BIOLAB: BENCHMARKING ROBOTIC MANIPULATION IN BIOLOGY LABORATORY FROM INDIVIDUAL SKILLS TO EXPERIMENT SOPS

Anonymous authors
Paper under double-blind review

ABSTRACT

Robotic automation of biology experiment requires more than executing isolated manipulations. A robot must follow a standard operating procedure (SOP) across multiple instruments while adapting to varied layouts and states produced by earlier steps. Existing laboratory benchmarks assess valuable skills and multi-step tasks, but offer limited evidence about complete SOP-level manipulation. We introduce BioLab, a framework for studying how laboratory skills carry through a full workflow. We develop BioLab-Sim with configurable unified platform and hierarchical annotation & checking for scripted demonstration collection and policy evaluation. Building on this framework, we construct BioLab-Bench by decomposing an SOP into basic laboratory skills (L1), context-dependent operation sequences (L2), and complete biology experiment SOPs (L3). Occurrence-aligned demonstrations and ordered physical checks support action learning and stage-level progress evaluation, while held-out layouts test spatial generalization. We evaluate six vision-language-action policies and find limited reliability even on L1 skills. To study procedural control on longer tasks, we further propose BioLab-HVLA, which combines execution memory and visual progress decisions with leaf-conditioned action control. This hierarchy supports longer workflow execution, although physical checks reveal early manipulation failures and discrepancies between predicted progress and actual completion. We demonstrate that BioLab provides a unified setting for learning, evaluating, and diagnosing the transition from laboratory skills to SOP-level biological manipulation.

1 INTRODUCTION

Biological experiments unfold through a series of physical operations. Samples are prepared, transferred, treated, and stored using different instruments. Researchers coordinate these steps while repeating many routine actions (Lan et al., 2025; Li et al., 2025). Recent advances in embodied AI suggest that robots could take on more of this work using standard laboratory equipment (Liu et al., 2023; Chen et al., 2025; Ren et al., 2026). For such automation to matter in practice, a robot must complete the procedure that connects these actions.

Current evaluations provide only a partial answer to whether robots can do so. Laboratory benchmarks test individual skills and multi-step tasks, including longer experimental sequences (Lan et al., 2025; Liu et al., 2026; Li et al., 2025). Yet success on these tasks does not establish whether the same skills remain reliable throughout a complete standard operating procedure (SOP). Earlier actions change the scene for later ones, while revisited tools and similar visual states may require different next steps. A final success score also cannot reveal where execution stopped. As learned policies are developed for laboratory use, this gap limits our ability to judge their practical value and diagnose failures. SOP-level evaluation therefore needs a continuous workflow and evidence of intermediate physical progress.

We introduce **BioLab** (Fig. 1) to support this evaluation. Its simulation framework, **BioLab-Sim**, keeps the instruments and objects needed throughout a workflow in a shared scene. The robot must find the relevant equipment at each stage and act on the state left by earlier operations. Layout presets and placement variation test performance across equipment arrangements. Demonstration

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.

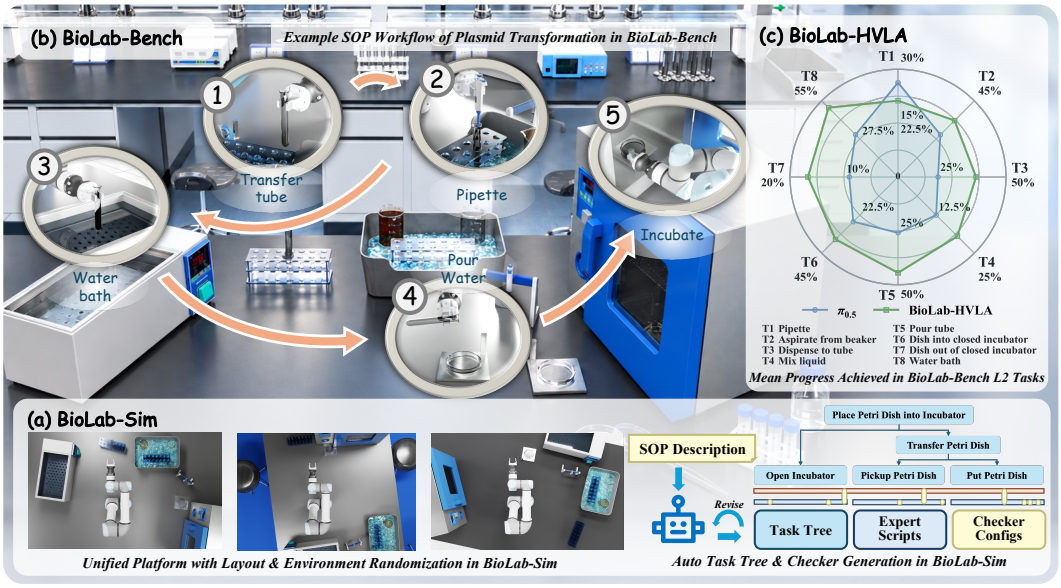

Figure 1: **Overview of BioLab.** (a) BioLab-Sim combines a unified multi-instrument platform with layout and environment randomization and SOP-driven generation of task trees, expert scripts, and checkers. (b) BioLab-Bench evaluates manipulation from individual skills to complete SOP workflows. (c) Mean progress of BioLab-HVLA and $\pi_{0.5}$ on eight L2 tasks in the Normal setting.

collection and policy evaluation use the same task definitions and physical checks. This gives skills and longer workflows a consistent physical setting.

Within this setting, **BioLab-Bench** tracks how performance changes as skills become part of a procedure. It has three levels: basic laboratory skills (L1), context-dependent operation sequences (L2), and the complete SOP manipulation workflow (L3). We derive this hierarchy by decomposing the SOP from the top down. Reusable task leaves define instructions, semantic targets, and physical criteria; ordered composites connect them into sequences and the full workflow. The shared definitions let us relate local skill performance to execution after preceding operations. Unique occurrence paths distinguish repeated uses of a task, keeping trajectory segments and physical checks aligned with the correct stage. The checks record satisfied criteria and mark later stages as not reached after a failure. Alongside root-task success, we report physical progress across the full plan.

To examine how procedural decisions affect workflow execution, we propose **BioLab-HVLA**, a simple hierarchical baseline where a VLM tracks progress and selects the next instruction for a VLA to execute. We evaluate six VLA policies on L1, compare root-conditioned, hierarchical, and oracle-instruction systems on L2, and evaluate L3 execution and real-robot experiment. Stage-level checks reveal where predicted progress diverges from physical completion. These findings show that accurate progress decisions alone cannot ensure successful manipulation. By tracing both sources of failure, BioLab makes SOP-level biological manipulation a measurable target for robotic systems.

2 Related Work

Embodied laboratory benchmarks. With the rise of embodied AI, many simulation benchmarks have been proposed (Liu et al., 2023; James et al., 2020; Chen et al., 2025; Han et al., 2026; Zhang et al., 2025b; Gao et al., 2026; Fei et al., 2025; Zhou et al., 2025; Chen et al., 2026a;b). While these benchmarks mainly focus on household environments, the evaluation of embodied intelligence in other specified scenarios (*e.g.*, laboratory) is still limited. Recently, more embodied benchmarks start to turn their attention to laboratory manipulation (Lan et al., 2025; Li et al., 2025; Liu et al., 2026; Ren et al., 2026; Wu et al., 2026; Wang et al., 2026; Cao et al., 2026; Tang et al., 2026; Bai et al., 2026; Li et al., 2024). LabUtopia provides a hierarchical benchmark towards chemistry experiments (Li et al., 2025). SafeLab (Bai et al., 2026) focuses on safety in chemical experiments.

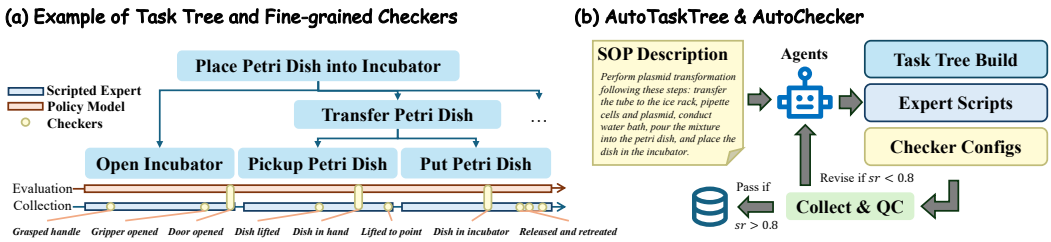

Figure 2: **AutoTaskTree and AutoChecker in BioLab-Sim.** (a) A Petri-dish placement task is decomposed into instruction-labeled composite and leaf tasks, yielding hierarchical stage annotations stored in native records. Yellow markers indicate physical checks along the scripted-collection (blue) and policy-evaluation (orange) timelines, with mode-specific checking requirements. (b) An SOP-driven agent generates task trees, expert scripts, and checker configurations. Simulated collection and quality control provide feedback for iterative revision (sr: collection success rate).

However, they are confined to chemical manipulations and cannot support simulation for biological experiments. AutoBio (Lan et al., 2025) builds the first benchmark for biological experiments. Pipette (Liu et al., 2026) provides a benchmark for pipetting. However, these benchmarks mainly focus on short-horizon biological manipulation skills. To fill this gap, we propose BioLab, a comprehensive manipulation benchmark towards long-horizon biological experiment operations.

Benchmarks for long-horizon manipulation. RLBench (James et al., 2020), LIBERO (Liu et al., 2023), and RoboTwin (Chen et al., 2025) have promoted robotic manipulation by benchmarking short tasks. Recently, the community has increasingly focused on long-horizon tasks (Peng et al., 2026; Han et al., 2026; Chen et al., 2026a; Zhang et al., 2025b; Mees et al., 2022; Zhang et al., 2025a). RoboCerebra (Han et al., 2026) evaluates high-level reasoning for long-horizon manipulation. RoboDojo (Chen et al., 2026a) establishes a unified sim-and-real benchmark for comprehensive evaluation. However, these benchmarks are designed for household scenarios, while long-horizon manipulation benchmarks dedicated to specialized domains (*e.g.*, laboratory) remain scarce. To fill this gap, we propose BioLab for long-horizon manipulation tasks in biological laboratory.

3 BioLab-Sim

BioLab-Sim supports full cycle of task construction, demonstration collection, and policy evaluation for biological experiment SOPs. Complete standard operating procedure (SOP) execution imposes two requirements: a) a unified platform containing all instruments needed across the workflow; b) a task structure that exposes intermediate stages for annotation and checking. To address these issues, we develop a unified platform in §3.1, and introduce hierarchical annotation and checking in §3.2. BioLab-Sim also provides annotated laboratory assets and occurrence-aligned recordings (§3.3).

3.1 Unified Platform with Layout and Environment Randomization

BioLab-Sim places all instruments required by an experiment SOP within the robot's workspace. This unified platform supports tasks of every granularity, from basic skills to complete SOPs. Specifically, for simple tasks that interact with only one instrument, BioLab-Sim still keeps the full equipment set in the scene. Unlike conventional lab benchmarks that place only task-required instruments on the workbench, our unified platform setting can better expose policies to spatial constraints and skill composition in lab workflows. We also introduce randomization on layouts, environments and object poses to guarantee strong generalization, whose details can be found in Appendix A.

3.2 Hierarchical Stage Annotation and Progress Checking

Existing laboratory benchmarks report task-level success for individual and composed tasks (Lan et al., 2025; Li et al., 2025). For a long SOP, a single success or failure outcome cannot show which stages were completed or whether later stages were reached. Manually defining hierarchical stage annotations and checkers for every step and sub-step also becomes difficult as the procedure grows.

