0.0	–	2.0	–	1.0	–	0.0	– / 0.8
π_0 (Black et al., 2024)	60.0	0.0	29.0	0.0	39.0	0.0	27.0	0.0	38.8 / 0.0
Molmoact (Lee et al., 2025)	–	0.0	–	6.0	–	0.0	–	0.0	– / 1.5
π_{RLinf}	–	59.0	–	78.0	–	42.0	–	14.0	– / 48.3
Code-based and harness-based methods									
CaP-X (Fu et al., 2026)	–	12.0	–	22.0	–	26.0	–	–	– / 20.0
RATS (Zhang et al., 2026b)	–	29.0	–	61.0	–	43.0	–	–	– / 44.3
HarnessVLA (Zhang et al., 2026e)	–	69.0 (+10.0)	–	91.0 (+13.0)	–	**66.0** (+24.0)	–	49.0 (+35.0)	– / 68.8 (–/+20.5)
EMERGE-Policy									
EMERGE-Policy	71.4 (+25.4)	**95.7** (+75.7)	**95.5** (+22.5)	**100** (+83.0)	56.5 (+10.5)	56.5 (+18.5)	65.4 (+19.4)	**59.1** (+51.1)	**72.2 / 77.8** (+19.4 / +57.0)

the proposed Sub-Agent workflow with a single Main Agent that receives the same cameras, tools, prompts, and interaction budget.

4.3 Real-World Robot Experiments

To evaluate real-time adaptation and long-horizon physical execution, we deploy EMERGE-Policy on a challenging real-robot benchmark: **Multi-layered Cup Stacking**. The task requires retrieving randomized paper cups, inverting them, and building a 3-tier pyramid (3, 2, and 1 cups per layer). This benchmark stresses three core capabilities: (1) *long-horizon memory and sub-agent coordination* over extended execution chains, (2) *millimeter-level precision* for adjacent cup placement, and (3) *closed-loop recovery* under physical execution noise.

Setup and Closed-Loop Workflow. As illustrated in Figure 4a, execution flows through a closed loop where the Main Agent plans subgoals, Sub-Agents verify visual feedback, and Operational Skills execute low-level primitives. We evaluate system performance across 100 standard trials (Figure 4b) and 120 perturbation trials (30 per setting across 4 categories, Figure 4c).



(a) EMERGE-Policy workflow

(b) Standard results

(c) Disturbance robustness

Figure 4: **Real-robot cup-stacking workflow and empirical evaluation.** (a) Closed-loop architecture illustrating Main Agent task planning, Sub Agent verification, and operational skill execution during multi-tier cup pyramid construction. (b) Stage-wise task completion rates under nominal conditions (100 trials). (c) System robustness across physical interruptions, object scale variations, camera view reductions, and ambient color/lighting shifts (30 trials per setting).

Table 4: **Ablation study on LIBERO-Plus.** Success rates (%) are reported for each perturbation, with the mean in Avg. across all perturbations. **Bold** and underlined values denote the best and second-best results, respectively. Colored values report the absolute change relative to the full EMERGE-Policy baseline: **red** indicates an improvement, and **green** indicates a decrease.

Method	State	Language	Layout	Background	Sensor	Camera	Light	Average
EMERGE-Policy	**95.0**	**98.0**	89.0	**97.0**	**94.0**	**85.0**	**100**	**93.9**
EMERGE-Policy (w/o Eval)	81.6 (-13.4)	87.1 (-10.9)	88.2 (-0.8)	88.2 (-8.8)	82.6 (-11.4)	81.2 (-3.8)	90.6 (-9.4)	85.6 (-8.3)
EMERGE-Policy (w/o Sub)	89.8 (-5.2)	90.2 (-7.8)	88.8 (-0.2)	92.1 (-4.9)	90.7 (-3.3)	82.8 (-2.2)	95.7 (-4.3)	91.2 (-2.7)
EMERGE-Policy (w/o WM)	87.6 (-7.4)	87.5 (-10.5)	**91.6** (+2.6)	89.7 (-7.3)	84.2 (-9.8)	84.9 (-0.1)	92.4 (-7.6)	88.3 (-5.6)

Robustness Analysis. Under nominal conditions, EMERGE-Policy achieves high stage-wise completion rates (Figure 4b). Under environmental perturbations (Figure 4c), the policy displays strong resilience:

- **Camera View Shift** (85% **success):** Reducing camera count/views causes the highest degradation (-9%) due to reduced spatial parallax affecting localization tools.
- **Execution Interruption** (86% **success):** When partially built structures are forcibly dismantled mid-task, Sub-Agent verification enables immediate dynamic replanning.
- **Scale Variation** (92% **success):** Tolerates 5%–15% scale shifts in cup geometry.
- **Visual/Color Shifts** (93%–94% **success):** Ambient lighting and surface color changes exert negligible impact, confirming robust visual task grounding.

4.4 Ablation Study

To quantify the individual and synergistic contributions of key components in EMERGE-Policy, we conduct systematic ablation experiments on LIBERO-Plus. We analyze two distinct configurations: (1) *Single-component removal* to evaluate component-specific necessity (Table 4), and (2) *Sequential cumulative stripping* to reveal multi-agent synergy and emergent capabilities (Figure 5).

Specifically, we evaluate the following variants: (i) **w/o Eval**: removes the Main-Agent's real-time criterion evaluation skill; (ii) **w/o Sub**: disables both Verification and Monitor Sub-Agents; and (iii) **w/o WM**: eliminates the predictive world model's imagination capability.

Individual Component Contributions. As summarized in Table 4, removing the Evaluation skill (**w/o Eval**) induces the most severe performance degradation, dropping the average success rate from 93.9% to 85.6% (a net decrease of 8.3%). The drop is especially acute under volatile state fluctuations, such as State (-13.4%), Sensor (-11.4%), and Language (-10.9%) perturbations. This

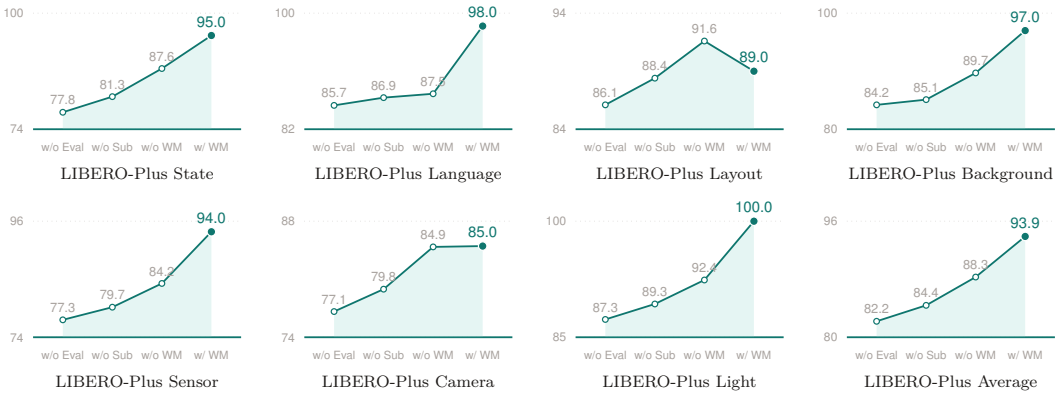


Figure 5: **Cumulative ablation study demonstrating emergent performance.** Sequential stripping of skills and sub-agents illustrates the non-linear degradation across diverse libero-plus perturbation suites.

underlines that real-time subgoal evaluation is paramount for policy stability against perceptual and physical noise.

