IN-MANU: PROBING THE CAPABILITIES AND SELF-IMPROVEMENT OF FRONTIER AGENTS IN ROBOTIC MANIPULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Frontier multimodal models can act as robot policies without robot-specific training: they interpret camera images and task instructions and issue end-effector commands directly. The extent of this capability relative to post-trained embodied foundation models, and the extent to which such an agent can improve it from its own experience, have not been characterized. We present In-MANU, a study of frontier agents in robotic manipulation that evaluates GPT-6-Astra on the 26 test tasks of the EBench mobile manipulation benchmark and compares it with seven post-trained vision–language–action and world-action policies. With a single annotated demonstration per task, the agent ranks second among the eight systems. It attains the highest success rate on short-horizon tasks and exhibits exploration, recovery from disrupted subgoals and correction after failed attempts within an episode, but it ranks below most post-trained policies on long-horizon and tabletop dexterous-and-precise tasks, which require precise alignment, physical contact and bimanual coordination. We further examine two means of reducing this gap. Access to a post-trained policy as a callable tool increases success only on tasks that the policy solves on its own, and the agent's delegation does not correspond to the competence of the policy. A self-improvement loop, in which a critic instantiated from the same model revises the agent's playbook and harness on the basis of execution feedback, alters the agent's strategy without a statistically significant improvement in test success, both for GPT-6-Astra and, in a smaller replication, for Claude Opus 5.5; the critic's episode-level motion-error reports coincide with measured deviations, but its aggregate account attributes all failure modes to execution.These findings suggest that physical execution reliability remains a major bottleneck for frontier agents under the evaluated control interface.

1 INTRODUCTION

Transferring knowledge from large pretrained models has become a principal approach to general-purpose robot control (Firoozi et al., 2023). Vision–language–action models (VLAs) adapt the semantic knowledge of vision–language models to visuomotor control (Ma et al., 2024; Kim et al., 2024; Black et al., 2025; Physical Intelligence et al., 2025b; Ma et al., 2026), whereas world action models (WAMs) exploit the physical and temporal priors of video generative models (Shen et al., 2026; Kim et al., 2026; Li et al., 2026a; Yuan et al., 2026b; Wang et al., 2026f; Cai et al., 2026). Frontier multimodal models constitute a third approach that requires no robot-specific training. GPT-6-Astra (OpenAI, 2026), for example, can act as a robot policy with frozen weights (Zhang et al., 2026d), either by interpreting camera images and instructions to issue end-effector commands directly (Jia et al., 2026; Lou et al., 2026) or by invoking tools through a harness (Chen et al., 2026), including frozen learned policies (Zhang et al., 2026e; RLinf, 2026).

The capability of such agents relative to post-trained embodied foundation models has not been characterized. It is not known on which manipulation tasks an agent of this kind outperforms post-trained policies, on which it does not, and whether its limitations arise from task interpretation, from planning over long procedures, or from the precision and coordination of physical execution. A related question is whether these limitations can be reduced, either by delegating execution to a post-trained policy or through improvement from the agent's own experience. Recent harness and

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.

self-improvement systems accumulate rules, skills and tools from experience (Gao et al., 2025a; Wang et al., 2026b; Ding et al., 2026; Li et al., 2026b; Tang et al., 2026; Xiao et al., 2026); several of them, however, depend on human feedback or predefined curricula (Xiao et al., 2026; Li et al., 2026b), or report most of their gains on new instances of the tasks used for improvement (Ding et al., 2026; Wang et al., 2026c; Zhang et al., 2026e).

We present In-MANU, a study of frontier agents in robotic manipulation conducted on EBench (Gao et al., 2026), whose capability annotations and controlled scene variations permit analysis both by capability and by individual case. GPT-6-Astra, operating through direct end-effector control with one annotated demonstration in context, is evaluated on all 26 test tasks and compared with seven post-trained VLA and WAM policies. In addition to aggregate success, the analysis considers partial task completion and failure patterns, differences in capability across execution demands, adaptive behavior within episodes, safety-relevant behavior and compositional generalization, and each finding is supported by the recorded trajectories. The results show that the agent interprets tasks flexibly but executes them less reliably than post-trained policies. It ranks second among the eight systems, attains the highest success rate on short-horizon tasks, and exhibits exploration, recovery from disrupted subgoals and correction after failed attempts. On long-horizon and tabletop dexterous-and-precise tasks, however, it ranks below most post-trained policies, and the recordings indicate that precise alignment, physical contact and bimanual coordination limit its performance on these tasks.

Two means of reducing this gap are then examined. In the first, a post-trained policy, $\pi_{0.5}$, is made available to the agent as a callable tool. In the second, which combines existing harness and self-improvement frameworks (RLinf, 2026; Xiao et al., 2026), the agent starts from these two basic tools, and after each round a critic instantiated from the same model reviews the executed episodes and revises the agent's playbook and harness on the basis of execution feedback, without task-specific instructions apart from one question from the authors about the repeated failures of two training tasks. The loop is trained on ten benchmark tasks and evaluated on these tasks and on seven task categories that it never encounters, with GPT-6-Astra and, in a smaller replication, with Claude Opus 5.5 (Anthropic, 2026).

The contributions and principal findings of this work are as follows:

- **A capability analysis of a frontier agent as a robot policy**. The agent's advantage is concentrated on short procedures, and its rollouts exhibit adaptation within an episode, whereas most post-trained policies achieve higher success on long-horizon and tabletop dexterous-and-precise tasks. The recorded trajectories point to precise contact and bimanual coordination as important limitations.
- **Coupling with a post-trained policy**. Access to $\pi_{0.5}$ as a tool increases success only on tasks that $\pi_{0.5}$ frequently solves on its own and yields no significant overall gain, and the proportion of execution that the agent delegates does not correspond to the competence of the policy.
- **Self-improvement from execution feedback**. Neither evaluated final version shows a statistically significant overall gain on the test tasks. The loop changes how the agent divides execution between its own control and $\pi_{0.5}$, but no learned rule addresses the millimeter tolerances of high-precision tasks. The motion errors that the critic states in individual reviews coincide with measured deviations; the critic, however, detects only half of the large deviations and, when asked for an aggregate account, attributes every failure mode to execution, which contradicts its own reviews.

2 Related Work

Frontier models as robot policies. Generalist robot control has largely been pursued by post-training large pretrained models: VLAs adapt vision–language backbones to visuomotor control (Kim et al., 2024; Black et al., 2025; Physical Intelligence et al., 2025b), and WAMs build on video pretraining to model dynamics and actions jointly (Ye et al., 2026; Kim et al., 2026; Yuan et al., 2026b). An alternative line of work leaves the weights of a frontier model unchanged and connects the model to the robot through an interface, such as programs that compose perception and control A$\pi_{0.5}$s (Liang et al., 2022), semantic action units that embodiment-specific interpreters ground in motion (Chen et al., 2026), or end-effector and joint commands issued by the model itself (Jia et al.,

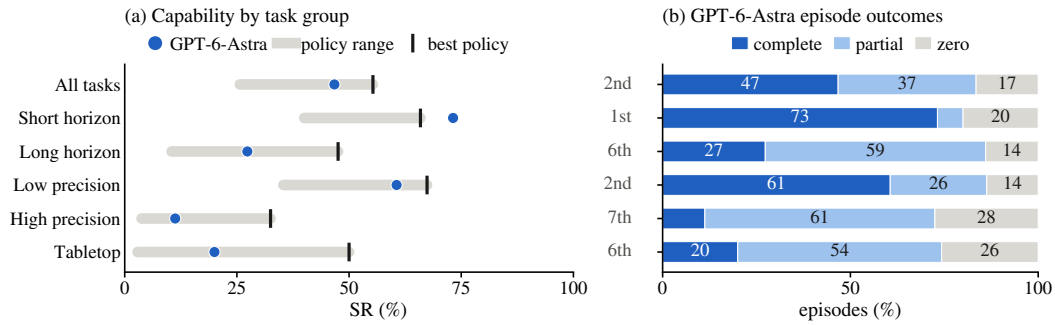


Figure 1: GPT-6-Astra on the 26 EBench test tasks by task group. (a) Success rate (SR) of GPT-6-Astra, range of the seven post-trained policies and best policy; right: rank of GPT-6-Astra among the eight models. (b) Share of GPT-6-Astra's episodes ending in complete success, partial credit or zero Score.

2026; Lou et al., 2026); GPT-6-Astra has also been evaluated directly as a robot policy (Zhang et al., 2026d). Hybrid harnesses allow the model to call a frozen VLA as one tool among scripted or analytic primitives (RLinf, 2026; Zhang et al., 2026e), with a memory that determines when the VLA is invoked and how control passes between subtasks (Xu et al., 2026). Each of these studies evaluates a single system configuration. The present work instead characterizes the capability of a frontier agent relative to post-trained policies on a common benchmark (Gao et al., 2026), stratified by execution demand, and uses the agent's delegation to $\pi_{0.5}$ as a probe of whether it recognizes where the policy is competent.

Self-improvement and its evaluation. Agents can improve without weight updates by accumulating verbal reflections (Shinn et al., 2023) or curated playbooks (Zhang et al., 2026c), by rewriting their own code (Zhang et al., 2026a;b), or by employing a coding agent to rewrite the harness around a model (Lee et al., 2026; Lin et al., 2026); Gao et al. (2025a) provide a survey. Embodied systems retain experience as executable skills (Wang et al., 2024), validated program repairs (Lu et al., 2026), controller prompts and tools (Elmaaroufi et al., 2026) or per-task programs that call frozen policies (Wang et al., 2026c), or construct reset and verification interfaces with human feedback before improving policies through physical experiments (Xiao et al., 2026). Concurrent work evolves the guidance of an agent together with parts of its interface (Wang et al., 2026b; Ding et al., 2026; Li et al., 2026b; Tang et al., 2026) or specifies protocols for doing so (Wang et al., 2026a), and weight-level methods incorporate experience into the policy parameters (Bousmalis et al., 2023; Physical Intelligence et al., 2025a; Yu, 2026). The validity of reported self-improvement has, however, been questioned: at equal budget, harness evolution does not consistently outperform test-time scaling (Wang et al., 2026e), and frequent updates may fail to improve held-out performance (Wang et al., 2026e; Zheng et al., 2026); selection on small, noisy evaluations admits spurious improvements (Shawn, 2026; Gideoni et al., 2026; Zhou et al., 2026); self-modifying systems have been observed to tamper with their evaluators (Wang et al., 2026d); and loops with access to episode logs detect planted silent defects late or not at all and seldom produce tools (Zhou et al., 2026; Yang et al., 2026). The loop studied here adopts an RPent-style experience store and ENPIRE-style tool construction. Its contribution lies in a paired evaluation on the training tasks and on unseen task categories, in a setting where faults such as unreachable poses and millimeter-scale misalignment produce no explicit error signal (Appendix G).

3 Experimental Setup

Benchmark. The main evaluation covers all 26 tasks of the EBench test split (Gao et al., 2026), which span nine scene categories across household, retail and industrial settings simulated with GenManip and Isaac Sim (Gao et al., 2025b). The tasks are annotated by mobility (19 mobile and 7 tabletop tasks), manipulation precision (14 low-, 8 medium- and 4 high-precision tasks) and horizon (11 short- and 15 long-horizon tasks), and each task is evaluated under controlled variations of

Table 1: Success rate on the 26 tasks of the EBench test set: ranking by overall SR and Score, and detailed success rate in different domains.

			Precision			Mobility		Horizon	
System	SR	Score	Low	Med.	High	Mobile	Tabletop	Short	Long
OpenWAM-α (Wang et al., 2026f)	**55.3**	**0.701**	**67.4**	45.6	**32.5**	**60.2**	42.1	65.9	**47.6**
GPT-6-Astra (OpenAI, 2026)	46.7	0.654	60.6	40.2	11.3	56.6	20.0	**73.2**	27.3
Qwen-RobotManip (Yuan et al., 2026a)	45.6	0.608	50.6	**50.2**	18.8	43.9	**50.0**	55.9	38.0
$\pi_{0.5}$ (Physical Intelligence et al., 2025b)	41.4	0.544	49.6	39.0	17.5	45.9	29.3	52.3	33.4
InternVLA-A1.5 (Ma et al., 2026)	34.2	0.468	44.6	26.0	13.8	42.3	12.1	46.8	24.9
π_0 (Black et al., 2025)	33.7	0.475	39.0	30.0	22.5	34.0	32.9	40.5	28.8
GigaBrain-0.7 (GigaBrain Team, 2026)	33.3	0.460	35.4	33.8	25.0	31.6	37.9	40.0	28.3
Fast-WAM (Yuan et al., 2026b)	25.6	0.371	35.7	19.0	3.8	34.0	2.9	46.4	10.4

objects, backgrounds, instructions and their combinations. Over the 510 evaluated episodes, performance is measured by the success rate (SR) and by the Score, which credits the stages completed by the end of an episode according to task-specific rules; both are averaged within each task and then equally across tasks.