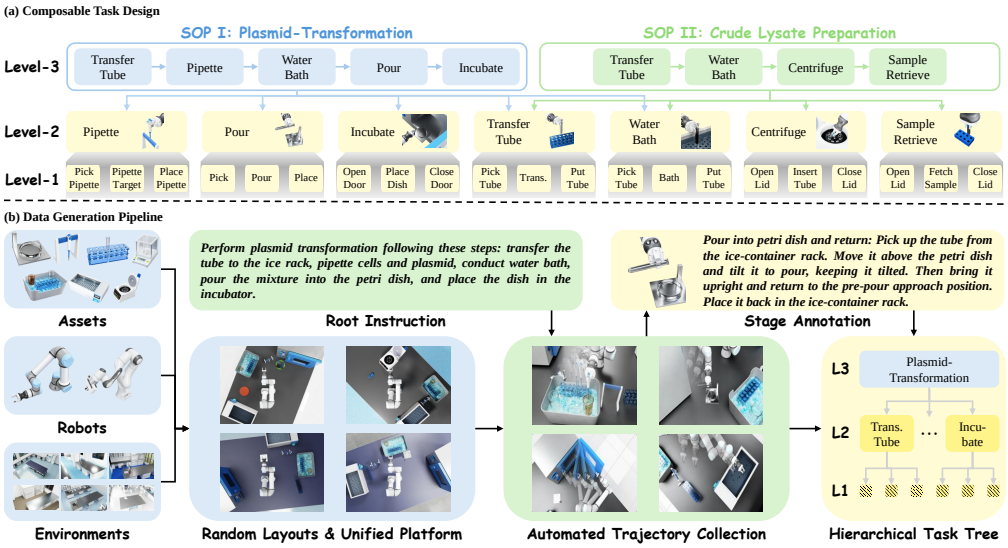

Figure 3: **Overview of BioLab-Bench.** (a) Based on inherent SOP nature of biological experiments, BioLab-Bench hierarchically partitions biological manipulation tasks into three levels. (b) Data generation pipeline features unified platforms with random layouts and hierarchical annotations.

BioLab-Sim addresses these with **AutoTaskTree** and **AutoChecker**. As shown in Fig. 2, AutoTaskTree represents every SOP as a tree where ordered composites connect reusable skills into sequences to construct the whole SOP. Every leaf or node of the tree carries its own instruction, forming hierarchical multi-stage annotations for the full protocol. Based on the task tree, AutoChecker attaches a sequence of checkers with a state pointer to each leaf. At every simulation step, it evaluates the criterion under the pointer and advances the pointer when that criterion passes. Once the current leaf completes, the task tree moves to the next node. Together, AutoTaskTree and AutoChecker show which stages completed, which remain pending, and which were never reached after a failure. Per-stage outcomes contribute to the progress metrics reported alongside task success.

To obtain these task trees and checkers without authoring every task by hand, we design an agent that takes an SOP and decomposes it while reusing existing composite and leaf tasks. The agent automatically annotates instructions for new nodes and designs checkers for new leaves. Candidate tasks then undergo iterative simulated collection and replay until they meet the quality requirements.

3.3 Interactive Assets and Aligned Demonstration Data

We build BioLab-Sim on LabScene, an open-source laboratory asset library of LabUtopia (Li et al., 2025). Specifically, we extend LabScene with functional assets needed for biology experiments, such as centrifuge tubes, racks, petri dishes, centrifuge, *etc.*. Each asset is annotated with operational affordances, such as grasp points on tubes and handles, and entry points for tubes, beakers, and wells. These annotations provide the interaction geometry used by task definitions across scene layouts.

BioLab-Sim stores each run in a native format before exporting it for training. The record synchronizes top and wrist camera streams, an optional front camera, measured robot state, commands, instructions, and checker timelines. Exporters convert these records into a LeRobot action dataset (Cadene et al., 2026) with selectable instruction granularity and a VLM dataset for stage-progress decisions (§5). Both preserve task-occurrence alignment for stage-wise supervision.

4 BioLab-Bench

4.1 From SOP Workflow to Basic Skills

General-purpose manipulation benchmarks have largely focused on household tasks (James et al., 2020; Liu et al., 2023; Chen et al., 2025). Biological laboratory SOPs connect instrument operations into extended procedures; an end-to-end BioLab episode can exceed four minutes. At this scale,

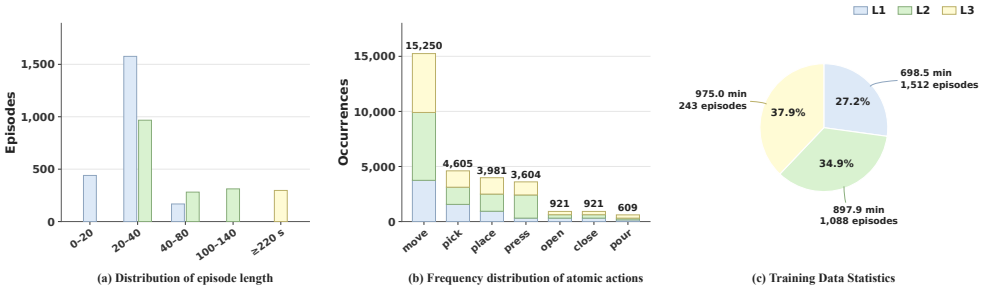

Figure 4: **Dataset Statistics.** BioLab-Bench features long trajectories and diverse skills.

terminal success alone cannot show where execution broke down. We construct BioLab-Bench from the top down, decomposing complete SOPs into operation sequences and reusable skills. This keeps each smaller task tied to its role in the procedure. Evaluation then proceeds from L1 to L3, adding procedural context at each level (Fig. 3(a)):

- **L1: Individual laboratory skills.** L1 evaluates 14 reusable skills, including pipetting, pouring, and picking. Each standalone task establishes performance on a local operation used in longer workflows. Some skills contain several physical stages and are not limited to atomic movements.
- **L2: Context-dependent operation sequences.** L2 comprises eight contextual task groups: Pipette, Aspirate from beaker, Dispense to tube, Mix liquid, pouring, water-bath handling, and two incubator operations (Appendix C). Starting states and required next actions depend on preceding steps. This level reveals difficulties that a standalone skill test cannot capture.
- **L3: Complete SOP workflows.** L3 connects operation sequences into full manipulation workflows for plasmid transformation, crude lysate preparation, and liquid culture expansion. It tests whether a policy can sustain the prescribed procedure through states produced by its own actions. Layouts and initial configurations vary across episodes, while SOP protocols remain fixed.

4.2 SOP-Aligned Data Generation

The BioLab-Bench data generation pipeline is shown in Fig. 3(b). BioLab-Sim first instantiates a multi-instrument scene from shared simulation assets, robot models, laboratory environments, and sampled layouts. Before each rollout, the task tree defines the instructions and physical checks for the selected L1, L2, or L3 task. Scripted experts then execute the task while BioLab-Sim records observations, actions, task occurrences, and checker outcomes. The recordings are exported as complete episodes or leaf segments for action learning and progress supervision. Their scene configurations are also retained for matched policy evaluation. Since the task tree is active during collection, stage annotations remain aligned with the operations and physical states observed in the rollout.

Fig. 4 summarizes the BioLab-Bench training set, which comprises 2,843 episodes across three levels: 1,512 at L1, 1,088 at L2, and 243 at L3. Together, these episodes span 42.5 hours. Episode duration increases with procedural scope: L1 trajectories average approximately 30 seconds, L2 trajectories approximately 60 seconds, and every L3 trajectory exceeds 220 seconds. L3 thus tests sustained execution over a substantially longer interval than the skill and operation-sequence tasks. The dataset also covers varied local manipulations; Fig. 4(b) shows the seven most frequent atomic action categories. Together, the episode counts, durations, and action distribution describe the data available for learning local skills and studying their composition into complete workflows.

Table 1 compares BioLab-Bench with existing laboratory benchmarks. BioLab-Bench brings long-horizon tasks, a unified multi-instrument platform, layout generalization tests, hierarchical annotations, and stage-level evaluation into one setting.

5 BioLab-HVLA

To investigate the difficulty of BioLab-Bench, besides off-the-shelf models, we also present BioLab-HVLA (Fig. 5), a simple baseline with hierarchical memory and planning capabilities.

Table 1: Embodied laboratory benchmarks. Chem. and Bio. denote chemistry and biology.

Benchmark	Domain	Tasks	Task scope		Scene variation			Evaluation	
			Long horizon	Unified platform	Scene rand.	Layout rand.	OOD	Hier. annot.	Stage eval.
Chemistry3D (Li et al., 2024)	Chem.	5	✗	✗	✗	✗	✗	✗	✗
LabUtopia (Li et al., 2025)	Chem.	30	✓	✗	✓	✓	✓	✗	✓
LabDex (Tang et al., 2026)	Chem.	41	✓	✗	✗	✗	✓	✓	✗
SafeLab (Bai et al., 2026)	Chem.	64	✗	✗	✓	✓	✗	✗	✗
Labimus (Wu et al., 2026)	Chem.	7	✓	✓	✓	✗	✗	✗	✗
RoboGenesis (Ren et al., 2026)	Chem.	–	✓	✗	✓	✓	✓	✓	✗
AutoBio (Lan et al., 2025)	Bio.	16	✗	✗	✗	✗	✗	✗	✗
Pipette (Liu et al., 2026)	Bio.	12	✗	✗	✓	✗	✗	✗	✗
BioLab-Bench (ours)	**Bio.**	**28**	✓	✓	✓	✓	✓	✓	✓

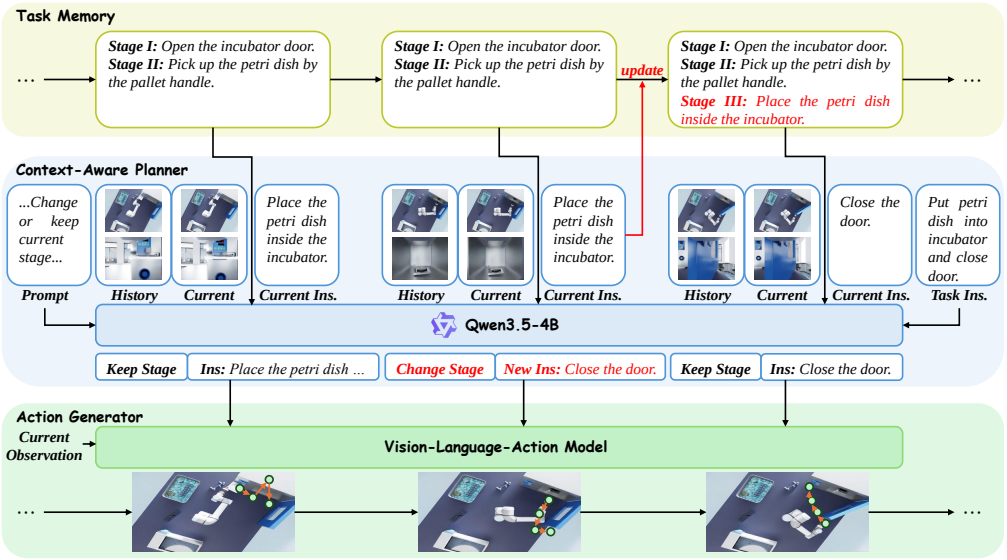

Figure 5: **Overview of BioLab-HVLA**. BioLab-HVLA features hierarchical memory and planning. Task memory is maintained to record long-horizon stage information, while history observations provide short-term visual memory for planning. The context-aware planner performs high-level planning, while VLA generates low-level actions.

Herarchical Memory. Memory is indispensable for a model to complete long-horizon biological experiments. Specifically, memory can be categorized into long-term memory and short-term memory. The long-term memory keeps track of completed steps, whereas the short-term memory distinguishes visually confusable actions that are hard to tell apart from single-frame images (*e.g.*, pouring a test tube vs. uprighting a test tube, opening a door vs. closing a door). In BioLab-VLA, we maintain long-term memory by appending completed stages to the task memory upon stage switching, which continuously serves as the context for the planner. For short-term memory, at timestep t, the observations at timestep $t - 15$ are provided to the planner as visual history to differentiate visually similar action processes.

Hierarchical Planning. To equip BioLab-VLA with long-horizon manipulation ability, we establish a hierarchical planning scheme. Specifically, the context-aware planner first derives the instruction for the current stage based on the global task instruction and context. The planner is required to determine whether to switch the stage. If no stage transition occurs, the current VLA instruction is retained. Otherwise, the system proceeds to the next stage, where a new instruction is generated for the VLA. The VLA then generates executable actions upon receiving instruction from the planner.

Table 2: Level 1 (L1) in-domain task results (%). Each method reports the success rate (SR) on the first line and task progress on the second line. Eech task has 36 rollouts per policy.