Disabling world-model-based predictive imagination (**w/o WM**) leads to the second-largest average decline (down to 88.3%, -5.6%), with substantial degradation under Language (-10.5%) and Sensor (-9.8%) shifts. Intriguingly, **w/o WM** exhibits a minor performance gain under Layout shifts ($+2.6\%$), suggesting that unconstrained dynamic imagination can occasionally introduce noise in static spatial layouts. Finally, omitting context-isolated sub-agents (**w/o Sub**) yields a moderate performance reduction (down to 91.2%, -2.7%). While sub-agents primarily ensure baseline stability, their absence notably impacts Language (-7.8%) and State (-5.2%) robustness.

Sequential Stripping and Emergent Synergy. Figure 5 illustrates the cumulative degradation as components are sequentially stripped away. Across individual perturbation modalities—including State, Language, Sensor, and Camera—performance exhibits a monotonic collapse as the framework degrades toward bare-bones variants.

On the aggregate metric, success rates steadily decay from 93.9% (Full Framework) $\rightarrow$ 88.3% (**w/o WM**) $\rightarrow$ 84.4% (**w/o WM + Sub**) $\rightarrow$ 82.2% (**w/o WM + Sub + Eval**). Crucially, under severe compounded shifts (e.g., State dropping to 77.8% and Sensor to 77.3%), performance collapses non-linearly. These results confirm that robust embodied execution does not stem from any isolated module; rather, it organically emerges from closed-loop multi-agent orchestration among hierarchical planning, sub-agent verification, world-model imagination, and criterion-grounded evaluation.

5 Conclusion and Future Work

We present EMERGE-Policy, a graph-structured agentic policy framework in which robot behavior emerges from the coordinated operation of perception, prediction, execution, verification, recovery, and memory. In this sense, EMERGE-Policy is not another low-level control policy; it provides the structure through which interactions among heterogeneous components become coherent system-level behavior. The Main Agent maintains compact task-level state and orchestrates a role-based skill interface, while isolated Sub Agents process dense perceptual and execution evidence. Operational, Imagination, and Evaluation Skills provide a common abstraction for action-producing policies, future-predictive models, and outcome assessment, and verified failures are routed through localized Branch Stack recovery before global replanning. Future work should make the harness evaluation protocol reproducible, quantify accuracy–latency–token trade-offs, and study how the same fixed orchestration graph transfers across planners, low-level policies, tasks, and embodiments. A particularly important direction is to distinguish safe within-episode adaptation from persistent cross-episode learning, enabling experience reuse without contaminating benchmark evaluation.

AI USE STATEMENT

In this work, we used generative AI tools for language editing, proofreading to improve text clarity, and debugging research code. We have not used generative AI tools for core research idea generation, experimental design, or data synthesis, and automated deep literature synthesis without manual verification is not applicable to this work. Additionally, we used generative AI tools for refining prose and formatting figures. We have reviewed all AI-assisted work. Specifically, LLM-generated code was verified, executed, and tested for correctness by the authors, and all suggested textual revisions were manually checked to ensure technical accuracy and consistency with our original research intent. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. Univla: Learning to act anywhere with task-centric latent actions. *arXiv preprint arXiv:2505.06111*, 2025.

Jisong Cai, Long Ling, Shiwei Chu, Zhongshan Liu, Jiayue Kang, Zhixuan Liang, Wenjie Xu, Yinan Mao, Weinan Zhang, Xiaokang Yang, et al. Aha-wam: Asynchronous horizon-adaptive world-action modeling with observation-guided context routing. *arXiv preprint arXiv:2606.09811*, 2026a.

Rui Cai, Jun Guo, Xinze He, Piaopiao Jin, Jie Li, Bingxuan Lin, Futeng Liu, Wei Liu, Fei Ma, Kun Ma, et al. Xiaomi-robotics-0: An open-sourced vision-language-action model with real-time execution. *arXiv preprint arXiv:2602.12684*, 2026b.

Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, et al. Worldvla: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*, 2025.

Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, et al. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024.

Tianxing Chen, Yue Chen, Zixuan Li, et al. Robodojo: A unified sim-and-real benchmark for comprehensive evaluation of generalist robot manipulation policies. *arXiv preprint arXiv:2607.04434*, 2026.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

Letian Fu, Justin Yu, Karim El-Refai, Ethan Kou, Haoru Xue, Huang Huang, Wenli Xiao, Guanzhi Wang, Dantong Niu, Fei-Fei Li, et al. Cap-x: A framework for benchmarking and improving coding agents for robot manipulation. *arXiv preprint arXiv:2603.22435*, 2026.

Jun Guo, Qiwei Li, Peiyan Li, Zilong Chen, Nan Sun, Yifei Su, Heyun Wang, Yuan Zhang, Xinghang Li, and Huaping Liu. Unified 4d world action modeling from video priors with asynchronous denoising. *arXiv preprint arXiv:2604.26694*, 2026.

Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations. *arXiv preprint arXiv:2412.14803*, 2024.

Jinbang Huang, Yuanzhao Hu, Zhiyuan Li, Ran Qi, Yixin Xiao, Zhanguang Zhang, Mark Coates, Tongtong Cao, and Yingxue Zhang. Roboharness: Memory-driven orchestration of heterogeneous robot policies for long-horizon planning. *arXiv preprint arXiv:2607.18060*, 2026.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022.

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. *arXiv preprint arXiv:2307.05973*, 2023.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Tao Jiang, Tianyuan Yuan, Yicheng Liu, Chenhao Lu, Jianning Cui, Xiao Liu, Shuiqi Cheng, Jiyang Gao, Huazhe Xu, and Hang Zhao. Galaxea open-world dataset and g0 dual-system vla model. *arXiv preprint arXiv:2509.00576*, 2025.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

Moo Jin Kim, Yihuai Gao, Tsung-Yi Lin, Yen-Chen Lin, Yunhao Ge, Grace Lam, Percy Liang, Shuran Song, Ming-Yu Liu, Chelsea Finn, et al. Cosmos policy: Fine-tuning video models for visuomotor control and planning. *arXiv preprint arXiv:2601.16163*, 2026.

Jason Lee, Jiafei Duan, Haoquan Fang, Yuquan Deng, Shuo Liu, Boyang Li, Bohan Fang, Jieyu Zhang, Yi Ru Wang, Sangho Lee, et al. Molmoact: Action reasoning models that can reason in space. *arXiv preprint arXiv:2508.07917*, 2025.

Fuhao Li, Wenxuan Song, Han Zhao, Jingbo Wang, Pengxiang Ding, Donglin Wang, Long Zeng, and Haoang Li. Spatial forcing: Implicit spatial representation alignment for vision-language-action model. In *International Conference on Learning Representations*, volume 2026, pp. 132324–132345, 2026a.

Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Fei Han, Mingrui Yu, Zelin Gao, Nan Xue, Xing Zhu, et al. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026b.

Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, et al. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024.

Ruiying Li, Yunlang Zhou, YuYao Zhu, Kylin Chen, Jingyuan Wang, Sukai Wang, Kongtao Hu, Minhui Yu, Bowen Jiang, Zhan Su, et al. Roboclaw: An agentic framework for scalable long-horizon robotic tasks. *arXiv preprint arXiv:2603.11558*, 2026c.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International conference on robotics and automation (ICRA)*, pp. 9493–9500. IEEE, 2023.

Yang Liu, Weixing Chen, Xinshuai Song, Tao Pu, Siwen Mo, Yongjie Bai, Zihao Chen, Qianran Sun, Liruo Zhong, Ying Shen, et al. Phyagentos: A self-evolving operating system for embodied agents with decoupled cognitive planning and physical execution. *arXiv preprint arXiv:2607.16636*, 2026a.

Yicheng Liu, Zibin Dong, Baijun Ye, Tianyuan Yuan, Tao Jiang, et al. G0.5: One autoregressive stream for robot reasoning and action. *arXiv preprint arXiv:2608.11739*, 2026b.

Yuejiang Liu, Fan Feng, Lingjing Kong, Weifeng Lu, Jinzhou Tang, Kun Zhang, Kevin Murphy, Chelsea Finn, and Yilun Du. World action verifier: Self-improving world models via forward-inverse asymmetry. *arXiv preprint arXiv:2604.01985*, 2026c.

Runyu Lu, Yubo Wu, Ethan Kou, Letian Fu, Wenli Xiao, Ajay Mandlekar, Yinzhen Xu, Guanya Shi, Ken Goldberg, Ang Chen, et al. Aspire: Agentic/skills discovery for robotics. *arXiv preprint arXiv:2607.00272*, 2026.

Hao Luo, Wanpeng Zhang, Yicheng Feng, Sipeng Zheng, Haiweng Xu, Chaoyi Xu, Ziheng Xi, Yuhui Fu, and Zongqing Lu. Being-h0. 7: A latent world-action model from egocentric videos. *arXiv preprint arXiv:2605.00078*, 2026.

Haoxiang Ma, Junhao Cai, Xiaoxu Xu, Hao Li, Yuyin Yang, Yang Tian, Jiafei Cao, Hongrui Zhu, Zherui Qiu, Yuqiang Yang, et al. Internvla-a1. 5: Unifying understanding, latent foresight, and action for compositional generalization. *arXiv preprint arXiv:2607.04988*, 2026.

Yao Mu, Junting Chen, Qinglong Zhang, Shoufa Chen, Qiaojun Yu, Chongjian Ge, Runjian Chen, Zhixuan Liang, Mengkang Hu, Chaofan Tao, et al. Robocodex: Multimodal code generation for robotic behavior synthesis. *arXiv preprint arXiv:2402.16117*, 2024.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.

Moritz Reuss, Ömer Erdinç Yağmurlu, Fabian Wenzel, and Rudolf Lioutikov. Multimodal diffusion transformer: Learning versatile behavior from multimodal goals. *arXiv preprint arXiv:2407.05996*, 2024.

Ameesh Shah, William Chen, Adwait Godbole, Federico Mora, Sanjit A Seshia, and Sergey Levine. Learning affordances at inference-time for vision-language-action models. *arXiv preprint arXiv:2510.19752*, 2025.

Yichao Shen, Fangyun Wei, Zhiying Du, Yaobo Liang, Yan Lu, Jiaolong Yang, Nanning Zheng, and Baining Guo. Videovla: Video generators can be generalizable robot manipulators. *Advances in neural information processing systems*, 38:95597–95621, 2026.

Lucy Xiaoyang Shi, Brian Ichter, Michael Equi, Liyiming Ke, Karl Pertsch, Quan Vuong, James Tanner, Anna Walling, Haohuan Wang, Niccolo Fusai, et al. Hi robot: Open-ended instruction following with hierarchical vision-language-action models. *arXiv preprint arXiv:2502.19417*, 2025.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. Progprompt: Generating situated robot task plans using large language models. *arXiv preprint arXiv:2209.11302*, 2022.

Huajie Tan, Xiaoshuai Hao, Cheng Chi, Minglan Lin, Yaoxu Lyu, Mingyu Cao, Dong Liang, Zhuo Chen, Mengsi Lyu, Cheng Peng, et al. Roboos: A hierarchical embodied framework for cross-embodiment and multi-agent collaboration. *arXiv preprint arXiv:2505.03673*, 2025a.

Shuhan Tan, Kairan Dou, Yue Zhao, and Philipp Krähenbühl. Interactive post-training for vision-language-action models. *arXiv preprint arXiv:2505.17016*, 2025b.

Weihao Tan, Wentao Zhang, Xinrun Xu, Haochong Xia, Ziluo Ding, Boyu Li, Bohan Zhou, Junpeng Yue, Jiechuan Jiang, Yewen Li, et al. Cradle: Empowering foundation agents towards general computer control. *arXiv preprint arXiv:2403.03186*, 2024.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

Xiaomi Robotics Team, Jun Guo, Piaopiao Jin, Jason Li, Peiyan Li, Yingyan Li, Futeng Liu, Wanli Peng, Optimus Qin, Yifei Su, et al. Xiaomi-robotics-1: Scaling vision-language-action models with over 100k hours of real-world trajectories. *arXiv preprint arXiv:2607.15330*, 2026.

Wei Wu, Fan Lu, Yunnan Wang, Shuai Yang, Shi Liu, Fangjing Wang, Qian Zhu, He Sun, Yong Wang, Shuailei Ma, et al. A pragmatic vla foundation model. *arXiv preprint arXiv:2601.18692*, 2026.

Ganlin Yang, Zhangzheng Tu, Yuqiang Yang, Sitong Mao, Junyi Dong, Tianxing Chen, Jiaqi Peng, Jing Xiong, Jiafei Cao, Jifeng Dai, et al. Eventvla: Event-driven visual evidence memory for long-horizon vision-language-action policies. *arXiv preprint arXiv:2606.20092*, 2026a.

Senqiao Yang, Jinhui Ye, Pengguang Chen, and Shu Liu. Vlact qwenoft qwen3-vl-4b for robodojo. Hugging Face model card, 2026b. URL `https://huggingface.co/JasonYang66/VLAct-Qwen3vl4b-OFT-RoboDojo`.

Yandan Yang, Shuang Zeng, Tong Lin, Xinyuan Chang, Dekang Qi, Junjin Xiao, Haoyun Liu, Ronghan Chen, Yuzhi Chen, Dongjie Huo, et al. Abot-m0: Vla foundation model for robotic manipulation with action manifold learning. *arXiv preprint arXiv:2602.11236*, 2026c.

Zhejian Yang, Yongchao Chen, Xueyang Zhou, Jiangyue Yan, Dingjie Song, Yinuo Liu, Yuting Li, Yu Zhang, Pan Zhou, Hechang Chen, et al. Agentic robot: A brain-inspired framework for vision-language-action models in embodied agents. *arXiv preprint arXiv:2505.23450*, 2025.