Agent. GPT-6-Astra (OpenAI, 2026), at high reasoning effort, serves directly as the control agent. At each decision step it receives RGB images and the robot state: end-effector (EEF) poses, gripper state, base pose and the elapsed simulator steps. The first observation contains the overview and both wrist views, and each subsequent call returns the views that the model requests. Depth, object poses, calibration, segmentation, intermediate scores and the step budget are withheld. The agent commands one or more EEF waypoints for the arms and the base, with gripper targets and durations; the waypoints are interpolated into an action chunk that executes open loop before the next observation is returned, and no post-trained policy generates or refines the actions (Appendix A). Each task comes with one demonstration, for which a separate GPT-6-Astra instance selects and annotates 10–18 keyframes of one training video and pairs them with the recorded states and actions.

Policies and protocol. The agent is compared with seven post-trained vision–language–action (VLA) and world-action (WAM) policies from the EBench leaderboard (Table 1), all trained on the public EBench training set of 6,600 episodes over the 26 tasks. Physical execution is subject to the same task-specific physics-step limits as the policies, whereas the agent's tokens, tool calls and inference time are not capped. The agent may end an episode before the limit, in which case the robot is held until the evaluation server determines SR and Score.

4 Benchmark Results

Without task-specific training, GPT-6-Astra ranks second of the eight models. With one annotated demonstration per task and no parameter updates, GPT-6-Astra attains an SR of 46.7% and a Score of 0.654, 8.6 points and 0.047 behind OpenWAM-α and ahead of the other six policies (Table 1). On the public leaderboard it ranks third; the other entry above it, AMapbot, has no public report or repository and is excluded from the analysis. With 20 episodes per task, differences of a few points between adjacent ranks are not resolved.

The rank depends on the task group. The agent attains the highest SR of the eight models on the short-horizon tasks (73.2%, against 65.9% for OpenWAM-α) but ranks sixth on the long-horizon tasks (27.3% against 47.6%) and sixth on the tabletop tasks (20.0%, against 50.0% for Qwen-RobotManip); its rank falls from second on low-precision to seventh on high-precision tasks (Figure 1a). Relative to the best policy on each task, its SR is higher on five tasks, equal on four and lower on seventeen (Appendix B). Across object, background, instruction and mixed variations, its SR stays between 44.2% and 50.8%, the narrowest range of the eight models.

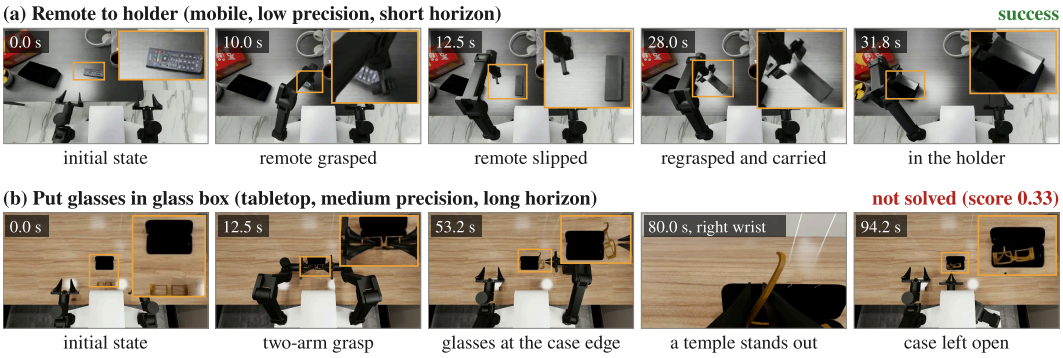


Figure 2: Two GPT-6-Astra evaluation episodes (overview camera unless marked; orange boxes are enlarged in the insets). (a) Remote to holder: the remote slips from the gripper during transport; the agent regrasps it and places it in the holder (success). (b) Glasses into case: after a two-arm transfer, a temple protrudes from the case and the lid stays open (Score 0.33).

5 Capability Analysis

5.1 Failures stop at a fixed stage rather than at the start

Because EBench credits intermediate stages, an unsuccessful episode can still receive partial credit. Of the 510 episodes, 237 succeed, and 188 of the 273 unsuccessful episodes receive partial credit; only 85 receive a Score of 0 (Figure 1b). Of the unsuccessful episodes, 229 run until the physics-step limit, and on the high-precision tasks they end at a common stage: all 18 unsuccessful episodes of nut tightening and all 16 of peg insertion receive a Score of 0.5 (Appendix Figure 8). The terminal Score cannot separate an imprecise final alignment from progress that stalls at that stage until the budget is exhausted. Tasks with similarly low success also fail differently: a Score of 0 ends 75% of the gear-installation and 65% of the utensil-placement episodes, but none of nut tightening, peg insertion or shop.

The 43 unsuccessful episodes that the agent ended itself separate the two failure modes more directly. In 22, its final message reports a drop that it could not recover, such as a jar or a bottle that fell out of reach. In the other 21, it reports a completed task that the scorer does not confirm: in all seven such dishwasher episodes, the agent states that both bowls are in the rack and ends the episode with the door open. The agent thus also misjudges completion, not only execution.

The interface compounds these failures: 84% of the agent's motion commands contain a single waypoint, it can correct only between action chunks while contact evolves within a chunk, and each retry consumes budget and may disturb a goal already achieved.

5.2 The deficit concentrates on tabletop dexterous-and-precise tasks

All tabletop tasks demand dexterous or precise manipulation, and we refer to them as dexterous-and-precise (D&P) tasks: frame placement, cup flipping and glasses packing through two-arm coordination, flipping and folding, and the others through tight contact. Even on the three D&P tasks labeled low or medium precision, GPT-6-Astra succeeds in 31.7% of episodes on average, compared with 91.7% for Qwen-RobotManip: 10% against 90% on cup flipping and 20% against 90% on glasses packing. Precision labels alone fail to account for this gap; the recordings reveal two further failure modes. In glasses packing, labeled medium precision, the agent completes the transfer into the case, but its corrective contacts pull a temple out and the regrasps exhaust the step limit before the lid is closed (Figure 2b; Appendix D), whereas OpenWAM-α folds the temples and closes the case. The second is bimanual coordination. Without a demonstration, the agent does not compose a two-arm plan for frame placement and attempts single-arm variants that leave the frame unsupported; with one, it divides the work between the arms and reaches 65%, above the 50% of OpenWAM-α (Section 5.5). Cup flipping, a purely bimanual problem, is not resolved by the demonstration.

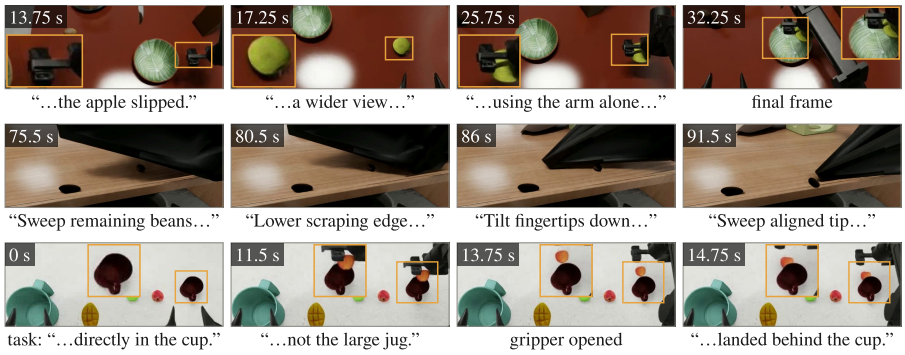


Figure 3: Adaptation in three GPT-6-Astra evaluation episodes, with the agent's stated reasons quoted below the frames (orange boxes are enlarged in the insets). (a) Apple to fruit bowl: after a slip, the agent takes a wider view and carries the apple with the arm alone (success). (b) Coffee beans, left wrist camera: after a sweep leaves the beans in place, the agent lowers the edge and tilts the fingertips (Score 0.5). (c) Fruit: the agent follows the cup named in the instruction rather than the jug of the demonstration, but the fruit lands behind the cup (Score 0.6).

5.3 Adaptation within an episode: exploration, recovery and self-correction

The recordings also show the agent revising its behavior within an episode. It regrasps a remote that has slipped from its gripper (Figure 2a). In apple transport (Figure 3a), it infers a slip from the closed finger gap, since the harness returns no failure signal, withdraws to a wider view, and re-plans the transport with the arm alone, avoiding the base motion that preceded the first slip; the episode succeeds. In coffee-bean collection (Figure 3b), after a sweep at 0.150 m leaves the beans in place, the agent repeats it 25 mm lower and then tilts the fingertips. In the fruit task (Figure 3c), it follows the cup named in the instruction rather than the jug of the demonstration, but its release misses the cup (Score 0.6). Such responses are common: 84% of the episodes contain at least one failed grasp, and in 294 of the 317 episodes whose stated reasons report a disruption, the agent follows with a corrective grasp. Experience is thus used within an episode but not retained across episodes; Section 6.2 tests whether it can be.

5.4 Safety in execution

Exploration and recovery also expose safety weaknesses. Objects drop from the gripper 270 times in 189 of the 510 episodes, and the agent's next three stated reasons mention only 51% of these drops. A bowl slips from one gripper while the other hand opens the dishwasher door, a dropped apple leaves the workspace while the agent keeps requesting targets for it, and a held jar strikes the table edge in an awkward pose (Appendix Figure 9). Grasp stability, workspace limits and scene contact must therefore be enforced during execution rather than restored afterward.

5.5 In-context learning helps where it supplies structure, not precision

In a paired comparison on all 20 test episodes of two tasks, in which only the demonstration is removed, the demonstration raises frame placement from 1 to 13 successes (exact McNemar test, $p < 0.001$; Appendix Figure 11). With it, the agent hands the frame from one arm to the other in 17 of 20 episodes, and every success includes such a handover, against 1 of 20 episodes without it. Gear installation remains at 3 of 20 successes in both conditions: the demonstration conveys the division of labor between the arms but not the millimeter alignment that seating a gear requires. The conditions were run on different dates, and four test scenes are the demonstration's own; without them, frame placement still rises from 1 to 10 of 16.

A composed instruction. In a separate pilot with an instruction outside EBench, placing a bookmark and then a pen on the same book, the agent placed the bookmark and then brought the pen onto the book in all 10 rollouts, with the scene's object names in the prompt; no policy was evaluated on it.

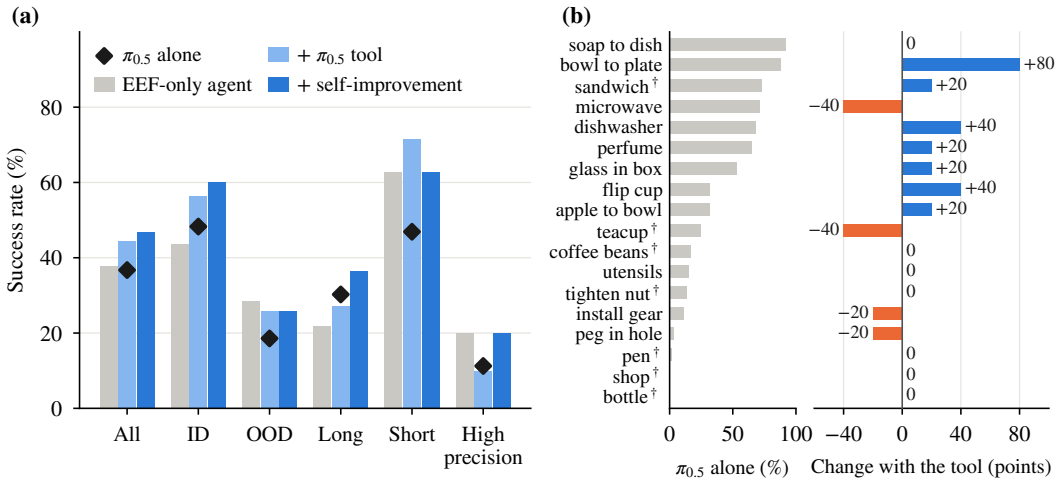


Figure 4: Success rate of GPT-6-Astra with and without $\pi_{0.5}$ as a tool on 18 test tasks (90 paired episodes per configuration). (a) By task group; diamonds: $\pi_{0.5}$ alone. (b) Per-task change from adding the tool (right), with tasks ordered by the success rate of $\pi_{0.5}$ alone (left); †: OOD task.