Policy	Open centr. lid	Close centr. lid	Open incub. door	Close incub. door	Centr. insert	Centr. remove	Pipette beaker	Pipette tube	Dish pads	Dish into open incub.	Tube racks	Tube ice	Pour tube	Water bath	**Overall**
$\pi_{0.5}$	63.9	44.4	27.8	80.6	0.0	0.0	16.7	30.6	38.9	44.4	25.0	30.6	41.7	52.8	**35.52**
	63.9	46.3	27.8	80.6	38.4	27.0	34.3	44.0	46.5	51.4	48.0	45.2	57.8	60.2	**47.95**
DM0.5	63.9	33.3	91.7	63.9	0.0	0.0	13.9	8.3	27.8	33.3	25.0	5.6	41.7	0.0	29.17
	63.9	33.3	91.7	63.9	25.5	29.0	26.4	22.2	33.3	36.4	44.4	40.9	55.0	49.4	43.95
X-VLA	47.2	8.3	11.1	63.9	0.0	0.0	2.8	8.3	11.1	30.6	5.6	0.0	11.1	2.8	14.48
	47.2	8.3	11.1	63.9	17.1	27.8	8.8	24.5	14.9	35.0	27.4	26.2	36.7	30.6	27.11
GR00T N1.7	5.6	13.9	2.8	33.3	0.0	0.0	0.0	0.0	11.1	2.8	5.6	0.0	5.6	0.0	5.75
	5.6	13.9	2.8	33.3	16.7	15.9	5.6	3.2	13.6	3.9	26.6	22.2	27.8	20.7	15.12
SmolVLA	5.6	22.2	5.6	11.1	0.0	0.0	0.0	0.0	0.0	2.8	0.0	0.0	0.0	0.0	3.37
	5.6	22.2	5.6	11.1	16.7	11.9	0.9	0.9	0.0	2.8	14.3	18.2	20.0	11.1	10.09
InternVLA	2.8	13.9	0.0	16.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.38
	2.8	13.9	0.0	16.7	6.9	5.6	0.0	0.0	0.0	0.0	5.6	12.3	15.6	8.0	6.23

Table 3: Level 1 (L1) Layout-OOD task results (%). Each method reports the success rate (SR) on the first line and task progress on the second line.Each task is evaluated on 12 rollouts per policy.

Policy	Open centr. lid	Close centr. lid	Open incub. door	Close incub. door	Centr. insert	Centr. remove	Pipette beaker	Pipette tube	Dish pads	Dish into open incub.	Tube racks	Tube ice	Pour tube	Water bath	**Overall**
$\pi_{0.5}$	66.7	33.3	50.0	83.3	0.0	0.0	8.3	0.0	41.7	8.3	8.3	8.3	0.0	58.3	**26.19**
	66.7	33.3	50.0	83.3	50.0	35.7	25.0	23.6	54.5	8.3	35.7	35.7	23.3	63.0	**42.02**
DM0.5	25.0	16.7	91.7	33.3	0.0	8.3	0.0	8.3	16.7	0.0	16.7	0.0	8.3	0.0	16.07
	25.0	16.7	91.7	33.3	38.9	34.5	11.1	23.6	16.7	0.0	46.4	39.3	40.0	31.5	32.05
X-VLA	50.0	0.0	25.0	25.0	0.0	0.0	8.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	7.74
	55.6	0.0	25.0	25.0	11.1	21.4	25.0	18.1	3.8	0.0	25.0	28.6	23.3	12.0	19.56
GR00T N1.7	16.7	8.3	8.3	50.0	0.0	0.0	0.0	0.0	8.3	8.3	0.0	0.0	0.0	0.0	7.14
	16.7	8.3	8.3	50.0	20.8	15.5	2.8	5.6	8.3	11.7	20.2	26.2	25.0	18.5	16.99
SmolVLA	16.7	0.0	0.0	41.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	4.17
	16.7	0.0	0.0	41.7	13.9	14.3	0.0	2.8	0.0	0.0	14.3	14.3	20.0	11.1	10.64
InternVLA	0.0	0.0	0.0	16.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.19
	0.0	0.0	0.0	16.7	4.2	4.8	0.0	2.8	0.0	0.0	3.6	8.3	16.7	7.4	4.60

6 EXPERIMENTS

We evaluate individual skills on L1, procedural execution on L2/L3, and adopt real-robot test.

6.1 L1: INDIVIDUAL LABORATORY SKILLS

Setup. We evaluate $\pi_{0.5}$ (Physical Intelligence et al., 2025), DM0.5 (Dexmal, 2026), X-VLA (Zheng et al., 2025), GR00T N1.7 (NVIDIA, 2026), SmolVLA (Shukor et al., 2025), and InternVLA-A1.5 (Ma et al., 2026). Each policy receives top-view and wrist images, robot state, and a task instruction. All six policies are trained on the same 1,512 trajectories spanning 14 L1 tasks. Each task is evaluated on 36 in-domain rollouts and 12 Layout-OOD rollouts per policy. The in-domain split samples new episodes from the training layout-preset distribution, whereas Layout-OOD uses a held-out preset. Training details are provided in Appendix F.

Results. Tables 2 and 3 report task-level performance. $\pi_{0.5}$ achieves the highest overall success rates, with 35.52% in-domain and 26.19% on Layout-OOD; DM0.5 follows with 29.17% and 16.07%, respectively. Fine-grained operations remain a major bottleneck: across all policies and both splits, centrifuge insertion succeeds in none of 288 rollouts, while centrifuge removal succeeds in only one of 288. The success rate of $\pi_{0.5}$ decreases by 26.27% relative to its in-domain result on Layout-OOD, indicating that changes in equipment layout add difficulty even at the skill level.

Table 4: Level 2 (L2) in-domain task results (%). Each method reports the success rate (SR) on the first line and task progress on the second line.

Method	Pipette	Aspirate from beaker	Dispense to tube	Mix liquid	Pour tube	Dish into closed incub.	Dish out of closed incub.	Water bath	**Overall**
$\pi_{0.5}$	3.75 26.25	8.33 25.00	5.56 18.52	0.00 12.50	2.78 25.93	11.11 26.39	0.00 9.03	2.78 30.56	4.07 22.02
BioLab-HVLA	1.25 21.25	13.89 33.33	13.89 36.11	2.08 19.44	13.89 44.44	8.33 36.81	0.00 16.67	22.22 50.00	8.14 **30.40**
Oracle	5.00 24.38	16.67 37.96	30.56 43.52	2.08 15.63	5.56 27.08	0.00 32.64	0.00 18.06	30.56 46.30	**10.17** 29.36

Table 5: Level 2 (L2) Layout-OOD evaluation results (%) across eight tasks. Each method reports the success rate (SR) on the first line and task progress on the second line.

Method	Pipette	Aspirate from beaker	Dispense to tube	Mix liquid	Pour tube	Dish into closed incub.	Dish out of closed incub.	Water bath	**Overall**
$\pi_{0.5}$	0.00 6.25	0.00 13.89	0.00 5.56	0.00 2.78	0.00 2.78	0.00 16.67	0.00 8.33	0.00 33.33	0.00 9.64
BioLab-HVLA	0.00 15.63	0.00 22.22	16.67 44.44	0.00 14.58	8.33 16.67	0.00 37.50	0.00 20.83	8.33 19.44	3.13 21.74
Oracle	3.13 16.41	0.00 30.56	16.67 38.89	0.00 14.58	8.33 16.67	0.00 33.33	0.00 20.83	8.33 19.44	**3.91** **21.81**

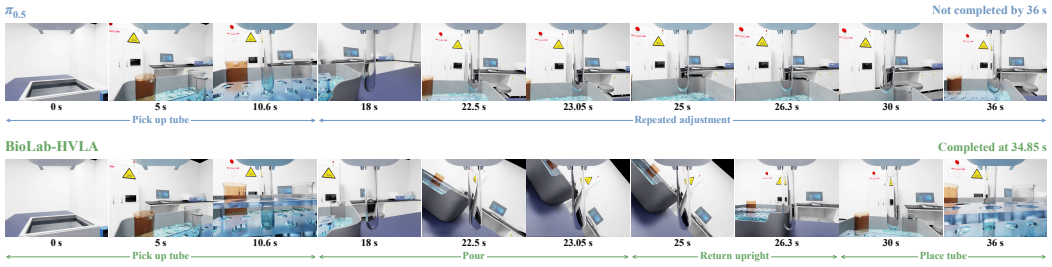

Figure 6: Visualization of the L2 task *Pour Water*. BioLab-HVLA can distinguish different stages autonomously, while $\pi_{0.5}$ struggles with similar observations.

6.2 L2: Context-Dependent Operation Sequences

Setup. We compare direct $\pi_{0.5}$ control with BioLab-HVLA on L2 tasks. Both methods use 1,088 training trajectories and are evaluated on 344 in-domain and 128 Layout-OOD rollouts. Direct $\pi_{0.5}$ receives the global task instruction throughout an episode. BioLab-HVLA instead uses a fine-tuned Qwen VLM to track progress and select the current sub-instruction for its VLA executor. We also evaluate an Oracle variant that replaces the VLM-selected sub-instruction with the simulator's ground-truth instruction for the current stage. This variant isolates the effect of instruction selection while retaining the same low-level executor.

Results. BioLab-HVLA approximately doubles in-domain SR of direct $\pi_{0.5}$ control (Table 4). On Layout-OOD, direct control achieves no successful rollouts, whereas BioLab-HVLA reaches 3.13% success, retaining 38.45% of its 8.14% in-domain SR (Table 5). Oracle reaches 10.17% in-domain and 3.91% on Layout-OOD. The relatively small Oracle gap suggests that instruction selection is not the only limiting factor. The low absolute success rates, together with the L1 results, point to local manipulation as a substantial remaining bottleneck. Figure 6 visualizes an example L2 rollout.

6.3 L3: SOP-Level Workflow Execution

Setup. We evaluate BioLab-HVLA on the plasmid-transformation SOP under in-domain setting as an L3 example. This approximately four-minute procedure contains 19 L1-level task occurrences.

Table 6: Real-robot evaluation on incubator operation and water-bath handling. Step columns report the number of rollouts completing each step out of 15 trials. Success rate (SR) and mean progress are percentages; boldface marks the higher value for each task.

Incubator operation

Method	N	Open door	Close door	SR (%)	Mean progress (%)
$\pi_{0.5}$	15	8	7	46.67	50.00
BioLab-HVLA	15	13	8	**53.33**	**70.00**

Water-bath handling

Method	N	Pickup	Water bath	Pour	Return	SR (%)	Mean progress (%)
$\pi_{0.5}$	15	12	11	9	3	20.00	58.33
BioLab-HVLA	15	13	12	10	6	**40.00**	**68.33**

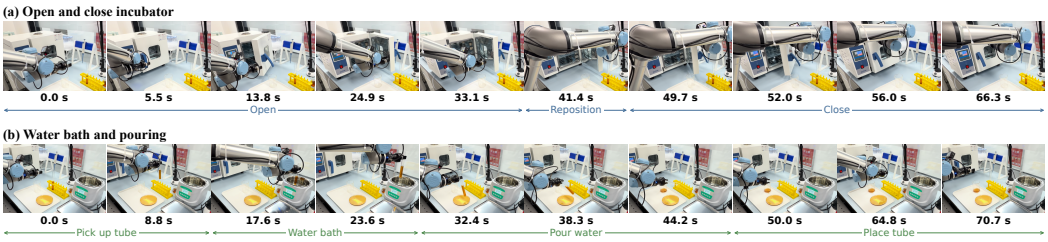

Figure 7: Visualizations of Real-Robot Experiments (BioLab-HVLA).

Results. BioLab-HVLA achieves 10.82% progress but no complete successful rollout. The nonzero progress shows that some physical requirements are satisfied, while zero success rate shows that the policy cannot yet sustain execution through the full procedure.

6.4 Real-Robot Experiments

Setup. We use a UR5e with a Robotiq Hand-E gripper, top-view and wrist-mounted Osmo Action cameras, and EVA-Client (Yang et al., 2026) for data collection and deployment. We fine-tune $\pi_{0.5}$ and BioLab-HVLA with collected real-robot data, then evaluate each method on 15 rollouts per task for incubator operation and water-bath-pouring handling.

Results. Table 6 shows that BioLab-HVLA generally outperforms direct $\pi_{0.5}$ control on evaluated tasks, and Figure 7 visualizes representative BioLab-HVLA executions. The advantage is consistent across the step columns: BioLab-HVLA completes every reported step at least as often as $\pi_{0.5}$, and reaches the final return step twice as often on water-bath handling. Its success rate on this task is also twice that of $\pi_{0.5}$. These trials show that stage-aware control also helps in physical scenes.