Angen Ye, Boyuan Wang, Chaojun Ni, Guan Huang, Guosheng Zhao, Hao Li, Hengtao Li, Jie Li, Jindi Lv, Jingyu Liu, et al. Gigaworld-policy: An efficient action-centered world–action model. *arXiv preprint arXiv:2603.17240*, 2026a.

Jinhui Ye, Ning Gao, Senqiao Yang, Jinliang Zheng, Zixuan Wang, Yuxin Chen, Pengguang Chen, Yilun Chen, Shu Liu, and Jiaya Jia. Starvla-α: Reducing complexity in vision-language-action systems. *arXiv preprint arXiv:2604.11757*, 2026b.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, et al. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026c.

Takuma Yoneda, Jiading Fang, Peng Li, Huanyu Zhang, Tianchong Jiang, Shengjie Lin, Ben Picker, David Yunis, Hongyuan Mei, and Matthew R Walter. Statler: State-maintaining language models for embodied reasoning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 15083–15091. IEEE, 2024.

Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-wam: Do world action models need test-time future imagination? *arXiv preprint arXiv:2603.16666*, 2026.

Shaopeng Zhai, Qi Zhang, Tianyi Zhang, Fuxian Huang, Haoran Zhang, Ming Zhou, Shengzhe Zhang, Litao Liu, Sixu Lin, and Jiangmiao Pang. A vision-language-action-critic model for robotic real-world reinforcement learning. *arXiv preprint arXiv:2509.15937*, 2025.

He Zhang, Lingzhu Xiang, Haitao Lin, Zeyu Huang, Minghui Wang, Dingyan Zhong, Yubo Dong, Yihao Wu, Yongming Rao, Dongsheng Zhang, et al. Hy-embodied-0.5-vla: From vision-language-action models to a real-world robot learning stack. *arXiv preprint arXiv:2606.14409*, 2026a.

Junyi Zhang, Jiaxin Ge, Hanjun Yoo, Letian Fu, Zihan Yang, Yaowei Liu, Raj Saravanan, Shaofeng Yin, Justin Yu, Dantong Niu, et al. Playful agentic robot learning. *arXiv preprint arXiv:2606.19419*, 2026b.

Likui Zhang, Tao Tang, Zhihao Zhan, Xiuwei Chen, Zisheng Chen, Jianhua Han, Jiangtong Zhu, Pei Xu, Hang Xu, Hefeng Wu, et al. Atomicvla: Unlocking the potential of atomic skill learning in robots. *arXiv preprint arXiv:2603.07648*, 2026c.

Wenyao Zhang, Hongsi Liu, Zekun Qi, Yunnan Wang, Xinqiang Yu, Jiazhao Zhang, Runpei Dong, Jiawei He, He Wang, Zhizheng Zhang, et al. Dreamvla: a vision-language-action model dreamed with comprehensive world knowledge. *Advances in Neural Information Processing Systems*, 38: 24195–24228, 2026d.

Yixian Zhang, Huanming Zhang, Feng Gao, Xiao Li, Zhihao Liu, Chunyang Zhu, Jiaxing Qiu, Yuchen Yan, Jiyuan Liu, Wenhao Tang, et al. Harness vla: Steering frozen vlas into reliable manipulation primitives via memory-guided agents. *arXiv preprint arXiv:2607.08448*, 2026e.

Wentao Zhao, Jiaming Chen, Ziyu Meng, Donghui Mao, Ran Song, and Wei Zhang. Vlmpc: Vision-language model predictive control for robotic manipulation. *arXiv preprint arXiv:2407.09829*, 2024.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, et al. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. In *International Conference on Learning Representations*, volume 2026, pp. 60580–60606, 2026.

Siyuan Zhou, Yilun Du, Jiaben Chen, Yandong Li, Dit-Yan Yeung, and Chuang Gan. Robodreamer: Learning compositional world models for robot imagination. *arXiv preprint arXiv:2404.12377*, 2024.

A Appendix

A.1 Extended Related Work

This section provides an expanded discussion of the literature briefly summarized in Section 2.

A.1.1 Detailed Analysis of VLAs and World Models

Vision-Language-Action (VLA) models such as RT-2 Brohan et al. (2023), Open X-Embodiment O'Neill et al. (2024), Octo Team et al. (2024), and OpenVLA Kim et al. (2024) transfer large-scale multi-modal pretraining directly to visuomotor control. Recent advancements like π_0 Black et al. (2024) and $\pi_{0.5}$ Intelligence et al. (2025) leverage flow matching to model expressively continuous action distributions. While these models excel at low-level trajectory generation, they lack explicit mechanisms for high-level task decomposition, outcome verification, and persistent state management. On the prediction side, world models like RoboDreamer Zhou et al. (2024), VLMPC Zhao et al. (2024), and World Action Verifier Liu et al. (2026c) predict future observations or assess policy rollouts. As modern models blur the line between action generation and prediction, EMERGE-Policy formalizes a functional interaction layer: backends are wrapped into Operational, Imagination, or Evaluation Skills, allowing heterogeneous VLAs and world models to collaborate seamlessly without architectural refactoring.

A.1.2 Agent Harnesses, Context Isolation, and Memory

LLM-based robot code generation (e.g., Code as Policies Liang et al. (2023), ProgPrompt Singh et al. (2022), RoboCodeX Mu et al. (2024)) and interface studies like CaP-X Fu et al. (2026) demonstrate that prompt design, perceptual abstraction, and tool grounding govern agent reliability. In parallel, frameworks such as Reflexion Shinn et al. (2023), Statler Yoneda et al. (2024), Cradle Tan et al. (2024), and ASPIRE Lu et al. (2026) introduce explicit world states, textual reflections, and self-repairing skill repositories for continuous operations. However, loading dense visual traces and intermediate execution histories into a single LLM context degrades planning performance. EMERGE-Policy solves this system bottleneck by offloading visual perception, state monitoring, and subtask verification to dedicated Sub Agents. By insulating the Main Agent's context window from raw perception traces, EMERGE-Policy achieves efficient context management without losing critical execution evidence.

A.1.3 Hierarchical Execution and Runtime Recovery Mechanisms

Classical language-guided frameworks like SayCan Ahn et al. (2022), Inner Monologue Huang et al. (2022), VoxPoser Huang et al. (2023), Hi Robot Shi et al. (2025), and Agentic Robot Yang et al. (2025) establish the fundamental paradigm of decoupling high-level deliberative planning from grounded execution. Modern orchestration harnesses such as RoboClaw Li et al. (2026c), HarnessVLA Zhang et al. (2026e), RoboHarness Huang et al. (2026), and RoboOS Tan et al. (2025a) focus on coordinating heterogeneous primitives, spatial memory, and cross-embodiment skills. Concurrently, supervision and verifier models like LITEN Shah et al. (2025), VLAC Zhai et al. (2025), and PhyAgentOS Liu et al. (2026a) provide runtime feedback and progress assessment against physical acceptance criteria. EMERGE-Policy builds upon these concepts by uniting capability invocation, context-isolated verification, and a localized recovery mechanism using a Branch Stack. When a subgoal violation occurs, the Main Agent performs localized sub-tree recovery before resorting to full-plan replanning, creating a robust, system-level execution loop.