6 Coupling with a Learned Policy and Self-Improvement

Experimental design. Two means of reducing the gap identified in Section 5 are examined: coupling the agent with a post-trained policy and improvement from its own execution experience. Both experiments use 18 of the 26 test tasks with five paired episodes each (90 per configuration) and compare the EEF-only agent of the main evaluation, the agent with $\pi_{0.5}$ available as a tool and an empty playbook (the *starting point*), the version obtained after five rounds of self-improvement (the *final version*) and the final version without the demonstration. $\pi_{0.5}$ *alone* denotes the unpaired mean success rate of three public EBench fine-tuning runs of $\pi_{0.5}$ on the same scenes (details in Appendix E.1).

Training and test tasks. The loop is trained on ten EBench tasks in their training configurations: four pick-and-place, two long-horizon mobile-manipulation and four tabletop tasks. Each configuration provides five pre-generated scenes; in round $r \in \{0, \ldots, 4\}$, every training task is executed on its r-th scene once with and once without the demonstration, for a total of 100 training episodes. Evaluation uses the first five scenes of the test configurations of 18 EBench tasks, of which 7 are short-horizon, 11 long-horizon and 4 high-precision. Following the task grouping of the benchmark, 11 of these tasks are in-distribution (ID) for the loop, namely the ten training tasks and one untrained pick-and-place task, whereas the remaining seven are out-of-distribution (OOD) and never enter the loop. The test scenes of the training tasks differ from the corresponding training scenes only in the instruction wording (two long-horizon tasks), only in the textures (four tabletop tasks) or in the object instances (four pick-and-place tasks); for six tabletop task–scene pairs, the demonstration is a recording of the test scene itself (Appendix E.1).

6.1 Coupling with a post-trained policy

Following RPent (RLinf, 2026), a frozen $\pi_{0.5}$ checkpoint fine-tuned on the EBench training subsets is made available to the agent as a tool that executes up to 300 physics steps per call, and at each call the agent selects either this tool or direct EEF control.

Gains are concentrated on tasks that the policy solves on its own. Access to the tool changes the overall success rate from 37.8% to 44.4% (+6.7 points), a difference that is not significant. Success increases on the ID tasks, on which $\pi_{0.5}$ alone succeeds in about half of the episodes, but not on the OOD and high-precision tasks (Figure 4a), and the per-task change is weakly correlated with the success rate of $\pi_{0.5}$ alone ($r = 0.44$, $p \approx 0.07$; Figure 4b); shop and bottle, which $\pi_{0.5}$

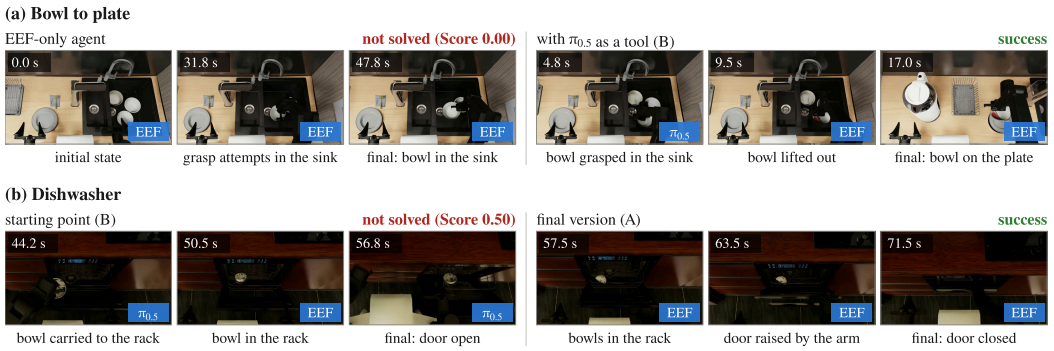


Figure 5: Keyframes from two test episodes, with the video time and the executor of each frame. (a) Bowl to plate: EEF only and with $\pi_{0.5}$ as a tool. (b) Dishwasher: starting point and final self-improved version, on the same scene.

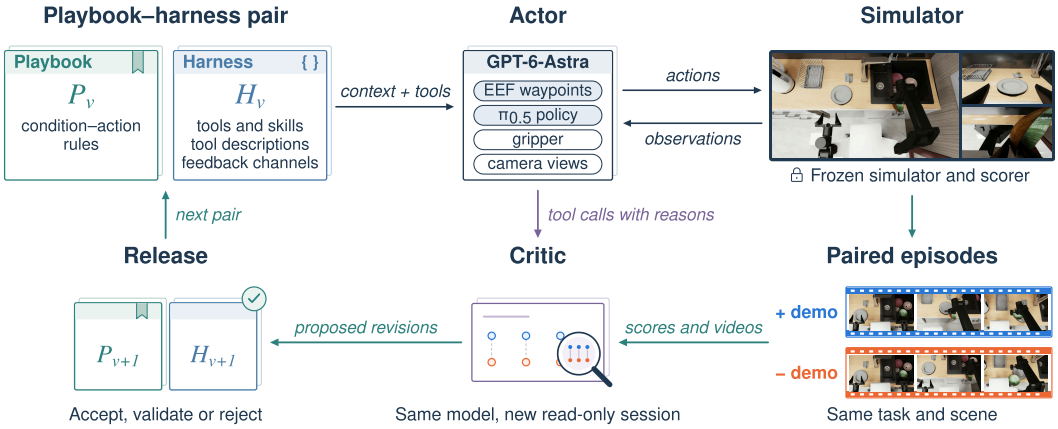


Figure 6: The self-improvement loop for GPT-6-Astra (five rounds, ten training tasks). The actor operates with the playbook P_v and the harness H_v through end-effector (EEF) waypoints or the frozen $\pi_{0.5}$ policy; each task is executed from the same scene with (+ demo) and without (− demo) the demonstration; the critic, a separate read-only instance of the same model, reviews the tool calls, Scores and videos, and the revisions it accepts constitute P_{v+1} and H_{v+1}. Frames: bowl-to-plate task in round 1, in which the run with the demonstration succeeds and the run without it fails.

alone never solves, remain unsolved. Coupling thus inherits the competence profile of the policy rather than extending it.

Coupling also reduces success on some tasks. Success decreases from 80% to 40% on microwave operation and from 100% to 60% on the teacup task. Where coupling succeeds, the executors are complementary: in Figure 5a, $\pi_{0.5}$ grasps the bowl in the sink and the agent places it on the plate under direct EEF control, whereas the EEF-only agent does not remove the bowl from the sink.

Delegation does not correspond to the competence of the policy. On the test tasks, the proportion of physics steps delegated to $\pi_{0.5}$ is nearly uncorrelated with the success rate of $\pi_{0.5}$ alone, both at the starting point ($r = 0.06$) and after self-improvement ($r = 0.25$), whereas on the training tasks, on whose configurations $\pi_{0.5}$ was fine-tuned, the two quantities are correlated ($r = 0.73$). Claude Opus 5.5, whose success rate at the same starting point is statistically indistinguishable (40.0% versus 44.4%), delegates more where $\pi_{0.5}$ alone is competent ($r = 0.59$).

6.2 SELF-IMPROVEMENT FROM EXECUTION FEEDBACK

The loop. The second approach combines an RPent-style experience store with ENPIRE-style tool construction (RLinf, 2026; Zhang et al., 2026e; Xiao et al., 2026) (Figure 6). The loop runs for five rounds on the ten training tasks (100 episodes); an automatic check rejects task-specific revisions, and neither model receives kinematic diagnostics or the millimeter tolerances of the scorers. The instructions to the critic were amended in task-independent ways and, in the final round, included one question from the authors about two repeatedly failing training tasks (Appendix E). The Opus version was evaluated after an additional, author-initiated critic pass on executor use.

Neither model improves on the test tasks. The final GPT-6-Astra version succeeds in 46.7% of the paired episodes, compared with 44.4% for its starting point, with unchanged OOD success (26%) despite the overlap of the ID tasks with the training scenes; the final Opus version attains 31.1%, compared with 40.0%. Neither difference is significant. Training Scores increase between the first and the last round (Appendix Figure 13), but each round uses different scenes, and the paired comparison on the test tasks does not establish a corresponding improvement.

The loop changes the strategy of the agent but not its precision. With the empty playbook, runs with and without the demonstration delegate 40% and 82% of physics steps to $\pi_{0.5}$, respectively; with every learned playbook, both delegate 60–76%. The final version ends fewer test episodes before they are scored (3 of 90 against 10), and after a rule on completing enclosures its stated reasons refer to closing the dishwasher in all five test episodes, against one at the starting point, while its success on this task rises from two to four (Figure 5b; Appendix F.6). The harness remains nearly unchanged: 20 of the 22 accepted revisions are playbook rules, no accepted rule of either model specifies a numerical tolerance, and gear installation is never completed in the last three rounds.

The critic diagnoses single episodes but not the distribution of failures. All six unsuccessful training episodes whose review names a motion error contain a deviation above 5 cm, but the critic names a motion error in only 6 of the 12 episodes with such a deviation, and, although 16 of its 47 reviews attribute the failure to judgment, its aggregate classification assigns all eight failure modes to execution (Table 16). The consolidated Opus guidance is inconsistent about when to take control from $\pi_{0.5}$, and in 31 of 90 test episodes its first call to $\pi_{0.5}$ has an invalid tool name (Figure 14).

7 CONCLUSIONS AND FUTURE DIRECTIONS

GPT-6-Astra interprets manipulation tasks flexibly but does not execute them reliably. With one annotated demonstration per task, it ranks second of eight models on EBench, first on short procedures but sixth on long-horizon and tabletop tasks, and it explores and recovers within an episode; precise alignment and contact remain the bottleneck. Neither a post-trained policy called as a tool nor self-improvement from execution feedback significantly narrows the gap (Section 6).

These observations suggest four directions: *combining agents with post-trained policies* through handoff protocols in which a local controller monitors slip and resistance; *teaching smaller policies to plan, explore and recover* from rollouts that pair failure states with revised strategies; *grounding experience in reusable tools* with diagnostics that images and outcomes do not provide, such as reachability and contact reports; and *treating safety as an active constraint*, with force, velocity and workspace limits in the action pipeline.

Limitations. The policies' results come from the leaderboard, all experiments use one simulator and one robot, and the recordings were selected to illustrate each behavior. The experiments of Section 6 use five episodes per task and one run of the loop per model, several training tasks are tested on nearly identical scenes, the protocol was amended during training, including one question from the authors about two failing training tasks (Appendix E), and the critic shares the actor's model (Tang et al., 2026).

AI USE STATEMENT

LLM agents are the object of study in this work: GPT-6-Astra and Claude Opus 5.5 act as robot policy, actor and critic. We used generative AI tools (LLM coding assistants) to help build the experiment infrastructure and the analysis scripts, and to draft and edit the manuscript, including its text, tables and figures. Every reported number was computed by scripts from logged data and checked against the experiment records, and every reference was checked against arXiv or publisher metadata. The authors reviewed all AI-assisted work and take responsibility for the final content of this work, including text, claims and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

All experiments are in simulation, with no human subjects. The simulator's end-effector controller occasionally leaves the end effector far from its commanded pose without reporting an error (3.1% of commanded waypoints by more than 5 cm; Appendix F); on hardware such behavior would be unsafe, and deploying a comparable system would require runtime guardrails below the agent (force, velocity and workspace limits) and human oversight.

REPRODUCIBILITY STATEMENT

Section 3 describes the systems, task splits, comparisons and statistics; Section 6.2 and Appendix E describe the loop, the critic protocol and the budgets. Every table and figure is computed by scripts from the logged episodes and critic outputs. We will release the episode logs, the critic inputs and outputs, every playbook–harness version, the actor and critic prompts and the analysis code.

REFERENCES

Anthropic. System card: Claude Opus 5.5, September 2026. URL `https://www-cdn.anthropic.com/fc1b44717c85dc068bc6ba5024219938094694bd/Claude%20Opus%205.5%20System%20Card.pdf`.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.010.