7 Conclusion

Biological automation ultimately depends on whether a robot can carry a complete procedure, not on isolated skills. BioLab makes this question measurable and diagnosable. BioLab-Sim hosts a full SOP on one randomized multi-instrument platform and attaches hierarchical stage annotations and physical progress checks to every step; BioLab-Bench organizes evaluation from individual skills (L1) through context-dependent sequences (L2) to complete SOPs (L3); and BioLab-HVLA supplies a hierarchical reference with execution memory and progress-driven instruction selection. Three findings emerge. First, current VLA policies remain unreliable even on single fine-grained skills. Second, hierarchical memory and planning double sequence-level success, although low-level manipulation remains a bottleneck. Third, the same stage-aware control also improves long-horizon execution in physical scenes. Together, these results establish a common basis for comparing how far policies progress through laboratory procedures and for identifying the stages where execution stops.

AI USE STATEMENT

Generative AI tools were used to assist with language polishing and code debugging. But all technical content, experimental design, results, and conclusions were independently verified by the authors, who take full responsibility for the content of the paper.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the model architecture, dataset construction procedure, data processing, and evaluation protocol in the main paper and Appendix. Additional implementation details and results are also included to facilitate reproduction of our results.

ETHICS STATEMENT

This work adheres to the ICLR Code of Ethics. The study does not involve human subjects or the collection of any personally identifiable or sensitive information. The experiments are conducted in simulated and controlled robotic environments and do not raise privacy concerns. We are not aware of any additional ethical issues associated with the research presented in this work.

REFERENCES

Fengshuo Bai, Yufeng Li, Ruihai Wu, Peishuo Wang, Yuhan Wang, Bernie Hao Zhu, Yuanfei Wang, Tawei Chou, Jing Gao, Runchuan Zhu, et al. Safelab: An interactive high-fidelity benchmark for embodied safety in scientific robotics. In *Forty-third International Conference on Machine Learning*, 2026.

Remi Cadene, Simon Aliberts, Francesco Capuano, et al. LeRobot: An open-source library for end-to-end robot learning, 2026.

Jianbao Cao, Zhangrui Zhao, Bohan Feng, Zixuan Hu, Rui Li, Haiyuan Wan, Chenxi Li, Jingyuan Li, Wenzhe Cai, Lei Bai, et al. Labbuilder: Protocol-grounded 3d layout generation for interactable and safe laboratory. *arXiv preprint arXiv:2605.02288*, 2026.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. RoboTwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025.

Tianxing Chen, Yue Chen, Zixuan Li, Junyuan Tang, Kailun Su, Haoran Lu, Weijie Wan, Baijun Chen, Songling Liu, Haowen Yan, et al. Robodojo: A unified sim-and-real benchmark for comprehensive evaluation of generalist robot manipulation policies. *arXiv preprint arXiv:2607.04434*, 2026a.

Tianxing Chen, Yuran Wang, Mingleyang Li, Yan Qin, Hao Shi, Zixuan Li, Yifan Hu, Yingsheng Zhang, Kaixuan Wang, Yue Chen, Hongcheng Wang, Renjing Xu, Ruihai Wu, Yao Mu, Yaodong Yang, Hao Dong, and Ping Luo. RMBench: Memory-dependent robotic manipulation benchmark with insights into policy design, 2026b. URL https://arxiv.org/abs/2603.01229.

Dexmal. DM0.5 model card, 2026. URL https://huggingface.co/Dexmal/DM05.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, et al. Libero-plus: In-depth robustness analysis of vision-language-action models. *arXiv preprint arXiv:2510.13626*, 2025.

Chen Gao, Liankai Jin, Xingyu Peng, Jiazhao Zhang, Yue Deng, Annan Li, He Wang, and Si Liu. Octonav: Towards generalist embodied navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 40074–40084, 2026.

Songhao Han, Boxiang Qiu, Yue Liao, Siyuan Huang, Chen Gao, Shuicheng Yan, and Si Liu. Robocerebra: A large-scale benchmark for long-horizon robotic manipulation evaluation. *Advances in Neural Information Processing Systems*, 38, 2026.

Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J. Davison. RLBench: The robot learning benchmark and learning environment. *IEEE Robotics and Automation Letters*, 5(2): 3019–3026, 2020.

Zhiqian Lan, Yuxuan Jiang, Ruiqi Wang, Xuanbing Xie, Rongkui Zhang, Yicheng Zhu, Peihang Li, Tianshuo Yang, Tianxing Chen, Haoyu Gao, Xiaokang Yang, Xuelong Li, Hongyuan Zhang, Yao Mu, and Ping Luo. AutoBio: A simulation and benchmark for robotic automation in digital biology laboratory, 2025. URL https://arxiv.org/abs/2505.14030.

Rui Li, Zixuan Hu, Wenxi Qu, Jinouwen Zhang, Zhenfei Yin, Sha Zhang, Xuantuo Huang, Hanqing Wang, Tai Wang, Jiangmiao Pang, et al. LabUtopia: High-fidelity simulation and hierarchical benchmark for scientific embodied agents. *arXiv preprint arXiv:2505.22634*, 2025.

Shoujie Li, Yan Huang, Changqing Guo, Tong Wu, Jiawei Zhang, Linrui Zhang, and Wenbo Ding. Chemistry3d: Robotic interaction benchmark for chemistry experiments. *arXiv preprint arXiv:2406.08160*, 2024.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023.

Zhe Liu, Huanbo Jin, Zhaohui Du, Zhe Wang, Dongzhan Zhou, Minting Pan, He Xu, Peijia Li, Jiaming Gu, Quan Lu, Qi Wang, Bin Ji, and Ting Xiao. Pipette: An embodied simulation platform, benchmark, and success-verified simulation augmentation framework for wet-lab robotics, 2026. URL https://arxiv.org/abs/2606.12936.

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.

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*, 7(3):7327–7334, 2022.

NVIDIA. GR00T N1.7-3B model card, 2026. URL https://huggingface.co/nvidia/GR00T-N1.7-3B. Model revision 2fc962b973bccdd5d8ce4f67cc63b264d6886495.

Xingyu Peng, Chen Gao, Liankai Jin, Annan Li, and Si Liu. BiCoord: A bimanual manipulation benchmark towards long-horizon spatial-temporal coordination, 2026. URL https://arxiv.org/abs/2604.05831.

Physical Intelligence, Kevin Black, Noah Brown, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization, 2025.

Baochang Ren, Xinjie Liu, Xi Chen, Yanshuo Liu, Chenxi Li, Daqi Gao, Zeqin Su, Jintao Xing, Zirui Xue, Rui Li, Xiangyu Zhao, Shuofei Qiao, Minting Pan, Wangmeng Zuo, Lei Bai, Dongzhan Zhou, Ningyu Zhang, and Huajun Chen. LabVLA: Grounding vision-language-action models in scientific laboratories, 2026. URL https://arxiv.org/abs/2606.13578.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, et al. SmolVLA: A vision-language-action model for affordable and efficient robotics, 2025.

Zhipeng Tang, Sihang Chen, Sha Zhang, Peihao Yang, Yan Liu, Wentao Zhao, Xinrui Liu, Rui Huang, Wensheng Du, Yuting Huang, et al. Labdex: A hierarchical benchmark for dexterous manipulation in laboratories. *arXiv preprint arXiv:2608.18618*, 2026.

Haobo Wang, Baoli Sun, Anqi Zou, Dongsheng Huang, Zelin Lv, Ning Wang, Rui Li, Dongzhan Zhou, Weiyu Guo, Zhihui Wang, et al. Labrobfail: A benchmark for robotic failure analysis in chemical self-driving laboratories. *arXiv preprint arXiv:2607.23704*, 2026.

Yuhan Wu, Zhao Jin, Tao Li, Yuheng Zhang, Zhichao Wang, Shuo Wang, Jun Jiang, Xiaobo Li, Yanyong Zhang, Jian Tang, et al. Labimus: A simulation and benchmark for humanoid dexterous manipulation in chemical laboratory. *arXiv preprint arXiv:2606.31037*, 2026.

Heqing Yang, Yang Yi, Liyao Wang, Linqing Zhong, Donglin Yang, Ruipu Wu, Zitong Bai, Fengjiao Chen, Manyuan Zhang, Linjiang Huang, and Si Liu. EVA-Client: A unified data collection, inference, and deployment framework for embodied policies on real robots, 2026. URL `https://arxiv.org/abs/2607.02646`.

Borong Zhang, Jiahao Li, Jiachen Shen, Yuhao Zhang, Yishuai Cai, Lu Liu, Hailu Ji, Yuanpei Chen, Juntao Dai, Jiaming Ji, et al. Vla-arena: An open-source framework for benchmarking vision-language-action models. *arXiv preprint arXiv:2512.22539*, 2025a.

Shiduo Zhang, Zhe Xu, Peiju Liu, Xiaopeng Yu, Yuan Li, Qinghui Gao, Zhaoye Fei, Zhangyue Yin, Zuxuan Wu, Yu-Gang Jiang, et al. Vlabench: A large-scale benchmark for language-conditioned robotics manipulation with long-horizon reasoning tasks. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11142–11152. IEEE, 2025b.

Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, Ya-Qin Zhang, Jiangmiao Pang, Jingjing Liu, Tai Wang, and Xianyuan Zhan. X-VLA: Soft-prompted transformer as scalable cross-embodiment vision-language-action model, 2025.

Xueyang Zhou, Yangming Xu, Guiyao Tie, Yongchao Chen, Guowen Zhang, Duanfeng Chu, Pan Zhou, and Lichao Sun. Libero-pro: Towards robust and fair evaluation of vision-language-action models beyond memorization. *arXiv preprint arXiv:2510.03827*, 2025.

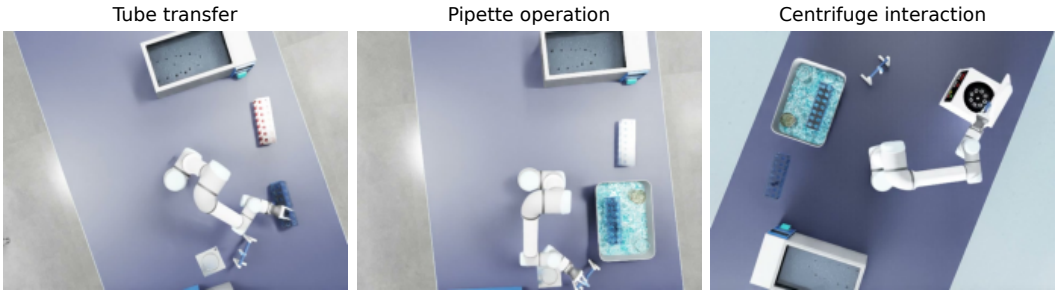

Figure 8: Example interactions from scripted training demonstrations: tube transfer, pipette use, and a centrifuge interaction. Top-camera views are shown; the policy also receives wrist images.

A Scene Construction and Randomization

Scene configurations register object types, concrete instances, placement slots, aliases, and semantic points. A task binds roles such as source tube, target rack, and target pipette to these instances. Semantic annotations include grasp/interaction frames, insertion axes, container points, placement regions, and articulation references. A new composition reuses existing task/checker implementations; a new manipulation type additionally requires its own criteria and expert policy. The runtime task suite is flat, while L1/L2/L3 organize the benchmark by procedural scope.

The base-scene configuration enables environment selection from the available environment children and robot-base yaw sampling in $[-1.57, 1.57]$ radians. A selected layout preset supplies equipment arrangements and configured local randomization. For example, one reference preset samples rack offsets of $[-0.08, 0.08]$ m in x and $[-0.05, 0.08]$ m in y, with $\pm 15^\circ$ yaw. The pipette's local offset is ± 0.05 m in both horizontal axes, also with $\pm 15^\circ$ yaw. Other equipment uses its own bounds; an incubator can rotate about the robot base instead of receiving independent translation. A tube can be initialized from a sampled rack-well point, and beakers follow the ice-container placement through dependent transforms.

Object sampling follows placement priorities and configured clearance constraints. This reference preset permits 20 scene attempts and 50 attempts per object, with 0.01 m horizontal clearance. A collection run selects the preset; within-preset episode sampling varies the permitted dimensions. Recorded episodes retain resolved values for restoration.