A.2 Main Agent Architecture and Context Engineering

A.2.1 Detailed State Structures and Memory Consolidation

As introduced in Section 3.2, state persistent tracking relies on four distinct components:

1. **Active Context Window** ($\mathcal{C}_t$): Serves as working memory. It maintains system prompts, active subgoal g_i, top Branch Stack frame, and single-line Skill metadata $\mathcal{I}_{\text{meta}}$.

2. **`PLAN.md`**: Structured Markdown file recording global subgoals $\langle \text{Target}_i, \text{Change}_i, \text{Criterion}_i \rangle$ and execution status flags.
3. **`HISTORY.md`**: Append-only chronological archive storing summarized observations, executed actions, and Sub-Agent evidence summaries.
4. **`MEMORY.md`**: Consolidated belief state containing dynamic spatial facts (e.g., `drawer_1_state: open`). Stale facts are overwritten when contradictory evidence arrives.

When $\mathcal{C}_t$ reaches token limits, the *Memory Integration Agent* executes a compression pass:

$$\mathcal{C}_{t+1} = \text{Compress}\left(\mathcal{C}_t, \texttt{PLAN.md}, \texttt{MEMORY.md}, \text{BranchStack}\right)$$

This operation modifies external textual representations without updating underlying model weights. All memory structures are reinitialized prior to every evaluation episode.

A.2.2 Failure Diagnosis and Localized Branch Recovery

When verification or monitoring identifies a failure, a structured failure packet $\mathcal{E}_{\text{fail}}$ is constructed:

$$\mathcal{E}_{\text{fail}} = \langle \text{SubgoalID}, \text{FailureType}, \text{ObservedVisualState}, \text{InferredCause} \rangle$$

The Main Agent ingests $\mathcal{E}_{\text{fail}}$ to synthesize a verbal reflection diagnosis:

$$\text{Diagnosis} = \text{LLM}_{\text{Reflect}}\left(\mathcal{C}_t, \mathcal{E}_{\text{fail}}\right)$$

Based on Diagnosis, a temporary localized recovery subgoal g_{rec} is pushed onto the Branch Stack. The execution loop pops g_{rec} and attempts resolution. If g_{rec} fails continuously for $N = 2$ attempts, the Branch Stack is cleared, and high-level replanning is triggered in `PLAN.md`.

A.3 Sub Agents: Context Isolation and Execution Protocols

As noted in Section 3.3, Sub Agents run as non-blocking background workers. Each instance receives a minimal prompt containing active subgoal g_i, task-relevant bounding boxes, and authorized tool handles.

A.3.1 Specialized Sub-Agent Workflows

- **Perception Sub Agent**: Queries 3D spatial APIs and segment models (e.g., SAM3) to output target affordances $\mathcal{Z}_t^{(\text{perc})} = \{\mathbf{p}_{\text{target}}, \mathbf{R}_{\text{orient}}, \mathcal{M}_{\text{mask}}\}$.
- **Verification Sub Agent**: Compares post-execution visual streams with Criterion_i, outputting binary verdict $\mathcal{V} \in \{\text{Pass}, \text{Fail}\}$ and visual delta logs.
- **Monitor Sub Agent**: Processes streaming visual observations during continuous primitive execution, returning anomaly warnings $\mathcal{Z}_t^{(\text{mon})}$ upon contact slip or trajectory deviation.

A.4 Tool API Schemas and Interface Specifications

Extending Section 3.4, all callable interfaces $K = \langle \mathcal{I}_{\text{meta}}, \Theta_{\text{param}}, \Phi_{\text{pre}}, \Psi_{\text{post}} \rangle$ are organized into three primary categories:

A.4.1 Workspace and System Execution APIs

- **Agent Mode API**: Controls system execution state (`PLANNING` $\leftrightarrow$ `EXECUTION`).
- **File System API**: Grants structured read/write operations over `PLAN.md`, `HISTORY.md`, and `MEMORY.md`.
- **Shell & Message APIs**: Handles low-level hardware pinging, emergency stops, and user interface feedback.

A.4.2 Perceptual Sub-Agent APIs

- **Scene Graph API**: Constructs real-time dynamic semantic graphs containing 3D bounding boxes and topological relations.
- **Vision-Language Segmentation API**: Generates bounding box prompts and visual segmentations via SAM3.

A.4.3 Embodied Action APIs

- **Primitive Motion Controller**: Dispatches analytical IK trajectories, linear pushing primitives, and pick-and-place routines.
- **VLA Policy Interface**: Interfaces with fine-tuned VLA policies (e.g., $\pi_{0.5}$, OpenVLA) for closed-loop continuous motor control.

A.5 Memory and Recovery

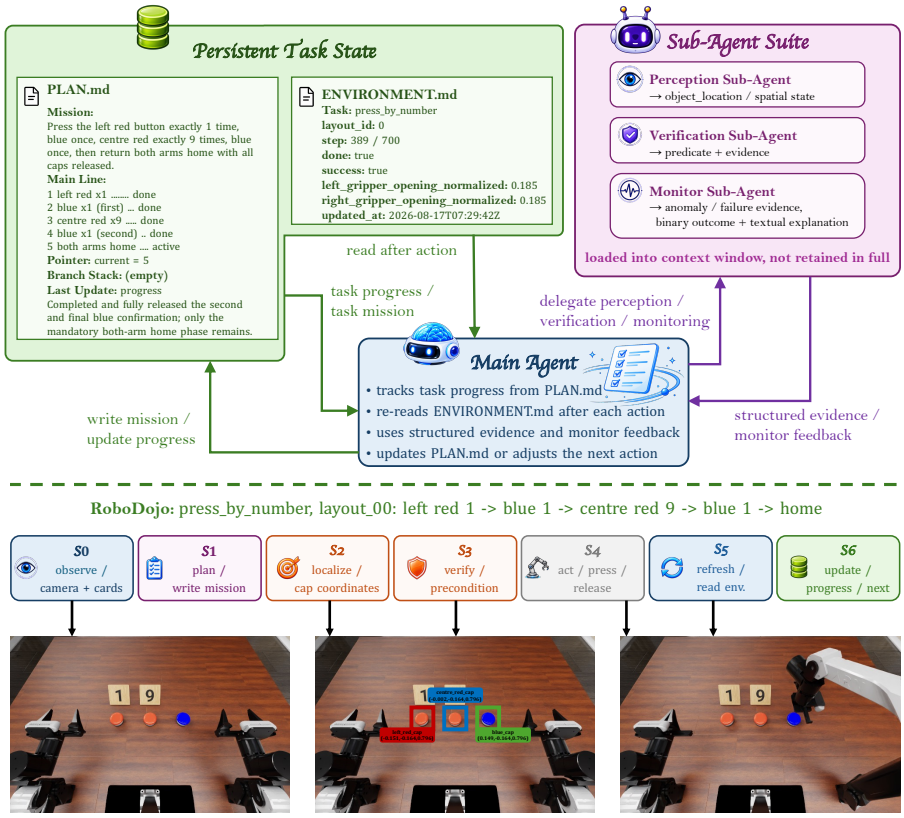


Figure 6: **Three-tier memory and context management.** `PLAN.md` and `ENVIRONMENT.md` maintain episode task state and current environment feedback. Specialized Sub Agents convert dense observations into structured evidence for the Main Agent, which consolidates relevant facts and updates execution using asynchronous visual verification.