Konstantinos Bousmalis, Giulia Vezzani, Dushyant Rao, Coline Devin, Alex X. Lee, Maria Bauza, Todor Davchev, Yuxiang Zhou, Agrim Gupta, Akhil Raju, Antoine Laurens, Claudio Fantacci, Valentin Dalibard, Martina Zambelli, Murilo Martins, Rugile Pevceviciute, Michiel Blokzijl, Misha Denil, Nathan Batchelor, Thomas Lampe, Emilio Parisotto, Konrad Żołna, Scott Reed, Sergio Gómez Colmenarejo, Jon Scholz, Abbas Abdolmaleki, Oliver Groth, Jean-Baptiste Regli, Oleg Sushkov, Tom Rothörl, José Enrique Chen, Yusuf Aytar, Dave Barker, Joy Ortiz, Martin Riedmiller, Jost Tobias Springenberg, Raia Hadsell, Francesco Nori, and Nicolas Heess. RoboCat: A self-improving generalist agent for robotic manipulation. *Transactions on Machine Learning Research*, 2023. arXiv:2306.11706.

Jisong Cai, Yao Mu, Ganlin Yang, Zhe Cao, Zhangzheng Tu, Xing Gao, Kailin Li, Xinyu Zhan, Lixin Yang, Yangkun Zhu, Haoxiang Ma, Ming Zhou, Qiaojun Yu, Yufei Xue, Liqun He, Yifei Yao, Yifan Zhu, Long Ling, Bingqi Jiang, Haoyu Guo, Xueyue Zhu, Bowen Zhou, Bin Zhao, Tianfan Xue, Chunhua Shen, and Weinan Zhang. InternW0: A foundational physical world model for efficient real-world interactions. *arXiv preprint arXiv:2609.27656*, 2026.

Yanzhe Chen, Zechen Bai, Zhijun Cao, Wenzheng Zeng, Kevin Qinghong Lin, Yiqi Lin, Guoqiang Liang, Kevin Yuchen Ma, Qiming Huang, and Mike Zheng Shou. Show-Harness: Just a VLM agent can play robots. *arXiv preprint arXiv:2609.10522*, 2026.

Xin Ding, Liang Mi, Mingzhe Huang, Zixuan Wang, Chao Zhang, Zixu Hao, Fu Chen, Xiangyu Li, Yikai Zheng, Yaoyu Guo, Weijun Wang, Kun Li, Hao Wu, Yunxin Liu, and Ting Cao. Zetta ζ:

An efficient closed-loop embodied harness for self-evolving physical intelligence. *arXiv preprint arXiv:2608.16590*, 2026.

Karim Elmaaroufi, Justin Svegliato, Sarunas Kalade, Graham Schelle, Sanjit A. Seshia, and Matei Zaharia. RHO: Your coding agent is secretly a roboticist. *arXiv preprint arXiv:2606.16458*, 2026.

Roya Firoozi, Johnathan Tucker, Stephen Tian, Anirudha Majumdar, Jiankai Sun, Weiyu Liu, Yuke Zhu, Shuran Song, Ashish Kapoor, Karol Hausman, Brian Ichter, Danny Driess, Jiajun Wu, Cewu Lu, and Mac Schwager. Foundation models in robotics: Applications, challenges, and the future. *arXiv preprint arXiv:2312.07843*, 2023.

Huan-ang Gao, Jiayi Geng, Wenyue Hua, Mengkang Hu, Xinzhe Juan, Hongzhang Liu, Shilong Liu, Jiahao Qiu, Xuan Qi, Yiran Wu, Hongru Wang, Han Xiao, Yuhang Zhou, Shaokun Zhang, Jiayi Zhang, Jinyu Xiang, Yixiong Fang, Qiwen Zhao, Dongrui Liu, Qihan Ren, Cheng Qian, Zhenhailong Wang, Minda Hu, Huazheng Wang, Qingyun Wu, Heng Ji, and Mengdi Wang. A survey of self-evolving agents: What, when, how, and where to evolve on the path to artificial super intelligence. *arXiv preprint arXiv:2507.21046*, 2025a.

Ning Gao, Yilun Chen, Shuai Yang, Xinyi Chen, Yang Tian, Hao Li, Haifeng Huang, Hanqing Wang, Tai Wang, and Jiangmiao Pang. GenManip: LLM-driven simulation for generalizable instruction-following manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12187–12198, 2025b. doi: 10.1109/CVPR52734.2025.01138.

Ning Gao, Jinliang Zheng, Xing Gao, Haoxiang Ma, Hanqing Wang, Yukai Wang, Jiantong Chen, Zanxin Chen, Shujie Zhang, Mingda Jia, Xuekun Jiang, Zihou Zhu, Xinyu Li, Shuai Wang, Hao Li, Wenzhe Cai, Yuqiang Yang, Xudong Xu, Zhaoyang Lyu, Yao Mu, Tai Wang, Jiangmiao Pang, Jia Zeng, Weinan Zhang, and Chunhua Shen. EBench: Elemental diagnosis of generalist mobile manipulation policies. *arXiv preprint arXiv:2606.18239*, 2026.

Yonatan Gideoni, Sebastian Risi, and Yarin Gal. Simple baselines are competitive with code evolution. *arXiv preprint arXiv:2602.16805*, 2026.

GigaBrain Team. GigaBrain-0.7: Scaling embodied foundation models to emergent capabilities with a three-system architecture. *arXiv preprint arXiv:2608.15875*, 2026. Angen Ye and Axiang Sun and Can Jin and Chenxi Cheng and Chong Shi and Dengke Shang and Dingqian Zhang and Guan Huang and Guangqiang Wang and Guangqing Ding and Guo Li and Hangcong Li and Hengyu Zhong and Hongtao Lu and Jianbo Qin and Jiming Mao and Jing Zhu and Jindi Lv and Jingzhi Cui and Junjie Xie and Junyi Bao and Kai Liu and Lei Yuan and Limin Long and Lv Feng and Mingming Yu and Peng Li and Pengfei Yi and Qi Li and Qianli Zhang and Qingfang Li and Qitang Hu and Rui Zhang and Shaoyan Sun and Shibo Sun and Shiying Duan and Tenghui Chen and Tianze Liu and Weijie Ke and Wenyao Xue and Xiaofeng Wang and Xiaoyu Tian and Xinyu Liu and Xinze Chen and Yang Wang and Yankai Wang and Yejun Zeng and Yifan Li and Yifei Nie and Yilong Li and Yilong Liu and Yongchao Feng and Yumeng Wang and Yun Ye and Zhichao Liu and Ziheng He and Zonghai Yang and Zheng Zhu (full author list).

Mengzhao Jia, Yang Lin, Xixin Zhang, Zhihan Zhang, Xiaobai Liu, and Meng Jiang. Agent as policy for robotic manipulation. *arXiv preprint arXiv:2609.12541*, 2026.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2024.

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

Yoonho Lee, Roshen Nair, Qizheng Zhang, Kangwook Lee, Omar Khattab, and Chelsea Finn. Meta-Harness: End-to-end optimization of model harnesses. *arXiv preprint arXiv:2603.28052*, 2026.

Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Fei Han, Mingrui Yu, Zelin Gao, Nan Xue, Xing Zhu, Yujun Shen, and Yinghao Xu. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026a. LingBot-VA.

Zexi Li, Yehang Zhang, Haojian Huang, Bohan Zhou, Wenqian Li, Chenxu Wang, Yifan Chang, Yangkai Wei, Tianyi Zhang, Ying-Cong Chen, Kaiwen Zhou, Yinchuan Li, and James Cheng. RACaP: Agentic reasoning, acting, and coding as policies for evolvable robot learning. *arXiv preprint arXiv:2609.29394*, 2026b.

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. *arXiv preprint arXiv:2209.07753*, 2022.

Jiahang Lin, Shichun Liu, Chengjun Pan, Lizhi Lin, Shihan Dou, Zhiheng Xi, Xuanjing Huang, Hang Yan, Zhenhua Han, Tao Gui, and Yu-Gang Jiang. Agentic harness engineering: Observability-driven automatic evolution of coding-agent harnesses. *arXiv preprint arXiv:2604.25850*, 2026.

Zeyu Lou, Yanhong Zeng, Yong Wang, and Chenyang Si. Know your body: A harness for direct and self-improving robot control with VLMs. *arXiv preprint arXiv:2609.28530*, 2026.

Runyu Lu, Yubo Wu, Ethan Kou, Letian Fu, Wenli Xiao, Ajay Mandlekar, Yinzhen Xu, Guanya Shi, Ken Goldberg, Ang Chen, Mosharaf Chowdhury, Yuke Zhu, Linxi "Jim" Fan, and Guanzhi Wang. ASPIRE: Agentic /skills discovery for robotics. *arXiv preprint arXiv:2607.00272*, 2026.

Haoxiang Ma, Junhao Cai, Xiaoxu Xu, Hao Li, Yuyin Yang, Yang Tian, Jiafei Cao, Hongrui Zhu, Zherui Qiu, Zhaxizhuoma, Yuqiang Yang, Jiaqi Peng, Xueyuan Wei, Yangkun Zhu, Jiahao Jiang, Xing Gao, Hanqing Wang, Feng Yuan, Kailin Li, Xueyue Zhu, Tai Wang, Yan Ding, Jiangmiao Pang, Jia Zeng, Jingjing Zhang, Bowen Zhou, Yao Mu, Chunhua Shen, and Weinan Zhang. InternVLA-A1.5: Unifying understanding, latent foresight, and action for compositional generalization. *arXiv preprint arXiv:2607.04988*, 2026.

Yueen Ma, Zixing Song, Yuzheng Zhuang, Jianye Hao, and Irwin King. A survey on vision-language-action models for embodied AI. *arXiv preprint arXiv:2405.14093*, 2024.

OpenAI. GPT-6 Astra system card. OpenAI Deployment Safety Hub, September 2026. URL `https://deploymentsafety.openai.com/gpt-6-astra`.

Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, Danny Driess, Michael Equi, Adnan Esmail, Yunhao Fang, Chelsea Finn, Catherine Glossop, Thomas Godden, Ivan Goryachev, Lachy Groom, Hunter Hancock, Karol Hausman, Gashon Hussein, Brian Ichter, Szymon Jakubczak, Rowan Jen, Tim Jones, Ben Katz, Liyiming Ke, Chandra Kuchi, Marinda Lamb, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Yao Lu, Vishnu Mano, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Charvi Sharma, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, Will Stoeckle, Alex Swerdlow, James Tanner, Marcel Torne, Quan Vuong, Anna Walling, Haohuan Wang, Blake Williams, Sukwon Yoo, Lili Yu, Ury Zhilinsky, and Zhiyuan Zhou. $\pi^*_{0.6}$: a VLA that learns from experience. *arXiv preprint arXiv:2511.14759*, 2025a.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025b.

RLinf. RPent documentation: System description. Online documentation, 2026. URL `https://rpent.readthedocs.io/en/latest/rst_source/development/architecture.html`.

Zayx Shawn. PACE: Anytime-valid acceptance tests for self-evolving agents. *arXiv preprint arXiv:2606.08106*, 2026.

Qiuhong Shen, Shihua Zhang, Yue Liao, Qi Li, Zhenxiong Tan, Shizun Wang, Shuicheng Yan, and Xinchao Wang. World action models: A survey. *arXiv preprint arXiv:2606.20781*, 2026.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 8634–8652, 2023. doi: 10.52202/075280-0377.

Junyi Tang, Jie Peng, Zezhen Ding, Yuan Shen, and Tianlong Chen. AdaHVLA: Adaptive harnesses for long-horizon vision-language-action execution. *arXiv preprint arXiv:2609.29204*, 2026.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=ehfRiF0R3a.

Kailin Wang, Haoxiang Jie, Yaoyuan Yan, Zhiyou Heng, and Zhaosong Li. RegenHarness: A robot agent harness with evidence-gated recursive self-improvement. *arXiv preprint arXiv:2609.27612*, 2026a.

Peidong Wang, Zhiming Ma, Ying Chang, Xufang Luo, Yiqun Zhang, Zihan Wang, Xiaocui Yang, Shi Feng, Yuqing Yang, and Dongsheng Li. Self-evolving embodied agents via skill-harness evolution. *arXiv preprint arXiv:2608.11350*, 2026b.

Xin Wang, Wenhao Wu, Menghao Zhang, Zhi Wang, Kun Shao, Jian Luan, Yang Li, Qing Li, Shangding Gu, Huichi Zhou, Shuqing Shi, Fei Ni, Shuo Lu, Weicheng Meng, Kang Li, Jin Wu, Kang Zhao, Shangmin Guo, Gen Li, Yongqiang Tang, Zhizhong Zhang, Yuan Xie, and Heng Qu. HarnessPAI: An evolving harness for physical AI. *arXiv preprint arXiv:2609.29166*, 2026c.

Xing Wang, Xiaoyi Zhang, and Jie Shao. Auditing harness tampering in self-improving agents. *arXiv preprint arXiv:2609.00069*, 2026d.