B Task Representation and Evaluation Details

Let $T = (V, E, \prec)$ be a task tree. Nodes V are task occurrences, edges E connect parents to children, and $\prec$ orders siblings. A node v has task type $\tau(v)$, materialized instruction $\ell(v)$, and occurrence path $p(v)$. Types enable reuse; paths distinguish uses within a sequence. The two tube visits in a mixing task therefore retain separate intervals and outcomes even when their language matches. The plasmid-transformation tree has 2 tube-transfer leaves, 6 pipette leaves, 3 water-bath leaves, 4 pouring leaves, and 4 incubator-placement leaves, totaling 19. Pouring and uprighting/return to the approach pose are separate occurrences.

Each leaf uses an ordered physical evaluation plan. Criteria describe object placement, articulation, grasp, and process geometry; their version-specific separation from handoff gates is described in Appendix G.2. Temporal criteria, such as retaining a grasp during a lift, apply over a defined interval. A condition remains pending while recovery is possible; completed stages retain their satisfaction. A composite stops at the first failed child and marks later children not reached. Policy-side memory and evaluator-side occurrence progress remain distinct: selecting a successor or reporting completion does not establish physical success.

L1 progress. For episode i, let $\mathcal{C}_i$ contain all required criteria, keyed by occurrence path, stage, and criterion name. Episode progress and its mean are

$$C_i = \frac{1}{|\mathcal{C}_i|} \sum_{c \in \mathcal{C}_i} z_{ic}, \qquad \overline{C} = \frac{1}{N} \sum_{i=1}^{N} C_i, \tag{1}$$

where $z_{ic} = 1$ for satisfaction retained from a completed stage or present in the final state of the active unfinished stage, and zero otherwise. Unreached criteria remain in the denominator; diagnostics are excluded and initially satisfied conditions may contribute. Archived evidence catalogs supply the full denominator. Pouring terminates at the tilted pouring endpoint: the source pickup, position tolerance of 0.08 m, and tilt of at least 60° remain required, but the two subsequent uprighting criteria are removed. Offline rescoring, without rerunning policies, changes the pour denominator from seven to five and promotes 20 of 288 pour outcomes to success.

L2/L3 progress. For the complete planned leaf-occurrence set $\mathcal{L}_i$, episode progress and its mean are

$$P_i = \frac{1}{|\mathcal{L}_i|} \sum_{\ell \in \mathcal{L}_i} \mathbf{1}[\ell \text{ succeeds}], \qquad \overline{P} = \frac{1}{N} \sum_{i=1}^{N} P_i. \tag{2}$$

For Root instruction, Hierarchical, and Oracle, each successful leaf contributes one. Unreached leaves contribute zero and remain in the denominator; repeated operations are distinct occurrences. The mean gives equal weight to each episode. Full plans are reconstructed and validated using archived configuration and occurrence evidence. Pouring has four leaves; L3 has 19. The task-success rate is the fraction of successful complete workflows, reported separately from progress.

Funnel conventions. L3 phase reach means entering its first native leaf, with the denominator fixed at all 18 episodes. Leaf-completion and real-robot milestone funnels instead count episodes that have completed each listed step. For a real task with K ordered, equally weighted steps, normalized progress p_i gives milestone count $\sum_i \mathbf{1}[p_i \geq k/K]$ at step k; success is $p_i = 1$ and mean progress is $N^{-1} \sum_i p_i$.

Execution protocol. Simulation policies act at 20 Hz, with at most 15 actions consumed per inference. The reported L1 campaign terminates a reached leaf immediately when its checker terminates and discards the remaining action suffix. The higher-level campaigns use successful-prefix completion: success reached within the selected prefix is latched, then the remaining selected actions finish before the task boundary, subject to the applicable non-scoring handoff gate. A failure, error, deadline, or cap can interrupt that tail and block downstream execution. No buffered actions cross leaves.

The simulation campaigns impose a per-leaf cap of 1,800 policy-action rows and no additional whole-episode action cap. Semantic checker deadlines remain active. L1 permits up to 45 seconds for reset stability; L2's source-bound reset settings are described in Appendix G.2. Simulation pauses during inference. There is one accepted execution-valid result per policy/scenario, not a best-of selection; invalid attempts are retained separately, and valid failures remain in the denominator. L3 starts from the restored observation at state step zero and empty memory, with no replay prefix, using the same 20 Hz control and 60 Hz physics. Its 45-second reset budget retains the source-bound Petri-dish settings.

C Task Inventory and Split Construction

Tables 8 and 9 give semantic-task allocations. A simulation task instantiates a semantic objective with a particular direction, source, or ordered sequence. An occurrence within that task is not a further dataset task. The inventories use Train / Normal / Preset-OOD / Reserved order; derived clips and VLM input variants do not add source episodes. L3 covers three SOP families; we detail plasmid transformation as the running example, with 81 Train and 18 Normal episodes: nine and two episodes, respectively, from each of the nine seen presets. Its Preset-OOD and Reserved counts are zero.

Table 7: Three-level source-demonstration inventory. Totals count level-specific episode records, not derived clips or alternate visual inputs. L1 and L2 together contain 4,181 records. L3 covers three SOP families with matched per-family collection; the L3 row and the total detail the plasmid-transformation family. Reserved episodes are excluded from training and reported evaluation.

Level	Semantic / simulation tasks	Train	Normal	Preset-OOD	Reserved	Total
L1	14 / 20	1,512	504	168	387	2,571
L2	8 / 13	1,088	344	128	50	1,610
L3 (plasmid transformation)	1 / 1	81	18	0	0	99
All levels						4,280

Table 8: L1 semantic-task inventory. Centrifuge removal includes placement back into the rack. Pipette, pouring, and water-bath labels denote the no-put-back objectives. A variant specifies a direction, source, or layout within a semantic task.

Semantic task	Variants	Train	Normal	Preset-OOD	Reserved	Total
Open centrifuge lid	1	108	36	12	3	159
Close centrifuge lid	1	108	36	12	3	159
Open incubator door	1	108	36	12	3	159
Close incubator door	1	108	36	12	3	159
Centrifuge insertion	1	108	36	12	0	156
Centrifuge removal	1	108	36	12	0	156
Pipette at beaker	2	108	36	12	44	200
Pipette at tube	1	108	36	12	0	156
Dish transfer between pads	4	108	36	12	84	240
Dish into open incubator	1	108	36	12	3	159
Tube transfer between racks	2	108	36	12	94	250
Tube transfer to/from ice	2	108	36	12	144	300
Pouring posture	1	108	36	12	3	159
Water-bath handling	1	108	36	12	3	159
Total	20	1,512	504	168	387	2,571

The nine seen presets for L1/L2 are `preset_ref_1_1` through `preset_ref_1_5` and `preset_ref_2_1` through `preset_ref_2_4`; the held-out preset is `preset_ref_3_1`. Reserved records include valid episodes outside selected training/test quotas. Metadata and Parquet row counts determine the episode inventory, and source-row lineage determines exported action counts.

L1's multi-variant groups comprise two beaker sources for pipetting, two rack-transfer directions, two rack/ice-container directions, and four pad-transfer variants. L2's Pipette group contains cell–tube, plasmid–tube, tube–cell, and tube–plasmid orders; its Mix liquid group contains cell–tube–plasmid–tube and plasmid–tube–cell–tube. Repeated tube interactions retain distinct occurrences. Across all four splits, L1 contains 6,441 native instruction rows and approximately 19.28 trajectory hours, with mean episode duration 26.99 seconds. Its Train split instead contains 836,674 action rows, 249 distinct root strings, 11.62 hours, and mean duration 27.67 seconds (median 26.95 seconds). The L2 leaf export contains 260 instruction texts and 8,856 videos; these counts reflect four-leaf pouring.

L2 task names and operational scope. Pipette denotes the ordered two-container sequence; Aspirate from beaker and Dispense to tube each include pickup, a container interaction, and returning the pipette. Mix liquid denotes the repeated four-visit sequence. These names express intended laboratory operations through manipulation-level proxies: the pipetting checkers assess tool motion and stage completion, not aspirated or dispensed liquid volume or mixing homogeneity. Display names do not change native task IDs, recorded instructions, or evaluation criteria.

Table 9: L2 semantic-task inventory. Return-inclusive objectives and ordered/repeated sequences are distinct from the L1 operation endpoints.

Semantic task	Variants	Train	Normal	Preset-OOD	Reserved	Total
Pipette	4	240	80	32	0	352
Aspirate from beaker	2	108	36	12	44	200
Dispense to tube	1	108	36	12	0	156
Mix liquid	2	200	48	24	0	272
Pouring + return	1	108	36	12	3	159
Dish into closed incubator	1	108	36	12	0	156
Dish out of closed incubator	1	108	36	12	0	156
Water bath + return	1	108	36	12	3	159
Total	13	1,088	344	128	50	1,610

C.1 Task-Level and Occurrence-Level Data Contracts

L1 is task-level, not primitive-only. Each L1 export episode retains the materialized root instruction throughout a complete selected-task demonstration. Of its 1,512 training sources, 432 have one leaf and 1,080 have two ordered leaves. Pickup–operation and pickup–placement are therefore valid L1 tasks. Internal leaf boundaries neither change the prompt nor split the export episode, and training chunks may span them.

Five simulation variants in four semantic groups redefine the final boundary for beaker/tube pipetting, pouring, and water-bath handling by omitting final tool/tube return placement. These are task definitions, not arbitrary truncations by the training adapter. L1 pouring is evaluated at the tilted pouring endpoint rather than after subsequent uprighting. Other L1 transfers retain destination placement; centrifuge-tube return is not a no-put-back variant.

Collection and language. Scripted experts supply simulation action demonstrations. Only successful, execution-valid Collection attempts are promoted; failed attempts are not additional demonstrations. A reset-stability barrier establishes row zero, and reset waiting is excluded from trajectory duration. Native records retain commands, states, instruction trees, resolved randomization, and checker evidence. Restored evaluation uses initial scenario values, not the source trajectory as online policy input. Real-robot demonstrations are collected separately through EVA-Client.

Training language is the materialized root or leaf text, not a fresh paraphrase from the exporter. Constrained language generation preserves targets, action order, and conditions; repeated wording may share a language index without merging demonstrations. The 249 L1 root strings and 260 L2 leaf strings are therefore vocabularies, not benchmark task counts. Language is not held out by split, so Preset-OOD is not a language-generalization test.

Split and source dependence. Episodes are allocated before generating clips, context pairs, or historical images; all derivatives inherit their source split. L1/L2 Normal shares preset identities with Train but uses different Native identities. Preset-OOD does not establish unseen-task or unseen-object-category generalization. For plasmid transformation, L3's explicit allocation assigns 81/18/0/0 Train/Normal/OOD/Reserved episodes and takes precedence over inherited Native split tags.

L1 and L2 share 674 Native $(\mathrm{run_id}, \mathrm{episode_id})$ identities: 432 Train, 144 Normal, 48 Preset-OOD, and 50 Reserved, without crossing splits. The corresponding no-put-back L1 tasks have different terminal semantics. Thus the 4,181 L1/L2 level-specific records are not independent acquisitions; the 99 plasmid-transformation records add complete SOP trajectories rather than treating L2 segments as new workflows.

Dense leaf supervision and versioning. A leaf is an occurrence without children, not a fixed tree depth or a single robot command. Closed-incubator placement has open/pickup/place/close leaves; its transfer parent is not supervised again over its children's actions. L2 has 324 three-leaf, 564 four-leaf, and 200 six-leaf training sources. Segments partition all original action transitions without gaps, duplication, or synthetic terminal rows, while the physical state carries across boundaries

Table 10: Training action exports. Root/leaf editions retain the same transitions under different supervision contracts. Duration is action rows divided by 20 Hz; cameras do not multiply trajectory time. L2 values use four-leaf pour segmentation.