The memory architecture of EMERGE-Policy is not a monolithic neural state but a multi-layer, externalised framework that decouples task progression from transient working memory, as illustrated in Figure 6. At its foundation, the agent maintains two persistent files that serve as the ground truth for long-horizon execution. `PLAN.md` stores the complete subgoal decomposition along with the completion status of each step, and is updated through two distinct modes – *new mission* to initialise the overall objective and progress to incrementally record milestones such as finishing a press cycle or advancing to the confirmation phase. Complementing this, `ENVIRONMENT.md` is automatically refreshed by the environment watchdog after every physical action; it contains simulation-level facts including joint states, step counters, and success flags. The agent mandatorily re-reads this file after

each *execute robot action* call, ensuring that every subsequent decision is grounded in the latest physical snapshot rather than in stale context-window contents. Together, these external state files provide a durable and inspectable record that enables safe recovery from interruptions and prevents the accumulation of outdated information within the limited token budget of the language model.

Above this persistent layer, the agent leverages structured evidence returned by specialised sub-agents for perception, planning, and verification. Sub-agents such as object location, task planning, and task verification produce compact JSON payloads containing explicitly labelled fields—for instance, spatial coordinates, predicate truth values with supporting evidence, or a step-wise action list. These outputs are consumed by the main agent as short-term working memory within its active context, but they are never retained in their full raw form. Instead, they are semantically distilled to retain only the decision-relevant facts, such as the exact 3D position of each button cap or the binary outcome of a pre-condition check. This design avoids cluttering the reasoning window with voluminous sensor traces or interaction histories, while still providing the precise, machine-readable evidence required for subsequent planning and verification steps.

The third layer consists of a closed-loop visual verification process that actively corrects the agent's remembered state. A dedicated visual monitor thread asynchronously evaluates the scene after each action, comparing observed visual cues against the expected subgoal criterion (e.g., whether the gripper is clear of the button). The monitor returns a binary judgement along with a textual explanation, which the main agent immediately incorporates into its updated belief. This feedback loop is critical because it catches discrepancies that may not be reflected in the motion execution result alone—for example, even when a recover command reports success, the visual monitor may indicate that the arm is still obstructing the view of the button, prompting further corrective actions. Consequently, the memory state is dynamically refined by perceptual evidence, rather than relying solely on proprioceptive or command-level feedback.

Overall, this three-tiered memory mechanism—persistent files for task and environment state, structured sub-agent evidence for concise contextual facts, and asynchronous visual verification for error correction—collectively ensures that the main agent operates within a bounded context window while maintaining reliable awareness of progress and physical conditions. The separation of roles not only enhances scalability for long-horizon tasks but also introduces transparency and auditability, as every major state transition and corrective decision is recorded in the external files or in the structured outputs of the sub-agents. This architectural choice underpins the agent's robustness in perturbed scenarios and its ability to resume or recover from failures without relying on an unbounded internal memory, thus aligning with the system-level orchestration philosophy of EMERGE-Policy. Through these three memory mechanisms, we achieved the best performance in the memory and open settings of Robodojo-Sim, shown in Table 5, indicating that it possesses strong historical information encoding and non-Markovian reasoning capabilities, and it performed exceptionally well in skill reorganization and the alignment of open vocabulary semantics to actions. It can remember the key visual information that has been obscured or disappeared and use it for subsequent decisions. This three-tiered memory mechanism is also visually reflected in Figure 6, where the agent repeatedly integrates persistent files, structured sub-agent evidence, and visual verification feedback throughout execution.

A.6 Imagination and Evaluation Skills

The functional distinction between Operational, Imagination, and Evaluation Skills (Section **??**) enables a principled mechanism for action selection under uncertainty. In *EMERGE-Policy*, the Imagination Skill generates a set of candidate action chunks from the current observation and task context, while the Evaluation Skill scores each candidate's predicted outcome against the active subgoal criterion. These two skills were implemented in accordance with the implementation method specified in the Cosmos Policy. The Main Agent then dispatches the highest-scoring candidate for physical execution, as illustrated in Figure 7. This closed-loop selection is performed *before* committing to a physical rollout, thereby reducing the risk of irreversible failures due to perceptual ambiguity or planning errors.

Concretely, given a current observation o_t and subgoal g_i, the Imagination Skill produces M candidate action sequences $\{\mathbf{a}^{(m)}_{t:t+H}\}_{m=1}^{M}$, each of horizon H. For each candidate, the Imagination Skill also predicts the corresponding future observation sequence $\{\hat{o}^{(m)}_{t+1:t+H}\}$, or a latent representation thereof.

Table 5: **RoboDojo simulation results across five capability dimensions.** Each entry reports the score and success rate (%) for five capability dimensions and their average. Published baseline and human-expert results are reproduced from RoboDojo (Chen et al., 2026) and its public leaderboard; EMERGE-Policy is evaluated using the same protocol. An em dash indicates that a result is not reported.