Yike Wang, Huaisheng Zhu, Zhengyu Hu, Yige Yuan, Zhengyu Chen, Shakti Senthil, Hannaneh Hajishirzi, Yulia Tsvetkov, Pradeep Dasigi, and Teng Xiao. Rethinking the evaluation of harness evolution for agents. *arXiv preprint arXiv:2607.12227*, 2026e.

Yuran Wang, Siqiao Huang, Mingleyang Li, Chenhao Zhang, Jiaqi Liang, Weiyang Jin, Yue Chen, Xuemin Chi, Donghao Zhou, Qize Yu, Yu-Kai Wang, Yuhan Rui, Shenzhe Yao, Zhen Yuan, Zhenhao Shen, Kefei Zhu, Zijie Zhu, Ning Gao, Xiaowei Chi, Guanqi He, Shanghang Zhang, Hao Dong, Lin Shao, and Hang Zhao. OpenWAM: An open, modular exploration towards systematic world-action model pretraining. *arXiv preprint arXiv:2609.07398*, 2026f.

Wenli Xiao, Jia Xie, Tonghe Zhang, Haotian Lin, Letian "Max" Fu, Haoru Xue, Jalen Lu, Yi Yang, Cunxi Dai, Zi Wang, Jimmy Wu, Guanzhi Wang, S. Shankar Sastry, Ken Goldberg, Linxi "Jim" Fan, Yuke Zhu, and Guanya Shi. ENPIRE: Agentic robot policy self-improvement in the real world. *arXiv preprint arXiv:2606.19980*, 2026. arXiv comment: 2026 Conference on Robot Learning.

Bingxin Xu, Yuzhang Shang, and Emilio Ferrara. Don't drop the BATON: Long-horizon robot manipulation via agentic subtask exploration and transition-aware memory. *arXiv preprint arXiv:2608.16889*, 2026.

Siqi Yang, Qianlan Yang, Yu-Xiong Wang, Saurabh Pujar, and Martin Hirzel. One Recipe, Many Harnesses: What Self-Evolution Encodes Across Languages and Models. *arXiv preprint arXiv:2608.10178*, 2026.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, Ayaan Malik, Kyungmin Lee, William Liang, Nadun Ranawaka, Jiasheng Gu, Yinzhen Xu, Guanzhi Wang, Fengyuan Hu, Avnish Narayan, Johan Bjorck, Jing Wang, Gwanghyun Kim, Dantong Niu, Ruijie Zheng, Yuqi Xie, Jimmy Wu, Qi Wang, Ryan Julian, Danfei Xu, Yilun Du, Yevgen Chebotar, Scott Reed, Jan Kautz, Yuke Zhu, Linxi "Jim" Fan, and Joel Jang. World Action Models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026.

Hang Yu. You don't need to stay in the loop: An agentic robotics loop for robot-policy improvement. *arXiv preprint arXiv:2608.07555*, 2026.

Haoqi Yuan, Zhixuan Liang, Anzhe Chen, Ye Wang, Haoyang Li, Pei Lin, Yiyang Huang, Zixing Lei, Tong Zhang, Jiazhao Zhang, Jie Zhang, Jingyang Fan, Gengze Zhou, Qihang Peng, Chenxu Lv, Xiaoyue Chen, An Yang, Fei Huang, Junyang Lin, Dayiheng Liu, Jingren Zhou, Chenfei Wu, and Xiong-Hui Chen. Qwen-RobotManip technical report: Alignment unlocks scale for robotic manipulation foundation models. *arXiv preprint arXiv:2606.17846*, 2026a.

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*, 2026b.

Jenny Zhang, Shengran Hu, Cong Lu, Robert Lange, and Jeff Clune. Darwin Gödel Machine: Open-ended evolution of self-improving agents. In *International Conference on Learning Representations (ICLR)*, 2026a.

Jenny Zhang, Bingchen Zhao, Wannan Yang, Jakob Foerster, Jeff Clune, Minqi Jiang, Sam Devlin, and Tatiana Shavrina. Hyperagents. *arXiv preprint arXiv:2603.19461*, 2026b.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, Urmish Thakker, James Zou, and Kunle Olukotun. Agentic context engineering: Evolving contexts for self-improving language models. In *International Conference on Learning Representations (ICLR)*, 2026c.

Wenbo Zhang, Kaixuan Wang, Yutao Ouyang, Xiaoyu Huang, Liyang Li, Kailun Su, Weiyang Jin, Wenhao Chai, Haotian Liang, Zhiyang Dou, Yue Chen, and Tianxing Chen. An unexpected robot policy: Early evaluations of GPT-6 astra on RoboDojo and beyond. *arXiv preprint arXiv:2609.24170*, 2026d.

Yixian Zhang, Huanming Zhang, Feng Gao, Xiao Li, Zhihao Liu, Yi Nie, Chunyang Zhu, Jiaxing Qiu, Yuchen Yan, Jiyuan Liu, Wenhao Tang, Jiaji Rao, Zhengru Fang, Changxu Wei, Yu Wang, Wenbo Ding, and Chao Yu. Harness VLA: Steering frozen VLAs into reliable manipulation primitives via memory-guided agents. *arXiv preprint arXiv:2607.08448*, 2026e. Code: RLinf/RPent.

Congjie Zheng, Chuanyi Xue, Bin Liang, Jun Yang, and Changshui Zhang. SEAGym: An evaluation environment for self-evolving LLM agents. *arXiv preprint arXiv:2606.17546*, 2026.

Jian Zhou, Sihao Lin, Jin Li, Shuai Fu, Gengze Zhou, and Qi Wu. Automating the design of embodied agent architectures. *arXiv preprint arXiv:2606.30111*, 2026.

Appendix

The appendix follows the order of the main text. Appendices A–D support the capability analysis of Sections 3–5, and Appendices E–G support the experiments of Section 6.

- Appendix A: the agent's instructions, tools, demonstration format and tool responses.
- Appendix B: per-task results and the capability profile.
- Appendix C: keyframes of evaluation episodes (failure modes, safety, in-context learning, other policies).
- Appendix D: call-by-call analyses of three evaluation episodes.
- Appendix E: design, implementation, critic instructions and learned playbook of the self-improvement experiments.
- Appendix F: additional results, recorded episodes and case analyses for Section 6.
- Appendix G: comparison with prior self-improvement and harness systems.

A Agent interface and prompts

This appendix reproduces the inputs and outputs of the agent from the stored records; the critic's instructions are given in Appendix E.3. Omissions are marked [...]; other bracketed text replaces images, identifiers and file paths or labels an excerpt; non-ASCII characters are transliterated; and line breaks within a paragraph result from typesetting.

A.1 Actor instructions and task prompt

System instructions. Every actor session of the main evaluation begins with the following system instructions, followed by instructions of the model host on returning images from tool calls. In the coupling and self-improvement runs, the list of prohibited actions ends with "or delegate to other agents", and when the demonstration is supplied, the `query_icl` paragraph that closes the task prompt below is inserted between the two parts.

```
You control one simulated EBench robot episode. Use only the supplied
↪  RGB and measured robot state and the declared robot tools. Do not
↪  read external files, use shell, browse, or delegate. The server
↪  judges success independently. Stop when it reports a terminal
↪  result or a budget is exhausted. Repeated observe at the same
↪  timestep shows the same scene. A closed gripper or finger contact
↪  alone does not prove a secure grasp. Choose continuous poses,
↪  motion timing and observation cadence yourself.

[... instructions of the model host on returning images from tool calls
↪  ...]
```

Task prompt. The first user turn consists of the task prompt, the demonstration (Appendix A.3) and the current observation. The excerpt below is the first turn of the final version on the bowl-to-plate test task (seed 000). The EEF-only agent of the main evaluation received the same text without the playbook (the second line and the twelve entries that follow it), the paragraph on `execute_pi05` and the paragraph on `query_icl`; the starting point received it without the playbook.

```
Complete this single task yourself using the tools: Take out one of the
↪  three bowls and stack on the plate.
Validated experience memory from earlier TRAIN rounds (hypotheses that
↪  passed validation, not guarantees; verify against the current
↪  views):
[... the twelve playbook entries ...]
```

```
Robot: manip/lift2/R5a, two arms and a movable planar base. Left arm is
 ↪  left in the overlook view. EEF poses are local to each arm's
 ↪  planner base: +X forward, +Y robot-left, +Z up; quaternion order
 ↪  wxyz. [...] There is no supplied depth, object pose or camera
 ↪  calibration. Only RGB and measured robot EEF/gripper/base state are
 ↪  available. [...] Use execute_eef sequentially, not parallel calls.
 ↪  Use observations to judge motion and object manipulation. The
 ↪  server determines task success independently. Do not start another
 ↪  episode. Resource limits: unlimited tool calls, physics deadline
 ↪  enforced by the server; no hard token limit; [...] When finished,
 ↪  state the observed outcome concisely.
You may call execute_pi05 to delegate a manipulation segment to a
 ↪  learned policy. Choose when to delegate using the current scene. It
 ↪  replans every 30 steps and automatically returns control after the
 ↪  requested segment (at most 300 steps). The pi0.5 prompt is fixed
 ↪  for this task; always pass instruction=null and never invent a
 ↪  per-call prompt. Inspect its returned RGB/state and take over with
 ↪  EEF waypoints or delegate again. [...]
Manipulation guidance: after visually aligning, close at a stationary
 ↪  pose and allow settling. A nonzero finger gap on the table is
 ↪  contact evidence, not proof of a secure grasp. Test each grasp by
 ↪  lifting only 0.01 m first, using motion_steps=16 or more, then hold
 ↪  8..16 steps and request the active wrist plus overlook views. [...]
RGB inputs are downsampled: overlook 640x360, wrist 384x288; no crop or
 ↪  object annotation.
The saved ICL demonstration remains available through query_icl
 ↪  throughout this episode. [...] Ground actions in the current
 ↪  RGB/state.
[... the demonstration: annotation text and ten captioned keyframe
 ↪  images ...]
END OF HISTORICAL DEMONSTRATION. The following images and measured
 ↪  state are the CURRENT episode. [...]
Measured robot state (EEF poses in each arm's planner base):
 ↪  {"state.gripper": [0.044, 0.044, 0.044, 0.044], "state.base":
 ↪  [-0.0, 0.0, -0.0], "state.ee_pose": [[[0.1002, 0.0, 0.16349], [1.0,
 ↪  0.0, 2e-05, -0.0]], [[0.1002, 0.0, 0.16349], [1.0, -0.0, 2e-05,
 ↪  -0.0]]], "timestep": 0}
CURRENT RGB: overlook, 640x360
[image: overview camera]
[... the left and right wrist views in the same format ...]
```

A.2 Tool interface

Tools. Table 2 lists the actor's tools with their argument schemas and their descriptions as declared to the model. The EEF-only agent of the main evaluation has the first three tools. Relative to it, the coupled agent has `execute_pi05` and, when the demonstration is supplied, `query_icl`; its task prompt adds the playbook, which is empty at the starting point, and the two paragraphs shown above; and from training round 1 onward, including the evaluation of the final version, every response to an action tool carries the feedback fields of Appendix A.4 (the end-effector residual only for `execute_eef`), which the tool descriptions do not document. A `run_skill` tool is declared only once a skill has been published; since no skill and no tool description was ever published for GPT-6-Astra, the descriptions in Table 2 apply to every round and configuration.

A.3 Demonstration format

Annotated demonstration. The demonstration enters the first turn as one text block followed by alternating keyframe captions and images. The text block of the bowl-to-plate task consists of the annotation of the source episode, a fixed paragraph on the interpretation of the records and one numeric record per keyframe; each record holds the measured end-effector poses (`ee0` left, `ee1`

Table 2: Actor tools, with descriptions reproduced from the tool declarations (omissions marked [...]). All tools except `query_icl` also take `cameras` (one to three of `overlook`, `left_wrist`, `right_wrist`), and `execute_eef` and `execute_pi05` a free-text `reason`. Availability: *all*, every configuration; *coupled*, the starting point and the final version with and without the demonstration; *demo*, the coupled configurations with the demonstration.