Quantity	L1 root	L2 leaf	L2 root	L3 leaf
Native training sources	1,512	1,088	1,088	81
Derived episodes	1,512	4,428	1,088	1,539
Observation–action rows	836,674	1,076,412	1,076,412	389,914
Distinct instruction strings	249	260	313	—
Trajectory hours	11.62	14.95	14.95	5.42
Mean episode duration (s)	27.67	12.15	49.47	12.67

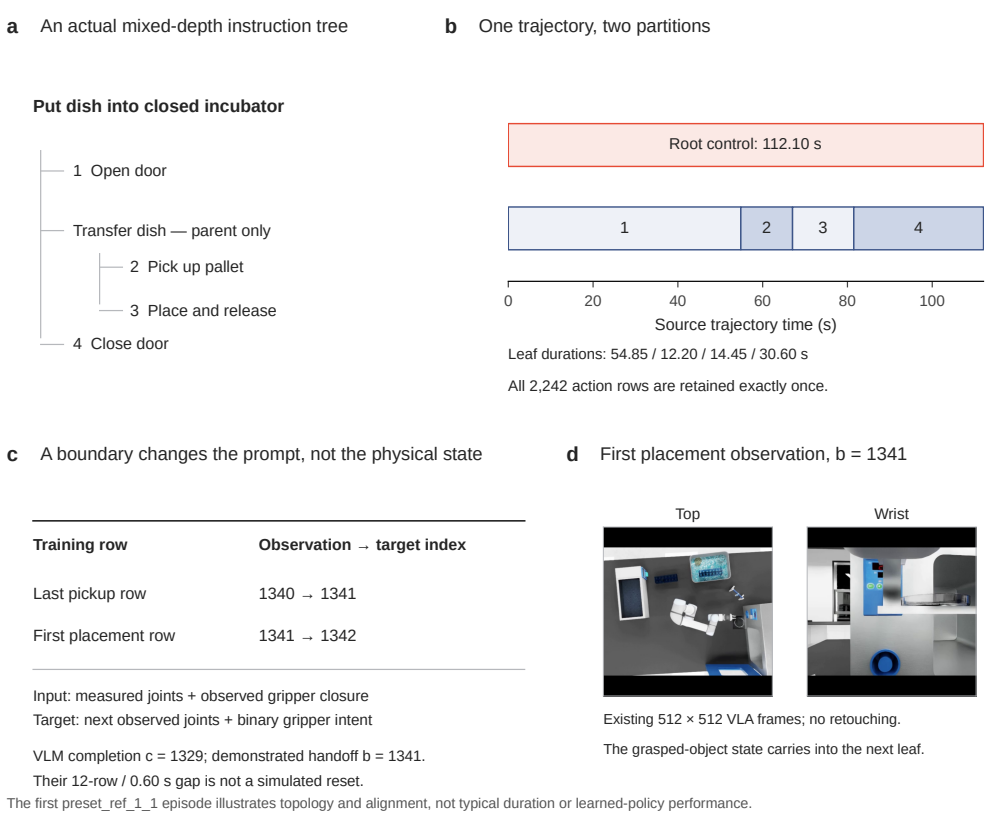

Figure 9: A scripted closed-incubator placement illustrates mixed-depth leaf extraction and exact action alignment. All 2,242 transitions are retained. The first placement input is the state left by pickup, not a reset. Pickup completion precedes demonstrated handoff by 12 observations (0.60 s), the gap that transition-window supervision targets. Images come from exported demonstrations, not policy rollouts.

(Figure 9). Under a 50-action horizon, 20.09% of leaf anchors and 4.95% of root anchors lack 50 real future targets; these are length-derived fractions, not loss-mask measurements.

The L2 leaf export contains 4,428 segments and 260 leaf strings: pouring separates each demonstration's operation and return at the recorded return-command onset, giving four pour leaves. Segments partition all 1,076,412 transitions. L3 uses 19 leaves per demonstration and supplies 1,539 leaf segments from its 81 plasmid-transformation Train episodes.

Validation scope. Source identities, ordered occurrence intervals, and instruction bindings connect Native, VLA, and VLM derivatives. The L2 export retains 390 identified inherited instruction-generation provenance exceptions, including 252 Train sources. Pour-segmentation changes are

validated together with their export bindings while preserving unchanged trajectory and media evidence. L3's Normal episodes are excluded from action training and normalization.

D Native Alignment and VLM Supervision

Action boundaries. The initial native row is a settled observation without a preceding command. Action row t has its pre-action observation at row $t-1$ and next-observed target at row t. Inclusive leaf action interval $[a, b]$ therefore pairs observations $[a-1, b-1]$ with targets $[a, b]$. The exported gripper action is binary command intent, while state retains the measured closed fraction. For an L2 training episode with n native rows, its segments jointly contribute $n-1$ pairs when all actions belong to instruction leaves. Segmentation does not reset the simulator. Each segment retains source identity, occurrence path, intervals, and artifact fingerprints. VLA images use aspect-preserving padding to 512×512; VLM images retain the unpadded 512×384 shape of resized 640×480 sources.

Paired context targets. Let c_t^- denote the outgoing context before a completed occurrence is appended and c_t^+ the context after the corresponding update. At the same completion observation o_t, training includes

$$(g, o_t, c_t^-) \mapsto u_{\text{update}}, \qquad (g, o_t, c_t^+) \mapsto u_{\text{hold}}, \tag{3}$$

where g is the root instruction and u is a structured update/retain target. The pair teaches whether visible completion has already been consumed; it is not a second physical transition.

Transition windows. For leaf k, completion observation c_k is the unique successful, execution-valid Collection check-run termination; handoff b_k is its final owned action's observation. The next VLA leaf begins from observation b_k, not $b_k + 1$. Sampling starts from the observation interval $[c_k, b_k]$. All rows are used when the interval has at most 12 observations; otherwise 12 evenly spaced integer rows include both endpoints. A window shorter than six rows is extended first by up to five post-boundary rows without entering the next transition core. Train and Reserved may additionally use up to two pre-completion rows. These targets are marked weak because they anticipate the checker boundary; Normal and Preset-OOD do not use them. If six rows remain unavailable, the available observations are retained. Adjacent windows do not overlap.

Each sampled window row produces both contexts in Equation 3. For a window of m selected rows, up to m ordinary holds are sampled from the leaf interior, excluding all transition windows and a one-row margin. Sampling is deterministic from source identity, occurrence path, and seed (default zero). The L2 Train export contains 45,322 records of each kind—ordinary hold, outgoing-context update, and updated-context hold—plus 1,088 initialization records. Weak pre-completion records are already included in the export; neither evaluation split uses weak targets. The first episode row supplies initialization. Final-leaf outgoing contexts append completion and select terminal state; updated contexts hold that terminal state without fabricating a post-episode trajectory.

Temporal inputs and counts. Current-only inputs contain current top/wrist images. The temporal variant adds one observation 15 steps earlier, ordered previous top/wrist then current top/wrist, and omits it when unavailable. Content-addressed images are shared across context pairs. L2 counts are 137,054 / 40,634 / 16,028 / 5,297 per variant, totaling 199,013, with 498,708 shared JPEGs across both variants. Alternative visual inputs do not double independent supervision. L3 adds 45,795 Train and 10,062 Normal examples per variant (55,857 total), derived only from the 99 complete plasmid-transformation trajectories, without L2 donors or fabricated earlier history. Its Train split contains 15,238 examples of each hold/update/context-pair type plus 81 initializations, including 324 weak records; Normal has no weak records. Both use complete same-episode instruction history. Together the L2 and L3 exports contain 254,870 records per visual variant.

Handoff distribution shift. Completion and expert handoff need not coincide: an update observation can precede the final demonstrated action owned by that leaf. An update target within this window can dispatch the next instruction before the state from which its VLA demonstration begins. Weak pre-completion examples add a further, explicitly marked anticipation.

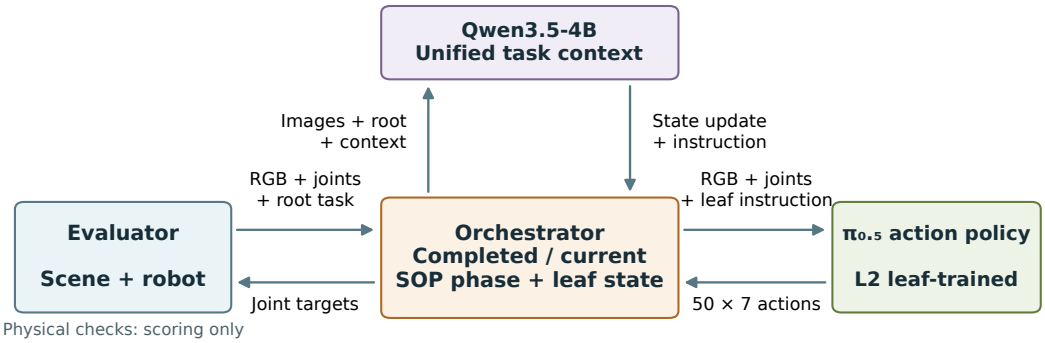

Figure 10: The BioLab-HVLA reference system. The orchestrator owns execution context and routes model calls. One Qwen model updates execution memory and the active instruction; a leaf-trained action policy executes it, with additional L3 adaptation for the SOP evaluation. Physical checker outcomes are used for scoring, not supplied to the learned planner.

E BioLab-HVLA Interface and Execution

At round k, let o_k be top/wrist images, q_k measured robot state, g the immutable root instruction, and c_k the stored context. The planner predicts a completed-memory append ΔM_k, state replacement $\widehat{s}_k$, and leaf instruction ℓ_k:

$$(\Delta M_k, \widehat{s}_k, \ell_k) = f_\phi(g, o_k, o_{k-1}, c_k). \tag{4}$$

The first round omits o_{k-1}; later rounds use the previous accepted round's synchronized observation. With a complete 15-step prefix this is exactly 15 control samples earlier. It is one earlier image pair, not fifteen frames; a changed execution interval requires reconsidering this temporal contract. Current-state decisions are grounded in o_k. L2 context contains completed instructions and the current leaf occurrence. L3 retains the same fields, using global indices 1–19 and stage-prefixed instructions to locate occurrences within the SOP. It does not instantiate separate nested phase and leaf planners.

Each query reconstructs a fixed system message and one user message rather than accumulating a chat transcript. Request IDs count control rounds, whereas occurrence indices advance only with task-state transitions. The L2 user message has a root `task` and `context` containing `Completed instructions` and `state`. Each occurrence has integer `index` and string `instruction`. Completed indices start at one and are consecutive; the current index follows. The output contains `append` (new completions), `replace` (new state or null for unchanged), and `vla` (executable instruction or null to stop). The VLA instruction must match the effective state. Initialization selects occurrence 1 without appending. Termination appends the final completion and selects the next index with a null instruction. For L3, memory strings use `stage name: bare leaf instruction`, whereas `vla` contains only the bare instruction. The fixed system/SOP text enumerates the five phases and distinguishes pouring, upright return, and rack placement. Memory starts empty. In the evaluated L3 service, a terminal-20 proposal is ignored without committing it or returning model-done; physical evaluation determines episode termination. No such proposal occurs in the reported 18 trials.

For example, pickup completion with current index 1 appends occurrence 1 and selects occurrence 2. The identical image with completion 1 already recorded instead requires empty append, null replacement, and the same occurrence-2 instruction. The orchestrator stages a structurally validated update, validates the returned action shape, dtype, and finite values, and commits context, observation cache, and response together only if the episode has not changed. Accepted L2 terminal transitions commit without an action-model call; L3 retains the terminal-proposal exception above. Hold transitions retain the occurrence but re-infer actions from the latest observation. Reset clears episode state and invalidates old in-flight results. A matching retry of the most recent committed request returns its cached response without advancing memory twice. Checker values, command

Table 11: Documented L1 training recipes. Batch size is 64 throughout. Horizons are predicted actions, not the 15-action consumed prefix. InternVLA updates comprise two stages with a fresh optimizer at the transition.

Policy / initialization	Train episodes	Updates	Trainable components	Horizon
$\pi_{0.5}$ / `pi05_base`	1,512	20,000	Full model	50
DM0.5 / DM05	1,512	20,000	VLM/action stack; embeddings frozen	50
X-VLA / X-VLA-Pt	1,512	30,000	Heads/prompts first; then backbone	30
GR00T N1.7 / N1.7-3B	1,512	20,000	Action stack; vision/language frozen	40
SmolVLA / smolvla_base	1,437	30,000	Action expert/projections, state projection	30
InternVLA-A1.5 / A1.5-base	1,437	1,000+29,000	Action adapter; then full action-only fine-tuning	50

labels, occurrence paths, and ground-truth successors remain provenance or evaluation information, not planner inputs.