Method	Generalization	Memory	Precision	Long-Horizon	Open	Average
GalaxeaVLA (G0.5) (Liu et al., 2026b)	18.95 / 13.00%	8.61 / 7.33%	28.25 / 20.42%	44.12 / 32.25%	1.73 / 1.58%	20.23 / 14.88%
Xiaomi-Robotics-1 (Team et al., 2026)	23.55 / 17.00%	7.81 / 6.56%	26.69 / 18.83%	38.39 / 23.67%	3.94 / 3.58%	20.07 / 13.93%
EMERGE-Policy	15.37 / 9.17%	**25.00 / 22.51%**	12.40 / 7.38%	29.54 / 17.42%	**19.98 / 11.20%**	20.46 / 13.54%
Hy-Embodied-0.5-VLA (Zhang et al., 2026a)	11.78 / 8.50%	13.37 / 12.11%	13.81 / 8.00%	25.74 / 14.92%	0.65 / 0.58%	13.07 / 8.80%
Spatial Forcing (Li et al., 2026a)	14.12 / 9.50%	5.43 / 4.11%	17.33 / 10.58%	23.26 / 14.58%	1.78 / 1.58%	12.38 / 8.04%
$\pi_{0.5}$ (Intelligence et al., 2025)	13.38 / 8.00%	5.78 / 4.56%	12.40 / 5.50%	23.54 / 14.67%	1.98 / 1.67%	11.41 / 6.91%
InternVLA-A1.5 (Ma et al., 2026)	10.36 / 7.00%	4.93 / 3.56%	15.23 / 10.17%	23.80 / 13.75%	1.43 / 1.42%	11.15 / 7.14%
VLAct (Yang et al., 2026b)	9.54 / 6.50%	0.66 / 0.56%	20.62 / 15.25%	20.12 / 13.67%	2.37 / 2.25%	10.66 / 7.60%
X-VLA (Zheng et al., 2026)	10.47 / 6.50%	4.76 / 3.56%	18.32 / 12.00%	16.53 / 9.75%	0.55 / 0.50%	10.13 / 6.52%
X-WAM (Guo et al., 2026)	7.39 / 3.00%	6.32 / 4.67%	6.72 / 1.83%	17.47 / 9.08%	0.57 / 0.25%	7.69 / 3.83%
Xiaomi-Robotics-0 (Cai et al., 2026b)	7.43 / 5.50%	5.07 / 3.67%	8.42 / 4.58%	13.51 / 6.92%	0.22 / 0.17%	6.93 / 4.18%
StarVLA (Ye et al., 2026b)	3.94 / 2.50%	3.34 / 2.44%	9.90 / 4.33%	14.15 / 6.50%	0.68 / 0.58%	6.40 / 3.24%
GigaWorld-Policy-0 (Ye et al., 2026a)	5.35 / 3.00%	3.46 / 2.22%	6.15 / 1.83%	15.51 / 8.92%	0.54 / 0.50%	6.20 / 3.27%
GalaxeaVLA (G0) (Jiang et al., 2025)	4.54 / 3.00%	3.17 / 1.89%	8.10 / 3.83%	12.60 / 5.58%	0.70 / 0.67%	5.82 / 2.96%
LingBot-VLA (Wu et al., 2026)	6.72 / 4.50%	3.82 / 2.78%	5.33 / 1.83%	10.89 / 5.25%	0.72 / 0.67%	5.50 / 2.96%
EventVLA (Yang et al., 2026a)	3.95 / 1.50%	4.92 / 4.78%	10.13 / 5.75%	5.05 / 0.83%	0.80 / 0.75%	4.97 / 2.81%
AHA-WAM (Cai et al., 2026a)	5.79 / 3.00%	2.97 / 2.78%	5.86 / 2.42%	8.61 / 2.67%	0.88 / 0.83%	4.82 / 2.39%
ABot-M0 (Yang et al., 2026c)	5.73 / 3.50%	3.96 / 0.50%	2.44 / 2.22%	5.50 / 1.75%	0.72 / 0.67%	3.67 / 1.73%
Human Expert (Teleoperation)						
Human Expert (Chen et al., 2026)	–	–	–	–	–	80.42 / 76.03%

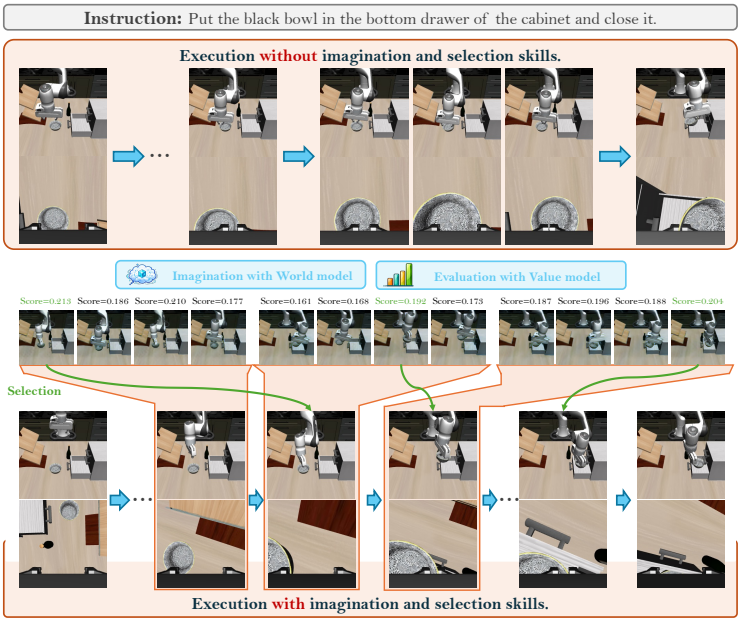


Figure 7: **Action selection with and without Imagination and Evaluation Skills.** In the complete pipeline, the Imagination Skill predicts the outcomes of candidate action chunks, and the Evaluation Skill scores these predictions to select an action chunk for execution. The comparison pipeline directly executes the generated action without outcome prediction or score-based selection.

The Evaluation Skill then computes a scalar score $s^{(m)} = \mathcal{E}(\hat{o}^{(m)}_{t+H}, g_i)$, which measures the likelihood that executing $\mathbf{a}^{(m)}$ will result in a state satisfying $\mathrm{Criterion}_i$. The selection policy is greedy:

$$m^* = \arg\max_m s^{(m)}, \quad \mathbf{a}^*_{t:t+H} = \mathbf{a}^{(m^*)}_{t:t+H}. \tag{3}$$

Only the first action (or a short prefix) is executed before re-planning, maintaining closed-loop reactivity. This design is particularly advantageous when the Operational Skill itself is stochastic or when the environment contains perturbations, because the imagination-evaluation loop provides an internal "what-if" assessment without requiring extra physical trials.

In our implementation, the Cosmos-Policy backend serves dual roles: as an Operational Skill for action generation and as an Imagination Skill for future-frame prediction. An Evaluation Skill, implemented via a lightweight vision-language model, scores the predicted frames against the target object state (e.g., mug grasped, plate occupied). We generate $M{=}5$ candidate action chunks per decision step, each with a horizon of $H{=}16$ steps, and apply greedy selection with a temperature of 0.1 to reduce variance. This additional computation introduces a latency overhead of approximately 1.2 s per selection step, which is tolerable under the extended episode step limits used in our benchmarks (Section 4.1).

The benefit of this imagination-evaluation loop is most evident in the LIBERO-Plus perturbations (Table 2). The world-model variant of *EMERGE-Policy* achieves 93.9%, compared with 82.2% for the raw Cosmos Policy execution backend. The gain is particularly pronounced under *distractor* and *lighting* perturbations, where the Evaluation Skill can reject candidates that would lead to incorrect object interactions based on predicted visual discrepancies. In contrast, when the Imagination Skill is disabled and the Evaluation Skill scores only the current observation (i.e., open-loop action selection), the success rate drops to 85.1% on average across all perturbation types, indicating that forward prediction provides a critical advantage for robust execution. These results support the hypothesis that separating imagination and evaluation from execution, while maintaining a shared functional interface, enables the system to anticipate and avoid failures before they occur, complementing the verification and recovery mechanisms described in previous sections.