Tool	Arguments	Available	Description as shown to the model
`execute_eef`	`waypoints`: 1–6 of {`left`, `right`: position, quaternion and two finger targets, or null; base pose or null; `steps` 1–64; optional `motion_steps`}	all	"Execute 1..6 native GenManip absolute EEF waypoints in order. Each arm target has xyz(m), unit quaternion(wxyz) in that arm's planner-base frame, and two finger joint targets(m): [0.044,0.044] open, [0,0] close. [...] The simulator performs IK; targets may remain unreachable, so inspect returned measured poses. No automatic grasp approval or task-stage rules are imposed. Returns final robot state and selected RGB cameras; sequences have no intermediate model observation. [...]"
`set_gripper`	`side`, `opening` in [0,1], `steps` 1–64	all	"Send a normalized gripper opening command: opening=0 fully closed, opening=1 fully open. [...] Use this for closing at rest; measured opening may remain nonzero due to contact. Returns selected RGB and measured robot state."
`observe`	—	all	"Return selected RGB views and measured robot state from the latest acknowledged simulator step. Does not advance physics. [...]"
`execute_pi05`	`steps` 1–300, `instruction` (string or null)	coupled	"Delegate a bounded manipulation segment to the trained EBench pi0.5 policy. It uses the CURRENT RGB images and robot state, predicts absolute joint/gripper targets, and replans every 30 steps. Choose this when learned dexterity may help with grasping, alignment, insertion or other fine manipulation. [...] After the requested steps, control automatically returns to you with fresh RGB and state; [...] pass instruction=null. [...]"
`query_icl`	`view` (text, frames or all), keyframe indices	demo	"The saved ICL demonstration remains available through query_icl throughout this episode. [...] Querying does not advance simulation. Ground actions in the current RGB/state."

right; position and wxyz quaternion), the summed finger gaps and the base triplet, together with the exported action target. The excerpt retains the record of one keyframe and the captions of two.

```
HISTORICAL DEMONSTRATION -- reference from a separate recorded episode,
↪  not the current observation or the current episode's action
↪  history.
Historical demonstration of bowl_to_plate, source episode 600.
↪  Reference instruction: Take out one of the three bowls and stack on
↪  the plate.
Arm roles: Right arm; base shifts from the sink toward the plate.
Observed sequence: Reach into the sink and grip the rim of the brown
↪  bowl. Lift it clear of the other bowls and sink rim, reposition the
↪  base toward the plate on the counter, place the bowl over the
↪  plate, lower it onto the plate, and open the fingers.
Observed outcome: The brown bowl is visibly supported on the white
↪  plate after the right fingers open; two patterned bowls remain in
↪  the sink.
Evidence limits: Final contact is partly occluded by the gripper. No
↪  official success label or force measurement is supplied.
This separate recorded episode is not the current observation or action
↪  history. [...] Transfer the sequence and relative geometry, ground
↪  commands in live RGB and robot state, and obey the live task
↪  instruction.
Source-aligned numeric keyframes:
[...]
```

```
{"frame":188,"source_timestamp_s":12.533333778381348,"measured":{"ee0_
↪  xyz_wxyz":[0.10019999742507935,-0.0,0.16348999738693237,1.0,0.0,1.
↪  9999999494757503e-05,-0.0],"ee1_xyz_wxyz":[0.3837299942970276,0.01
↪  508999988436699,0.045830000191926956,0.7011500000953674,9.99999974
↪  7378752e-05,0.7130100131034851,-9.999999747378752e-06],"finger_gap
↪  s_m_ee0_ee1":[0.08799000084400177,0.015189999714493752],"recorded_
↪  base_triplet":[0.0,0.15227000415325165,0.0]},"exported_action":{"e
↪  e0_xyz_wxyz":[0.10019999742507935,-0.0,0.16348999738693237,1.0,0.0
↪  ,9.999999747378752e-06,-0.0],"ee1_xyz_wxyz":[0.009809999726712704,
↪  0.04008999839425087,0.2037300020456314,0.7348499894142151,0.245710
↪  00039577484,-0.2593800127506256,0.5764899849891663],"finger_gaps_m
↪  _ee0_ee1":[0.08799000084400177,0.0],"recorded_base_triplet":[0.0,0
↪  .15227000415325165,0.0]}}
[...]
Historical demonstration: Historical episode 600; frame 188; 12.533s;
↪  overlook camera. Grasp and lift the brown bowl.
[image: keyframe 3]
[...]
Historical demonstration: Historical episode 600; frame 440; 29.333s;
↪  overlook camera. Open the right fingers around the supported bowl.
[image: keyframe 8]
[...]
```

A.4 Tool responses and feedback channels

Direct EEF control. Every tool response consists of a JSON receipt, a text line with the measured robot state and a caption and an image for each requested camera. The exchange below is call 11 of a training episode without the demonstration (dishwasher task, round 3), whose review is quoted in Appendix E.3. The commanded right-arm position lies 0.435 m from the position reached; the receipt reports this residual but no error, and the reason stated with the next call is appended.

```
[Request, call 11: execute_eef]
{"cameras": ["overlook", "right_wrist"], "reason": "Approach outer
↪  dishwasher door edge from front, clear of bowls in rack.",
↪  "waypoints": [{"base_xy_yaw_deg": null, "left": null, "right":
↪  {"xyz_m": [0.36, 0.18, -0.15], "quaternion_wxyz": [1, 0, 0, 0],
↪  "gripper_m": [0.044, 0.044]}, "motion_steps": 48, "steps": 56}]}
[Response]
{"executed_waypoints": [{"waypoint": 1, "executed_steps": 56,
↪  "measured_after": {"state.gripper": [0.044, 0.044, 0.044, 0.044],
↪  "state.base": [-0.47971, -0.12028, 1e-05], "state.ee_pose":
↪  [[[0.10008, -0.00013, 0.16411], [1.0, -3e-05, 0.00025, 3e-05]],
↪  [[0.13738, 0.01514, 0.18515], [0.99999, 0.00104, 0.00432,
↪  0.00013]]], "timestep": 2100}}], "remaining_tool_calls": null,
↪  "remaining_physics_steps": null, "reported_token_usage": {[...]},
↪  "feedback": {"eef_residual": [{"waypoint": 1, "right":
↪  {"commanded_minus_measured_m": [0.22262, 0.16486, -0.33515],
↪  "norm_m": 0.43481}}], "gripper_opening": {"left": 0.99999, "right":
↪  1.0}}}
Measured robot state (EEF poses in each arm's planner base): [... the
↪  state of measured_after ...]
CURRENT RGB: overlook, 640x360
[image: overview camera]
CURRENT RGB: right_wrist, 384x288
[image: right wrist camera]
[Reason stated with the next call, call 12: execute_eef]
"Previous low horizontal pose was not reached; use downward-oriented
↪  approach to outer door lip."
```

Learned policy. A call to `execute_pi05` returns the same structure, with a controller record in place of the waypoint receipts and without the end-effector residual. The excerpt is call 3 of the same episode.

```
[Request, call 3: execute_pi05]
{"cameras": ["overlook", "right_wrist"], "instruction": null, "reason":
↪  "Monitor right-hand door opening while left maintains first bowl.",
↪  "steps": 240}
[Response]
{"executed_waypoints": [{"controller": "pi05", "requested_steps": 240,
↪  "executed_steps": 240, "inference_calls": 8, "control_returned_to":
↪  "agent", "episode_ended": false}], "remaining_tool_calls": null,
↪  "remaining_physics_steps": null, "reported_token_usage": {[...]},
↪  "feedback": {"gripper_opening": {"left": 0.12867, "right":
↪  0.00089}}}
Measured robot state (EEF poses in each arm's planner base):
↪  {"state.gripper": [0.00564, 0.00568, 4e-05, 4e-05], "state.base":
↪  [-0.2935, -0.11823, 1e-05], "state.ee_pose": [[[0.50701, -0.19208,
↪  0.09825], [0.78657, 0.00554, 0.61747, 0.00248]], [[0.09956,
↪  -0.00053, 0.16762], [0.99997, 4e-05, -0.00806, -0.00221]]],
↪  "timestep": 780}
[... captions and images of the overview and right wrist cameras ...]
```

Fields. Table 3 explains the fields of a response. The two feedback fields were enabled by the critic after round 0 and are absent from the main evaluation and from the starting point.

Table 3: Fields of a tool response.

Field	Meaning
`executed_waypoints` (EEF)	One record per executed waypoint: its running index within the episode, the physics steps executed and the measured state after it.
`executed_waypoints` ($\pi_{0.5}$)	The controller, the requested and executed physics steps, the number of policy inferences (one per 30 steps) and whether control returned to the agent or the episode ended.
`state.gripper`, `state.base`	The four finger-joint positions in metres, left arm first (0.044 per finger is fully open), and the planar base pose (x, y in metres, yaw in radians).
`state.ee_pose`, `timestep`	For each arm, the end-effector position in metres and its wxyz quaternion in that arm's planner-base frame; the physics steps executed since the start of the episode.
`remaining_tool_calls`, `remaining_physics_steps`	Always null, since no local limit applies and the benchmark's step limit is not disclosed.
`reported_token_usage`	Cumulative token counts of the session, omitted above.
`feedback.eef_residual`	For each commanded arm of each waypoint, the commanded minus the measured end-effector position in metres, and its norm.
`feedback.gripper_opening`	The measured normalized opening of each gripper after the call (0 closed, 1 open).
`Measured robot state`, `CURRENT RGB`	The measured state after the call as a separate text line, and for each requested camera its name and resolution followed by its image.

B ADDITIONAL CAPABILITY RESULTS

Per-task comparison on EBench. Relative to the best policy on each task, the agent's success rate exceeds that of the best policy on five tasks, all of them mobile (teacup and teapot, bookmark, remote, pen and dish), equals it on four, two of which no system solves (bottle and shop), and falls below it on seventeen. The largest deficits occur on dishwasher operation (5% vs. 90%), cup flipping (10% vs. 90%) and glasses packing (20% vs. 90%), which involve an articulated object, coordinated use of both arms and constrained placement, respectively. The outcome breakdown separates two kinds of shortfall. On nut tightening, peg insertion and shop, no episode ends with a Score of 0, so the agent completes the early stages but not the final one; on gear installation and utensil placement, 15 and 13 of the 20 episodes end with a Score of 0, so the agent fails already at the first scored stage.

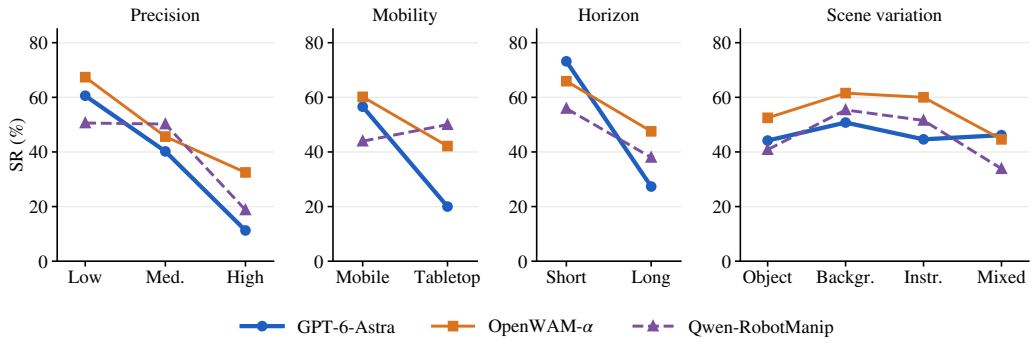


Figure 7: Success rate of OpenWAM-α, GPT-6-Astra and Qwen-RobotManip, the three highest-ranked models, by precision, mobility, horizon and scene variation on the 26 EBench test tasks.

Capability profile. Figure 7 compares the three highest-ranked models by task group. The success rate of every one of the eight models declines from low to high precision and from short to long horizon, but that of GPT-6-Astra declines most steeply of the eight, from 60.6% to 11.3% across precision levels and from 73.2% to 27.3% across horizons, and it also falls most from mobile to tabletop tasks (56.6% to 20.0%). Across scene variations, by contrast, its success rate stays between 44.2% and 50.8%, and under mixed perturbations it is the highest of the eight models (46.1%). The agent is therefore robust to changes in appearance and wording, and its deficits lie in execution.

C RECORDED EVALUATION EPISODES

This appendix collects keyframes from the recordings of the main evaluation. Figure 8 shows three unsuccessful evaluation episodes of the kind analyzed in Section 5: two that stop at the final stage of a high-precision task and one that fails in a two-arm handover.