For non-null ℓ_k, the action policy predicts

$$A_k = \pi_\theta(o_k, q_k, \ell_k), \qquad A_k \in \mathbb{R}^{50\times 7}. \tag{5}$$

The first 15 rows are consumed at 20 Hz, covering 0.75 simulated seconds for a fully executed nonterminal prefix. Internal arm deltas are relative to the same current state for all 50 predictions, not differences between successive predicted states. They are inverted to absolute targets; the gripper is not offset by measured closure. Binary training intent does not force the action model's continuous gripper output to be binary. The successful-prefix rule is detailed in Appendix B. An accepted L2 model-issued stop returns no action: completed physical outcomes are retained and unfinished or unreached operations remain incomplete. L3 termination follows the physical evaluator as specified above.

F Training Configurations

F.1 L1 Baseline Recipes

All six runs start from pretrained models and use the same corrected L1 release, four A100 80 GB GPUs, and global batch 64. They do not share an optimization recipe (Table 11). SmolVLA and InternVLA reserve the 75 episodes with derived index modulo 20 equal to 19, leaving 1,437 training episodes and 795,163 training rows. This internal holdout comes from the published Train split, not the benchmark Normal/OOD scenarios. The other four use all 1,512 training episodes. Final checkpoints are used for all six policies.

$\pi_{0.5}$. Full BF16 fine-tuning uses EMA decay 0.99, AdamW with $\beta = (0.9, 0.95)$, $\epsilon = 10^{-8}$, weight decay 10^{-10}, and gradient clipping 1.0. The learning rate warms up for 1,000 updates to 2.5×10^{-5}, then follows a 30,000-update cosine schedule toward 2.5×10^{-6}, stopped after 20,000 updates. FSDP uses four devices and seed 42. Top/wrist images are resized to 224×224; the unused third view is zero-filled and masked. State/action are padded internally to 32 dimensions. Arm actions are relative to the current state, whereas gripper intent stays absolute. Quantile statistics are computed after the relative-arm transform from 4,096 frames. The final checkpoint is step 19,999 (zero-based).

DM0.5. Full SFT updates the Gemma3/action stack except token embeddings and the tied language-model head; no LoRA is used. AdamW has $\beta = (0.9, 0.95)$, $\epsilon = 10^{-8}$, weight decay 10^{-10}, and clipping 1.0. Peak learning rate is 2.5×10^{-5} after 1,000 warmup updates, with cosine decay to one tenth over 20,000 updates. Training uses BF16/TF32, FSDP full sharding, activation checkpointing, and seed 42. Images are processed at 448×448 with crop, small rotation, and color augmentation. Quantile-normalized current-relative arm targets and absolute gripper targets

use flow-matching loss masked to seven physical channels. The last 50 anchors per episode are dropped, leaving 761,074 windows. The final checkpoint is update 20,000.

X-VLA. The Florence2/action-transformer model uses a seven-dimensional joint adaptation padded to 20 channels. The first 1,000 updates train soft prompts and action heads only, followed by unfreezing and 2,000 updates of backbone warmup. AdamW uses $\beta = (0.9, 0.95)$, weight decay zero, and clipping 1.0. Peak rates are 10^{-4} for core/heads and 10^{-5} for VLM/soft prompts, decaying after update 3,000 to one tenth. BF16 DDP runs with rank-dependent seed $0 + \mathrm{rank}$. Images use 224×224 resize, color jitter, and ImageNet normalization. The auto joint path uses current-relative arm deltas in radians without dataset state/action normalization. Its objective is 100 times clean-target reconstruction MSE over seven channels, not the same loss as the flow-matching baselines. Dropping the final 30 anchors per episode leaves 791,314 windows. The final checkpoint is update 30,000.

GR00T N1.7. The diffusion model, projector, and action stack are trained while language and vision backbones remain frozen. BF16/TF32 and DeepSpeed ZeRO stage 2 use seed 42. AdamW has peak learning rate 10^{-4}, weight decay 10^{-5}, and clipping 1.0, with 1,000 warmup updates and a 20,000-update cosine schedule. The data pipeline crops to 244×244 and targets 224×224 images. Arm actions are relative and gripper targets absolute. The 40-action horizon matches this checkpoint; the sampler uses shuffle, shard size 1,024, and episode sampling rate 0.1. The final checkpoint is update 20,000.

SmolVLA. The vision/language backbone is frozen; the action expert, action projections, and state projection are trained. AdamW uses $\beta = (0.9, 0.95)$, $\epsilon = 10^{-8}$, weight decay 10^{-10}, and clipping 10. A 1,000-update warmup reaches 10^{-4}, followed by cosine decay to 2.5×10^{-6} at update 30,000. BF16 DDP uses seed 1000. Two images are padded to 512×512 without extra dataset image augmentation. Train-only mean/std statistics follow the current-anchor relative-arm transform. Dropping the last 30 anchors from each training episode leaves 752,053 windows. Importantly, the actual training loss averages flow-matching error over all 32 internal channels: the 25 padding channels are *not* masked. This implementation difference is retained in the reported baseline, not retrospectively described as a seven-channel masked objective. The final checkpoint is update 30,000.

InternVLA-A1.5. The Qwen3.5-based model uses action-only flow matching, with no video-generation model or auxiliary video/VQA/subtask losses. Stage 1 trains the action adapter for 1,000 updates; stage 2 initializes from those weights and trains the unfrozen model for 29,000 additional updates with a new optimizer/scheduler. AdamW uses $\beta = (0.9, 0.95)$, $\epsilon = 10^{-8}$, weight decay 0.01, and clipping 1.0. Peak learning rates are 2.5×10^{-5} and 10^{-5}, warmups 100 and 1,000 updates, and cosine endpoints 2.5×10^{-6} and 10^{-6}, respectively. BF16 DDP uses seed 1000; gradient checkpointing is enabled in stage 2. Images are 224×224 and state is tokenized. Train-only mean/std normalization follows current-relative arm conversion. All 795,163 training anchors are retained; padded future timesteps and nonphysical channels are masked. This differs from dropping incomplete windows, and even a terminal anchor retains its current source-action target. The final checkpoint is stage-2 update 29,000, or 30,000 cumulative updates.

Comparability and artifact binding. All arm deltas above refer to the same current observation throughout a predicted chunk, not sequential differences. Different horizons, tail policies, loss masks, sampling, and frozen components make nominal updates or epochs insufficient measures of matched supervision. Training losses are not comparable policy scores.

F.2 L2 Hierarchy and Higher-Level Controls

L2 visual coordinator. Qwen3.5-4B is adapted on the history-augmented L2 Train records for one epoch, totaling 2,074 optimizer updates at global batch 64. Language-side all-linear LoRA uses rank 32, alpha 64, and dropout 0.05; vision and aligner are frozen. Learning rate is 10^{-4} with cosine scheduling and 5% warmup. Assistant-only supervision uses the non-thinking template and excludes empty thinking supervision. The saved adapter contains 64,929,792 parameters.

The adapter at update 2,000 is selected on a frozen development subset of 26 Normal episodes (3,107 requests) using paired correctness and preset tie-breaking rules; the 2,000 and 2,074 checkpoints tie on these development metrics. Preset-OOD is excluded from selection. Generation is greedy with temperature zero, seed 42, maximum 256 output tokens, and context limit 4,096.

Original L2 leaf executor. The L2 $\pi_{0.5}$ executor uses full fine-tuning, not a LoRA variant, for 20,000 updates at global batch 64 and FSDP degree four, selecting checkpoint 19,999. Top maps to the base view, wrist to the left-wrist view, and a zero third view is masked. Standard preprocessing resizes/pads to 224×224, and checkpoint-specific quantile normalization follows current-relative arm conversion. The internal action dimension is 32, exported as a finite float32 50×7 block using ten flow-matching inference steps and token budget 200. The action service restores arm offsets but does not add the measured gripper state, clip, or threshold that seventh output.

Root instruction model. The root-conditioned $\pi_{0.5}$ checkpoint is update 19,999 of a 20,000-update run on the L2 root export, which retains 1,088 complete demonstrations and 1,076,412 action transitions; the leaf export contains 4,428 segments over the same transitions.

F.3 SOP-Level Adaptation

Planner. The L3 Qwen3.5-4B adapter uses only the 45,795 history-augmented records from 81 plasmid-transformation Train trajectories; no L2 examples or cross-episode donor histories are mixed in. It is trained for two epochs (1,432 updates) with global batch 64: four devices, per-device batch four, and four-step gradient accumulation. BF16 and DeepSpeed ZeRO stage 2 are used with fused AdamW, cosine scheduling, learning rate 10^{-4}, 5% warmup, seed 42, maximum sequence length 4,096, and LoRA rank/alpha/dropout 32/64/0.05. The fixed system message and full SOP accompany complete same-episode memory. Current/history images use the same 15-step temporal offset as L2. The saved update-1,432 adapter is evaluated.

Executor. The L3 executor starts from the L2 leaf model's EMA parameters, with a fresh optimizer and schedule. Full BF16 fine-tuning uses the 1,539 plasmid-transformation Train segments (389,914 frames), 5,000 new updates, batch 64, four-device FSDP, 16 loader workers, and seed 42. Learning rate warms up over 500 updates to 10^{-5} and follows a 5,000-update cosine schedule toward 10^{-6}. AdamW uses $\beta = (0.9, 0.95)$, $\epsilon = 10^{-8}$, weight decay 10^{-10}, clipping 1.0, and EMA 0.99. Evaluation uses the final update-5,000 checkpoint and new Train-only quantile statistics, not the L2 normalization asset. All 389,914 training anchors and their 50-action windows enter normalization; window tails repeat the current leaf's final action without crossing into another leaf. The data reader discards the temporal padding mask, so repeated targets and internal padding channels remain in its loss. Normal is excluded from this action training and normalization.

F.4 Real-Robot Fine-Tuning

Demonstrations and initialization. Both real-robot models use the same 60 manually accepted demonstrations: 20 incubator-opening, 20 incubator-closing, and 20 complete water-bath/pour/return trajectories. Opening and closing are separate collection tasks, combined in the evaluation workflow. The export contains 59,372 frames and 120 top/wrist videos at 30 Hz; all 60 episodes enter training, without a held-out demonstration subset. The original 1280×720 videos are retained and padded/resized to 224×224 by the model reader. State/actions are seven-dimensional joint/gripper values; the EVA gripper-openness convention is converted once to the model's closure convention, while six joint angles remain in radians.

Direct $\pi_{0.5}$ control starts from official `pi05_base` parameters. BioLab-HVLA initializes its executor from the simulation L2 checkpoint at update 19,999 and fine-tunes a lightweight Qwen planner on manual stage annotations of the collected demonstrations (20 per task). Both action models use fresh optimizers and independently computed real-data normalization. The final update-10,000 checkpoints are used for deployment.

Common optimization recipe. Each model receives 10,000 full-model updates with global batch 64, four-device FSDP on A100 80 GB GPUs, 16 loader workers, and seed 42. AdamW uses $\beta =$

Table 12: Autoregressive evaluation of the L2 VLM (%). Exact match compares the complete parsed JSON, including instruction text; paired accuracy requires both old/update and new/hold records at the same observation to be correct.

Split	Model	Requests	Schema	Protocol	Exact JSON	Paired
Normal	Base	39,083	7.75	6.63	3.95	0.00
Normal	Adapter 2,000	39,083	100.00	100.00	92.84	79.45
Preset-OOD	Base	15,563	9.55	8.22	4.99	0.00
Preset-OOD	Adapter 2,000	15,563	100.00	99.99	93.20	80.37

Table 13: L1 operation execution. Each policy is evaluated on the same 504 Normal and 168 Preset-OOD scenarios. Success is the complete task outcome; progress is mean required-criterion coverage over all evaluated episodes.

Policy	Normal (504 scenarios)			Preset-OOD (168 scenarios)		
	Successes	Success (%)	Progress (%)	Successes	Success (%)	Progress (%)
$\pi_{0.5}$	179	35.52	47.95	44	26.19	42.02
DM0.5	147	29.17	43.95	27	16.07	32.05
X-VLA	73	14.48	27.11	13	7.74	19.56
GR00T N1.7	29	5.75	15.12	12	7.14	16.99
SmolVLA	17	3.37	10.09	7	4.17	10.64
InternVLA	12	2.38	6.23	2	1.19	4.60

$(0.9, 0.95)$, $\epsilon = 10^{-8}$, weight decay 10^{-10}, clipping 1.0, and EMA 0.99. A 1,000-update warmup reaches 2.5×10^{-5}, followed by a 10,000-update cosine schedule toward 2.5×10^{-6}. Quantile statistics use all 59,372 training frames and their 50-action future windows, clamped within each episode. Six arm targets are relative to the current observation; the gripper remains absolute. The internal dimension is 32 and the exported action chunk is 50×7. The interface converts closure back to EVA openness for execution.