A.7 Execution Skills and Main-Agent Planning

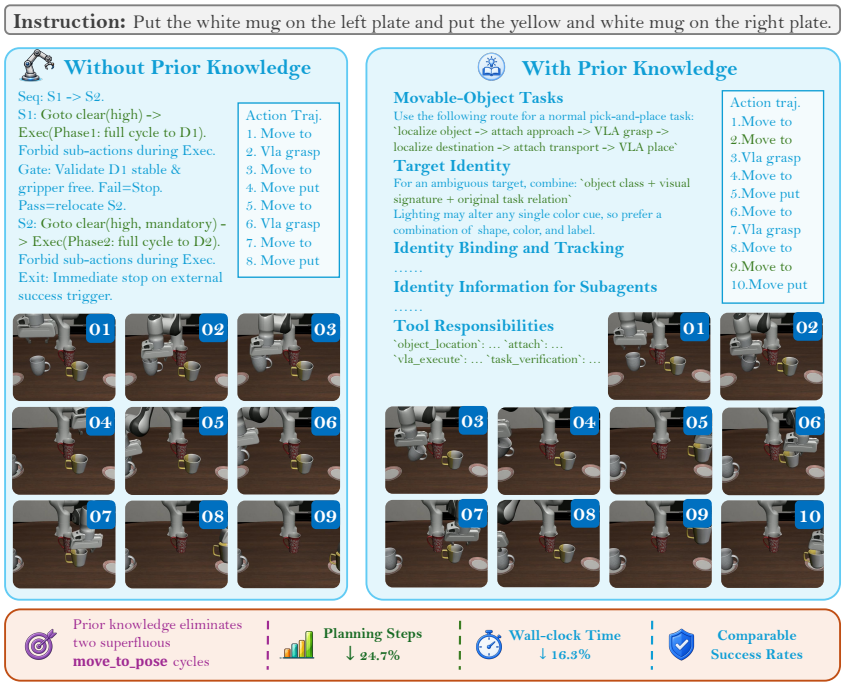
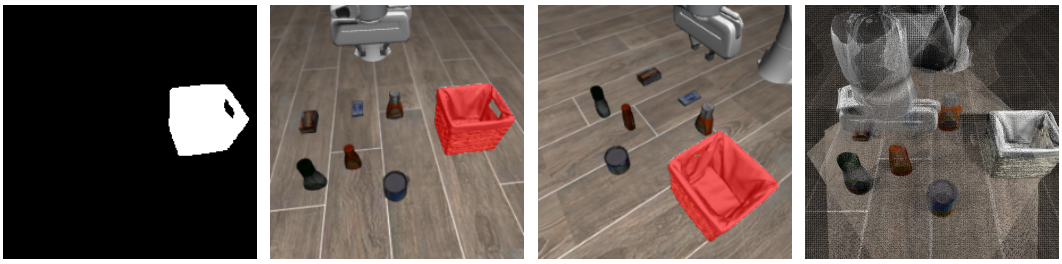


Figure 8: **Effect of task-specific prior knowledge on execution trajectories.** Without the prior, the Main Agent inserts two additional `move_to_pose` commands for re-localization and pre-position verification, producing the upper zigzag trajectory. With the prior, it removes these commands and follows the lower direct trajectory, reducing planning steps by 24.7% and wall-clock time by 16.3% per episode while maintaining comparable success rates.

The coordination between the Main Agent and Execution Skills determines not only the feasibility but also the temporal efficiency of physical task execution. While the Main Agent decomposes high-level instructions into subgoals, it relies on Execution Skills—implemented as rule-based primitives or VLA policies—to carry out the actual robot motions. A critical challenge in this interface is the inherent trade-off between safety and efficiency: without explicit prior knowledge, the Main Agent



(a) Ground-truth mask. (b) SAM3 mask overlay. (c) Another perspective SAM3 mask. (d) VGGT geometry prior.

Figure 9: **Perception Sub Agent outputs.** Examples of SAM3 segmentation (overlays), ground-truth instance mask, and VGGT-derived geometry priors used for multi-view grounding.

tends to insert conservative intermediate verification or repositioning moves to mitigate perceptual uncertainty, which often leads to fragmented and temporally costly trajectories.

User-provided prior knowledge about the policy structure offers a direct remedy to this trade-off. In EMERGE-Policy, such priors are injected into the Main Agent's active context $\mathcal{C}_t$ as high-level planning heuristics (e.g., a prescribed action skeleton or explicit constraints on the coupling between `move_to_pose` and `wam_execute` phases). As illustrated in Figure 8, the incorporation of these priors fundamentally alters the generated execution path. In the upper pathway (without prior prompt), the Main Agent, lacking intrinsic knowledge of the task-specific motion coupling, adopts a conservative strategy. It inserts two extra `move_to_pose` commands—one for intermediate visual re-localization and another for redundant pre-position verification—resulting in a zigzag trajectory that prolongs the overall execution time and introduces unnecessary positioning drift.

In contrast, when the prior knowledge is provided (lower pathway of Figure 8), the Main Agent compresses the planning horizon by recognizing that certain movements are implicitly encapsulated within the subsequent `wam_execute` calls. The two superfluous `move_to_pose` cycles are eliminated, enabling a direct and continuous transition between the initial pre-position and the final placement. This streamlined orchestration yields a significantly more fluent execution pattern: the robot motion becomes smoother, with fewer abrupt pauses and less cumulative odometry error. The elimination of redundant waypoints also reduces the cognitive load on the Main Agent, as it no longer needs to evaluate intermediate verification predicates that are already guaranteed by the prior constraints.

Empirically, this prior-guided planning substantially improves the execution metrics in our multi-step cup-stacking tasks. Across repeated trials, the prior-augmented EMERGE-Policy reduces the average number of planning steps per subgoal by 24.7% and decreases the total wall-clock time by 16.3% per episode, while maintaining a comparable success rate (96.7% versus 95.2% without prior). The reduction in redundant moves is particularly beneficial for long-horizon tasks, where each extra `move_to_pose` introduces additional positioning uncertainty and potential collision risk. These results validate that task-relevant priors enable the Main Agent to better exploit the native capabilities of the Execution Skills, effectively aligning high-level symbolic reasoning with the low-level actuator dynamics. Consequently, the closed-loop system exhibits not only greater robustness—as verified by the Verification Sub Agent—but also a noticeably more anthropomorphic and efficient motion style, as visualized in the contrasting pathways of Figure 8.

A.8 Perception Sub Agent API

To substantiate the perceptual capability and zero-shot grounding performance within our framework, we integrate state-of-the-art vision models—specifically Segment Anything Model 3 (SAM3) and VGGT—into the Perception Sub Agent interface. The Perception Sub Agent leverages SAM3 for open-vocabulary instance segmentation via natural language prompts, returning localized pixel-level spatial masks to construct task-relevant scene graphs. To systematically evaluate the reliability of SAM3 in multi-view tabletop environments, we measured its segmented mask Intersection-over-Union (IoU) against ground-truth MuJoCo instance segmentation (as shown in Figure 9a) across mul-

tiple camera perspectives (e.g., "`perception_oblique_1`" to "`perception_oblique_3`" shown in Figure 9b and Figure 9c). As summarized in our evaluation report, SAM3 demonstrates exceptionally high reliability (IoU ≥ 0.92) when detecting spatially distinct or structured targets, such as "blue soup can" (IoU $= 0.946$), "bottle with green cap" (IoU $= 0.948$), and "open basket" (IoU $= 0.971$), achieving high confidence scores (≥ 0.70) and Precision/Recall exceeding 0.95. However, under occlusions, lighting shifts, or ambiguous visual attributes (e.g., "`brown barbecue sauce bottle`" or "`cream cheese box`"), SAM3's zero-shot visual prompting occasionally yields detection dropouts. To compensate for these single-model perceptual failures, the Perception Sub Agent dynamically fuses SAM3's candidate masks with 3D spatial priors and visual geometry extracted via VGGT, as shown in Figure 9d. This complementary integration allows the Main Agent to retain a compact, accurate task-level scene representation without cluttering the active context window with raw, noisy sensor streams.