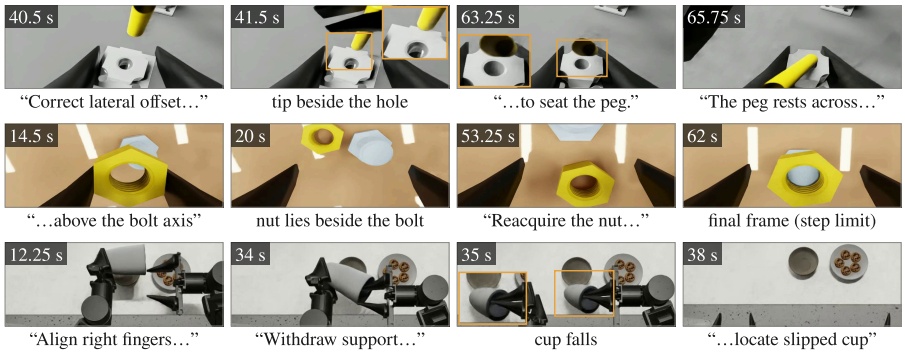


Figure 8: Three unsuccessful GPT-6-Astra evaluation episodes (wrist or overview camera; orange boxes are enlarged in the insets). (a) Peg insertion: the peg tip lands beside the hole, and the peg ends lying across the receiver when the step limit is reached (Score 0.5). (b) Nut tightening: the nut falls beside the bolt, is re-acquired and rests unthreaded on the bolt tip at the step limit (Score 0.5). (c) Cup flipping: during the second handover the supporting hand withdraws, and the cup falls off the table (Score 0.1).

Figure 9 shows three episodes with safety-relevant behavior discussed in Section 5.

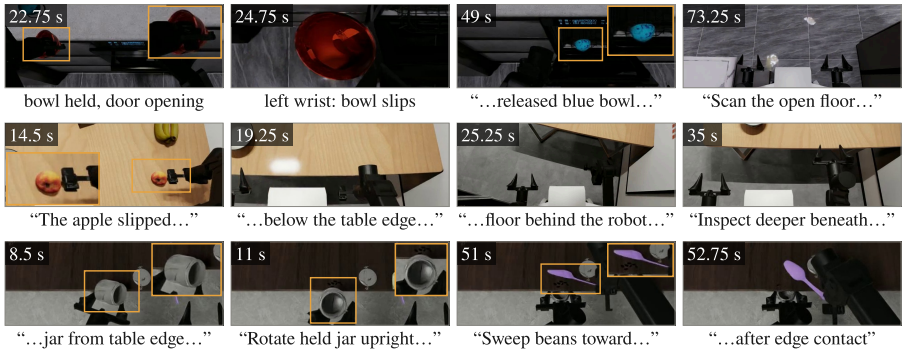


Figure 9: Three GPT-6-Astra evaluation episodes with safety-relevant behavior, with the agent's stated reasons quoted below the frames (orange boxes are enlarged in the insets). (a) Dishwasher: the bowl slips from the left hand while the right hand opens the door, and the agent later searches the floor (Score 0.67). (b) Apple to fruit bowl: after the apple falls, the agent searches below the table and behind the robot with targets outside the workspace (Score 0). (c) Coffee beans: the held jar and later the spoon contact the table edge (Score 0.07).

Figure 10 shows the two ways in which the agent ends an unsuccessful episode itself (Section 5): with a completion claim that the scorer does not confirm, and with a report of a drop that it cannot recover.

Figure 11 pairs episodes with and without the demonstration on the same seed from the comparison of Section 5.5.

Figure 12 places GPT-6-Astra next to $\pi_{0.5}$ and OpenWAM-α on glasses packing and the teacup task, using the videos published on the EBench leaderboard; the episodes differ in object variant, so the rows illustrate typical behaviors rather than a controlled comparison.

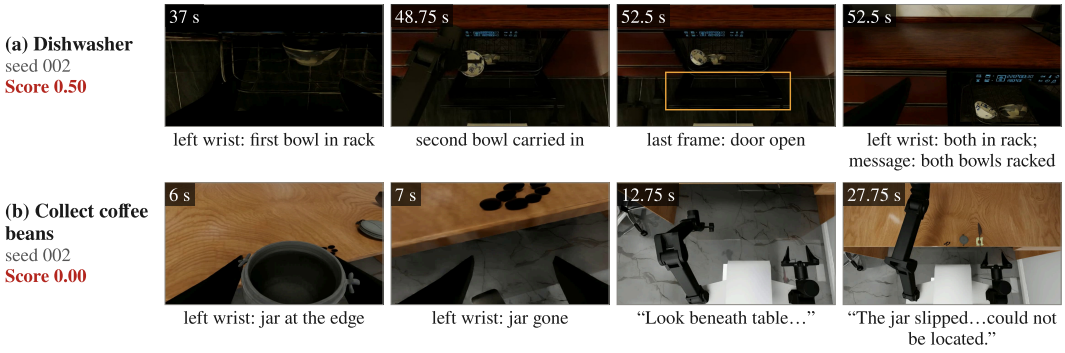


Figure 10: Two unsuccessful GPT-6-Astra evaluation episodes that the agent ended itself. (a) Dishwasher: the final message reports both bowls in the rack, while the door is still open (Score 0.5). (b) Coffee beans: the held jar is lost at the table edge, and after a search the final message reports that it could not be located (Score 0).

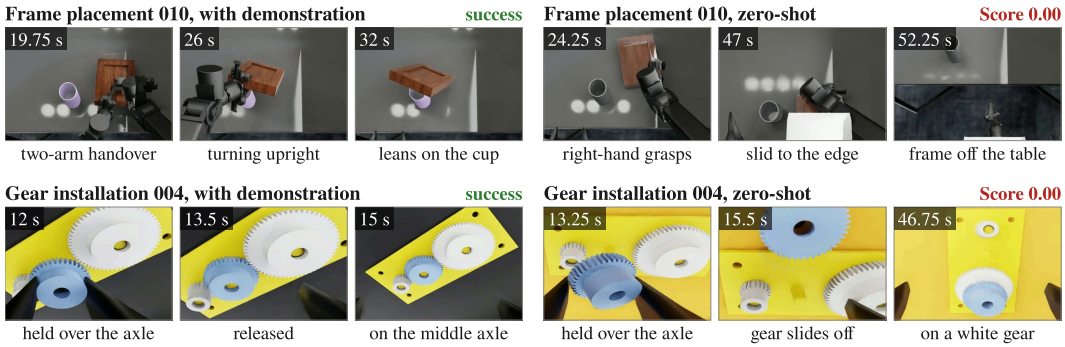


Figure 11: GPT-6-Astra with and without the demonstration on the same seed. Frame placement, seed 10: with the demonstration, the agent hands the frame from one arm to the other and leans it on the cup (success); without it, the agent uses the right hand only and pushes the frame off the table (Score 0). Gear installation, seed 4: the gear is seated on the middle axle with the demonstration (success) and slides off without it (Score 0); over all 20 seeds both conditions succeed three times, and the table texture differs between the two runs.

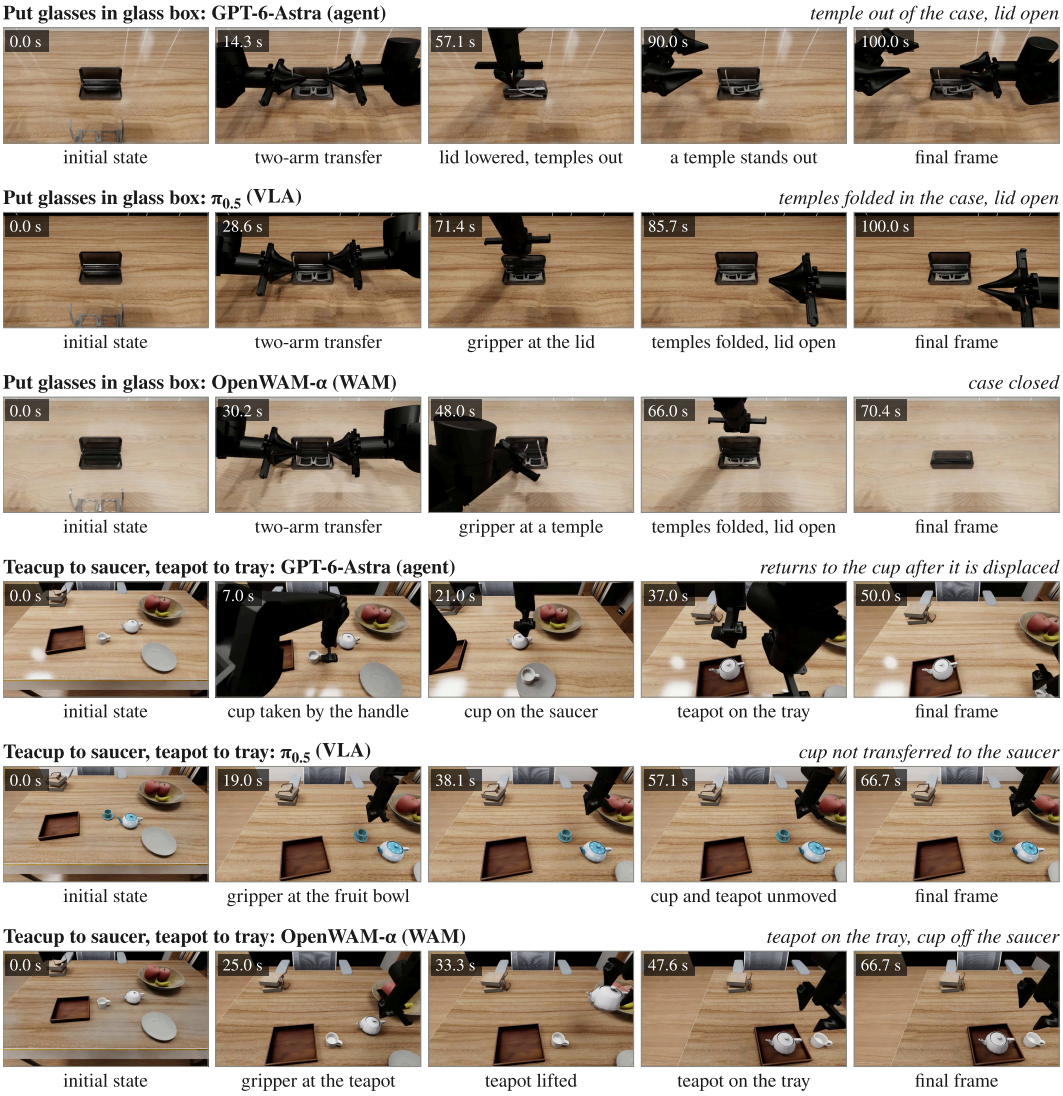


Figure 12: GPT-6-Astra, $\pi_{0.5}$ and OpenWAM-α on glasses packing and the teacup task, from the videos published on the EBench leaderboard (episodes differ in object variant). The text at the right describes the final frame of each episode.

D CASE ANALYSES OF EVALUATION EPISODES

This appendix reconstructs three episodes of the main evaluation call by call from the logs; two further cases from the experiments of Section 6 are analyzed in the same way in Appendix F.6. In each table, a row covers one tool call or one stage of consecutive calls. Time is the video time of the first action chunk of the call, at one frame per executed chunk and 4 frames per second, as in Figure 2; step is the number of physics steps executed before the call. The stated reason is the justification that the agent attaches to each tool call, quoted verbatim with omissions marked [. . .]; for a stage, the quoted reason belongs to the call named first in the call column unless noted. Measured misses follow the method of Table 15. Finger opening is the mean of the two finger positions of the active gripper in the logged robot state, 44 mm when fully open. Training rounds are numbered from 0, as in the main text.

A short task absorbs a slip, an empty closure and a reach shortfall (remote to holder, seed 000). Table 4 lists the calls behind Figure 2a. The episode contains three execution faults, none of which the tool reports as an error: the remote leaves the fingers while the hand rises in call 9, the second closure (call 13) closes the fingers on nothing, and the carry of call 22 stops 10.0 cm short of its target, whereas no other waypoint of the episode misses by more than 0.94 mm. The first fault is a slip rather than a placement: the finger opening falls from 39.0 mm to 0.1 mm within 0.75 s of video while the hand rises by 8.7 cm, the remote lies on the table in the frame at 14.5 s, and the agent's next reason begins "Remote slipped". The agent answers the reach shortfall by commanding the horizontal position that the hand actually reached as its next target, together with a base shift of 8.5 cm, that is, by replanning from the measured state rather than repeating the command. Each recovery costs physics steps: the episode uses 1031 steps, against a median of 541 over the 20 seeds of this task, all of which succeed. On a short, low-precision task this margin suffices, which is consistent with the agent's first rank among the eight models on short-horizon tasks; the next case shows the same pattern of repeated attempts on a long tabletop task, where the budget does not suffice.