F.5 Offline Structured-Prediction Evaluation

Table 12 evaluates generated outputs, not teacher-forced token accuracy. The unadapted model is subject to the same strict contract, without removing Markdown fences or repairing malformed answers. Its gap to the adapter therefore combines format/protocol adaptation with task learning, not visual reasoning alone. For the selected model on Normal, outgoing-context accuracy is 79.45%, ordinary-hold accuracy 99.08%, and updated-context accuracy 100%; aggregate exact match obscures the harder transition decisions. Inputs use demonstration images and teacher-constructed memory.

G Detailed Results and Rollout Analysis

L1 operation breakdown. Table 13 gives success counts and aggregate coverage; Figure 11 and Tables 2–3 show per-task results. After pouring-endpoint rescoring, the six policies yield 562 successes and 3,470 failures. Every successful episode has 100% criterion coverage, and no unsuccessful episode reaches 100% in this campaign.

$\pi_{0.5}$ closes the incubator door in 29 of 36 Normal scenarios but opens it in ten. DM0.5 succeeds in 33 opening and 23 closing trials. Sixteen of SmolVLA's 17 Normal successes are door/lid interactions; all 12 InternVLA Normal successes are in this group. $\pi_{0.5}$ reaches 38.43% coverage on Normal centrifuge insertion despite zero complete successes. DM0.5 and SmolVLA have zero Normal water-bath successes but 49.38% and 11.11% coverage, respectively. These contrasts illustrate why overall ranking and terminal success alone are insufficient diagnostics.

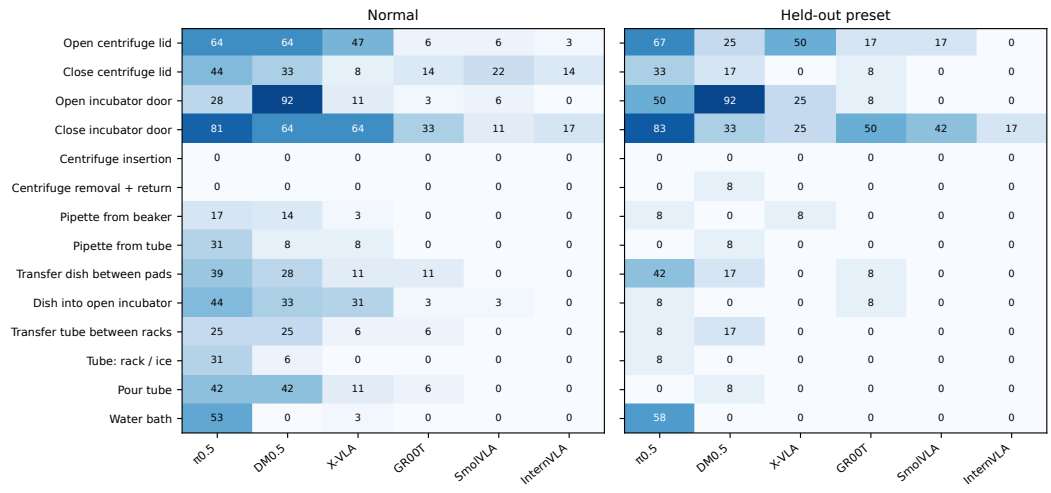

Figure 11: L1 success rates (%) by semantic task, with 36 Normal and 12 held-out scenarios per cell. Both panels use the same color scale. Labels retain the operation endpoints specified in the inventory.

G.1 L2 Task Outcomes and Physical Progress

All three methods cover 344 Normal and 128 Preset-OOD scenarios. Root instruction completes 14/472 tasks (2.97%) with 18.66% mean leaf completion; Hierarchical completes 32/472 (6.78%) with 28.05% progress; Oracle completes 40/472 (8.47%) with 27.31% progress. Overall values average episode outcomes, rather than assigning equal weight to simulation tasks. The per-task results in Tables 4–5 should therefore be interpreted together with their sample counts: the Pipette group has 80/32 Normal/OOD trials, Mix liquid has 48/24, and each remaining semantic group has 36/12. Individual simulation variants have as few as six OOD trials.

Four-leaf pouring. Pouring comprises four leaves: pickup, pouring, uprighting and return to the pre-pour approach pose, and final rack placement. Both systems use the same leaf-trained $\pi_{0.5}$ checkpoint and the same 48 sources. Hierarchical receives the official root instruction and predicts local commands through Qwen; Oracle receives the native current leaf's official instruction. Table 15 counts physical leaf completions, not predicted instruction switches. Hierarchical attains greater completion at each milestone in this subset. In both systems, the decline after pouring identifies an important distinction between performing the operation and leaving a state suitable for subsequent manipulation.

Progress reconstruction. Root progress is recomputed offline from its 472 original accepted outcomes: each successful evaluation-leaf occurrence contributes one and every planned leaf remains in the denominator. Per-episode fractions are averaged, yielding 22.02% Normal and 9.64% OOD. The archived per-leaf results and an independent reconstruction from condition evidence agree for all 472 episodes. There are 323 successful occurrences among 1,840 planned occurrences; their pooled ratio is *not* the reported 18.66% episode mean. All 14 successful episodes complete their full plan, and no failed episode reaches 100%. The calculation retains the original trajectories and success labels.

G.2 L2 Evaluation Protocol and Handoff Gates

Semantic success and physical handoff. For the later Hierarchical/Oracle evaluations, object, door, and process geometry determine leaf success. Measured gripper state is a separate, non-scoring handoff gate: open fraction must exceed 0.6 for release and closed fraction 0.3 for holding. The gate is checked at the end of the selected action prefix. Failure to pass it continues the current native leaf without revoking semantic credit, forcing a gripper command, or extending the action budget. Pure tool-center lift/retreat and exact close-command requirements are removed where they do not define

Table 14: L2 simulation-task detail. Cells show success rate above mean leaf completion (%). N/O denote Normal/Preset-OOD. Source counts and version boundaries are unchanged from the semantic aggregates.

Simulation task	N/O trials	Root instruction		Hierarchical		Oracle	
		N	O	N	O	N	O
Cell → tube	20/8	0.00	0.00	0.00	0.00	0.00	0.00
		27.50	0.00	16.25	21.88	21.25	3.13
Plasmid → tube	20/8	0.00	0.00	0.00	0.00	0.00	12.50
		30.00	12.50	32.50	18.75	18.75	21.88
Tube → cell	20/8	15.00	0.00	5.00	0.00	10.00	0.00
		28.75	6.25	20.00	15.63	25.00	21.88
Tube → plasmid	20/8	0.00	0.00	0.00	0.00	10.00	0.00
		18.75	6.25	16.25	6.25	32.50	18.75
Aspirate from cell beaker	18/6	16.67	0.00	16.67	0.00	27.78	0.00
		27.78	11.11	31.48	33.33	42.59	38.89
Aspirate from plasmid beaker	18/6	0.00	0.00	11.11	0.00	5.56	0.00
		22.22	16.67	35.19	11.11	33.33	22.22
Dispense to tube	36/12	5.56	0.00	13.89	16.67	30.56	16.67
		18.52	5.56	36.11	44.44	43.52	38.89
Mix liquid: cell first	24/12	0.00	0.00	4.17	0.00	0.00	0.00
		12.50	4.17	21.53	16.67	13.89	20.83
Mix liquid: plasmid first	24/12	0.00	0.00	0.00	0.00	4.17	0.00
		12.50	1.39	17.36	12.50	17.36	8.33
Pour tube	36/12	2.78	0.00	13.89	8.33	5.56	8.33
		25.93	2.78	44.44	16.67	27.08	16.67
Dish into closed incubator	36/12	11.11	0.00	8.33	0.00	0.00	0.00
		26.39	16.67	36.81	37.50	32.64	33.33
Dish out of closed incubator	36/12	0.00	0.00	0.00	0.00	0.00	0.00
		9.03	8.33	16.67	20.83	18.06	20.83
Water bath	36/12	2.78	0.00	22.22	8.33	30.56	8.33
		30.56	33.33	50.00	19.44	46.30	19.44

Table 15: Physical completion funnel for the revised L2 pour task. Counts are over 48 sources per method (36 Normal, 12 OOD); percentages average completed leaves out of four. Upright return is distinct from rack placement.

Method	Pickup	Pour	Upright return	Rack placement	SR (%)	Progress (%)
Hierarchical	33	24	9	6	12.50	37.50
Oracle	29	11	4	3	6.25	24.48

the operation; object lift, object–tool distance, insertion/extraction, and process geometry remain. This gate constrains native execution, not the planner's independent memory updates. Official source instructions are bound by full task path; missing or ambiguous bindings are rejected rather than replaced with template or demonstration-time switches.

Accepted outcomes and reset conditions. Each method retains one execution-valid outcome per source. Invalid attempts are excluded from task-success denominators and supplemented independently; valid successes and failures are not rerun for best-of selection.

Each pour evaluation initially accepts 40 outcomes and supplements eight reset-invalid sources, reaching 48/48 unique valid sources. The reset budget is 45 seconds.

Offline versus closed-loop decisions. The L2 coordinator's 92.84%/93.20% Normal/OOD exact-JSON scores are evaluated on demonstration-derived images and teacher-constructed memory (Appendix F.5). In the hierarchical campaign, logged premature model stops and instruction/checker mismatches demonstrate why generated completion cannot replace physical scoring. A mismatch alone is not a causal diagnosis: both stale and premature instructions require alignment with actions and physical evidence.

Table 16: L3 native leaf-entry and physical-completion counts, each out of 18 autonomous episodes. Zero entry denotes an unattempted leaf. Completed-leaf counts, not reached-leaf counts, determine progress.

Index	SOP phase	Leaf occurrence	Entered	Completed
1	Tube transfer	Pick up tube	18	13
2		Place tube in ice rack	13	13
3	Pipetting	Pick up pipette	13	7
4		Pipette cell beaker	7	4
5		Pipette tube (first visit)	4	0
6		Pipette plasmid beaker	0	0
7		Pipette tube (second visit)	0	0
8		Return pipette	0	0
9	Water bath	Pick up tube	0	0
10		Water-bath handling	0	0
11		Return tube to ice rack	0	0
12	Pouring	Pick up tube	0	0
13		Pour into dish	0	0
14		Upright and return to approach	0	0
15		Place tube in ice rack	0	0
16	Incubator placement	Open incubator	0	0
17		Pick up dish/pallet	0	0
18		Place dish/pallet in incubator	0	0
19		Close incubator	0	0

G.3 Context-Dependent Failures and SOP Progress

The autonomous L3 evaluation on plasmid transformation completes 37 of 342 planned leaf occurrences. Five episodes complete zero leaves, six complete two, three complete three, and four complete four. Table 16 separates physical entry from completion. Phase entry across the five SOP phases is therefore 18, 13, 0, 0, 0; later phases receive no execution evidence. The model's final memory index is 1 in five episodes, 3 in five, 5 in three, 7 in two, and 8, 15, or 16 in one each. These indices are current predicted occurrences, not counts of physically completed leaves. No terminal-20 proposal is observed.

All 18 sources have one valid autonomous outcome, without invalid attempts or supplementation. They produce 2,651 inference requests and 39,765 policy actions, consuming 15 actions per request.

G.4 Real-Robot Scoring Details

Incubator progress is 0, 0.5, or 1 for initial state, opening, and closing. Tube progress is the completed fraction of four milestones: pickup, water-bath handling, pouring into the beaker, and rack return. The above-beaker milestone has identical completion counts to water-bath handling in this cohort and is not separately scored.

Existing five-stage annotations are rescored as $0.2 \mapsto 0.25$, $0.4, 0.6 \mapsto 0.5$, and $0.8 \mapsto 0.75$, with 0 and 1 unchanged; executions and success outcomes are unchanged. Episode means are $8.75/15$ (58.33%) and $10.25/15$ (68.33%) for $\pi_{0.5}$ and BioLab-HVLA; equivalently, divide summed milestone counts by 15×4. Episodes are equally weighted.