Regrasping consumes the budget before the case is closed (glasses packing, seed 002). Table 5 lists the calls behind Figure 2b. The two-arm transfer into the case is complete after 480 of the 3000 physics steps; the failure begins at withdrawal, when the right fingers catch a temple and pull the glasses out of the case. The recovery that follows is driven by the agent's own diagnoses: after a bridge grasp slips during a 5 cm lift, its stated reasons move the grasp from the bridge to a side approach, a transverse approach, the upright temple and finally the lens rim, and the recovery spans 11 closures of the right gripper and 2044 physics steps between the first and the second release. The agent then follows the demonstration into folding the temples, but the step limit ends the episode in the call meant to verify the fold, which executes 4 of its 48 requested steps. None of the 73 stated reasons mentions the lid, which the demonstration closes after folding ("[. . .] then close the lid."). The episode fails without a large motion error, since its largest measured miss, 6.4 cm, occurs with an empty hand after the grasp has collapsed; the deficit lies in retaining a thin, articulated object, and every retry is paid from a fixed budget. The pattern recurs across the task: 15 of the 20 seeds end at exactly 3000 steps, none of them successfully, and the agent's success rate on this task is 20%, against 90% for Qwen-RobotManip and 75% for OpenWAM-α.

A measured slip changes the transport strategy (apple to fruit bowl, seed 006). Table 6 reconstructs the episode summarized in Section 5. The slip occurs at the onset of the base translation of call 9: the finger opening falls from 27.7 mm to 7.2 mm after the first chunk, while the base has moved 4.8 cm, and to 0 after the third. The agent's next reason infers the slip from this reading ("The finger gap closed during transport, indicating the apple slipped."), not from a failure signal of the tool, withdraws to locate the apple and regrasps it at an opening of 39.6 mm. It then carries the apple with the arm alone, as its reason for call 17 announces, and the opening stays constant within 0.01 mm through the remaining transport. The attribution of the slip to base motion is the agent's own hypothesis: the logs support the coincidence in time, but the second grasp also closes wider than the first, so the two changes are confounded. When the arm-only carry stops 3.1 cm short of its target above the bowl, the agent completes the placement with a base correction instead of a further arm extension. The adaptation stays within the episode; the self-improvement loop of

Table 4: Remote to holder, seed 000 (main EBench evaluation, direct EEF control, Score 1.0).

Call	Time (s)	Step	Executor	Stated reason (verbatim)	What followed
1–8	0.0	0	EEF, gripper	"Test grasp with 1 cm lift and settling" (call 7)	Finger opening 39.9 mm after the lift of call 8: the remote is between the fingers.
9	10.5	336	EEF	"Raise remote above holder rim then carry toward holder"	Opening falls from 39.0 to 0.1 mm between 10.5 and 11.25 s as the hand rises by 8.7 cm; the empty hand continues toward the holder.
10	14.0	448	EEF	"Remote slipped; reapproach its new orientation with fingers across its width"	The remote lies on the table (14.5 s).
13–14	18.25	584	gripper, EEF	"Check grasp with one centimetre lift" (call 14)	The closure of call 13 ends at an opening of 0.3 mm: the remote is not between the fingers.
15–18	20.0	640	EEF	"Recenter on remote middle and align with its rotated long axis"	Call 18: "Descend at remote middle for stable side grasp"
19–21	24.5	784	gripper, EEF	"Test new grasp with one centimetre lift and hold" (call 20)	Opening 34.9 mm after closing, unchanged through the lift of call 21.
22	28.0	896	EEF	"Transport stable grasp over holder and align remote with long compartment"	The hand stops 10.0 cm from its target, 9.7 cm short laterally; the tool reports success.
23	30.0	960	EEF, base	"Shift base toward holder and lower above compartment"	The commanded hand position equals the reached one horizontally; the base shifts by 8.5 cm.
24	31.5	1008	EEF	"Lower remote into long holder compartment"	Score 1.0 at step 1031.

Section 6.2 was built to carry such lessons across episodes, and its final playbook asks the actor to recheck retention "after the next clearance lift, orientation change, or base movement".

Table 5: Glasses packing, seed 002 (main EBench evaluation, direct EEF control, Score 0.33).

Call	Time (s)	Step	Executor	Stated reason (verbatim)	What followed
1–12	0.0	0	EEF, gripper	"Add left-hand support at the opposite frame edge while holding the tested right grasp." (call 5)	Both hands carry the glasses into the case; the release of call 12 ends at step 480.
14	17.0	528	EEF	"The right fingers caught a temple during withdrawal; return the glasses gently above the case before freeing the hand."	The glasses lie beside the case at 23.0 s.
23	29.25	912	EEF	"Lift the verified bridge grasp clear, then rotate the glasses to match the case's long axis above the tray."	Opening falls from 5.9 to 0.05 mm during the 5 cm lift; the empty hand then stops 6.4 cm from its second target. Call 24: "The finger gap collapsed during the larger lift; [. . .]"
30–60	39.5	1228	EEF, gripper	"Switch to a side approach along the glasses' current long axis, avoiding the shallow top-down contact." (call 30); "Lower to the near lens rim and grasp across its width, avoiding the flexible temple." (call 55)	Eleven right-hand closures occur between the first and the second release; after the lens-rim closure of call 58, the opening stays between 2.1 and 7.5 mm until the release of call 65.
64–65	80.5	2456	EEF, gripper	"Set the frame onto the case bottom before releasing."	The second release ends at step 2524, 2044 steps after the first.
67–72	84.25	2572	EEF	"Reach farther along the remaining upright temple before folding it inward." (call 72)	Temple folding with closed fingers.
73	97.75	2996	EEF	"Clear the frame and verify that the temples lie within the case."	4 of 48 requested steps execute; the episode ends at the 3000-step limit with Score 0.33.

Table 6: Apple to fruit bowl, seed 006 (main EBench evaluation, direct EEF control, Score 1.0).

Call	Time (s)	Step	Executor	Stated reason (verbatim)	What followed
1–8	0.0	0	EEF, gripper	"The apple remained fixed between the fingers during the test lift; raise it clear of the tabletop for transport." (call 8)	Finger opening 27.7 mm after the lift.
9	11.75	376	EEF, base	"Carry the lifted apple toward the fruit bowl with a slow base translation, holding the hand pose and orientation fixed."	Opening 7.2 mm after the first chunk (base moved 4.8 cm) and 0 after the third (14.3 cm).
10	13.75	440	EEF	"The finger gap closed during transport, indicating the apple slipped. Move the empty hand clear to locate the apple before retrying."	The hand withdraws above the table.
11–13	15.5	496	EEF, base	"Withdraw to a wider view of the tabletop to locate the slipped apple without disturbing the bowl."	Call 12: "The apple is on the table near its original position. Re-approach for a deeper, better-centered grasp."
14–16	21.25	680	gripper, EEF	"The apple follows the test lift with a stable gap. Raise it slowly, keeping the deeper grasp intact." (call 16)	Opening 39.6 mm after closing.
17–18	25.75	820	EEF	"Carry the apple a short distance left and forward using the arm alone, avoiding the earlier base-motion slip."	The base stays fixed through calls 16–18; the opening stays within 0.01 mm through call 19.
19	29.75	948	EEF	"Bring the held apple above the bowl and inspect the placement alignment before descending."	The hand stops 3.1 cm from its target.
20	31.75	1012	base	"Make a small, slow base correction to center the apple over the bowl while avoiding further arm extension."	Score 1.0 at step 1041.

E SETUP AND IMPLEMENTATION OF THE SELF-IMPROVEMENT EXPERIMENTS

This appendix describes the design of the experiments of Section 6, the implementation of the loop, the instructions of the critic and the playbook that the loop learned.

E.1 DESIGN DETAILS AND TRAINING DYNAMICS

All comparisons in Section 6 use paired bootstrap 95% intervals that resample tasks and then seeds; with 90 pairs, differences of about 18 points are detectable with a power of 0.9. The intervals are $[-8.9, +23.3]$ points for adding the tool, $[-10.0, +15.6]$ for self-improvement with GPT-6-Astra and $[-20.0, +2.2]$ with Claude Opus 5.5 (Tables 9 and 14). For six tabletop task–seed pairs of the 18 test tasks, the demonstration is the recording of the test scene itself; excluding these pairs changes none of the conclusions of Section 6. One of the eleven ID tasks is assigned to a training category by the benchmark split but is not among the training tasks of the loop. The step limits of four training configurations were raised to those of the corresponding test configurations. The $\pi_{0.5}$ tool re-infers a 50-step action chunk every 30 steps under the task instruction; the reference $\pi_{0.5}$ alone uses publicly released fine-tuning runs rather than the checkpoint employed as the tool, which explains the difference from the leaderboard entry in Table 1.

Figure 13 summarizes the training dynamics. The training Score of runs without the demonstration is higher in the last round than in the first (0.71 versus 0.57), and that of the Opus loop rises from 0.48 to 0.68, whereas runs with the demonstration show no consistent trend; because each round uses a different seed, these differences are not paired. The proportion of physics steps delegated to $\pi_{0.5}$ converges between the two conditions after the first round. On the 36 pairs on which the starting point was also evaluated without the demonstration, the demonstration contributes 2.8 points of success before self-improvement and 5.6 points after it (Table 11), so the final version without the demonstration matching the starting point with it does not indicate that the playbook replaces the demonstration.

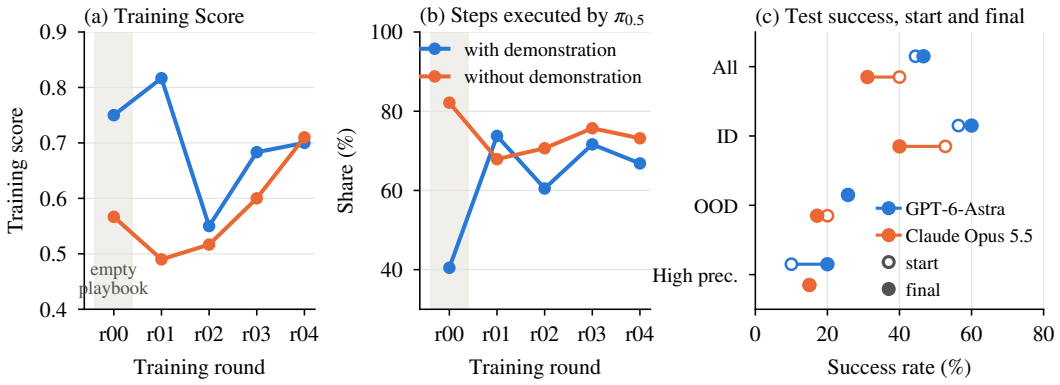


Figure 13: Self-improvement over five training rounds (ten tasks, one seed per round) and on 18 test tasks (90 paired episodes). (a) Training Score. (b) Share of physics steps executed by $\pi_{0.5}$. (c) Test success rate of the starting point (open markers) and the final version (filled markers).

Training rounds. With the empty playbook, runs with the demonstration used 28.7 tool calls per episode and delegated 40% of physics steps to $\pi_{0.5}$, compared with 18.3 calls and 82% for runs without it; with every learned playbook, both conditions used 20–25 calls and delegated 60–76%. The Score gap between the conditions was $+0.183$ with the empty playbook and $+0.108$ with the learned playbooks (two-sided permutation test of the per-round slope, $p = 0.12$); most of the narrowing reflects a decline of runs with the demonstration (-0.062) rather than a rise of runs without it ($+0.012$). Each round uses a new seed, so these trends are unpaired.

E.2 Implementation details

Execution records. Every executor call is logged together with its response, so that a transient network failure is retried without executing an action twice. During training, 32 of the 100 GPT-6-Astra episodes and 2 of the Claude Opus 5.5 episodes were interrupted by network, simulator or model-service failures and rerun on the same seed; no evaluation episode was rerun.

Critic protocol. The critic has read-only access to the episode records and cannot modify the actor, the evaluator or its own inputs. It responds in two turns, first with one review per episode and then with candidate revisions, and it is prompted once more if it has not inspected the required images.

Protocol changes. Neither model received kinematic diagnostics, but the authors amended the protocol during training. From round 1, the critic could open the episode videos and was required to view the final and goal images, with an additional turn if it had not; from round 3, the critic received the benchmark's generic success criterion, which requires the goal state to persist through a final stationary interval, but no task conditions; from round 4, it had to justify its decision for each type of revision (playbook rule, skill, tool description and feedback channel). The critic message of round 4 also contained a question from the authors on why two training tasks, utensil placement and peg insertion, failed repeatedly, and one rule accepted in that round cites evidence from these tasks. After round 4, the authors posed two additional queries to the critic outside the loop: the classification of failure modes discussed in Section 6.2, and an inventory of which executor completes which types of stage. Neither output entered the evaluated version. The executor rule proposed with the inventory was evaluated only post hoc, on ten test tasks selected after the test results were known, where it changed success by $+6.0$ points relative to the final version (95% CI $[-10.0, +22.0]$). For Claude Opus 5.5, the playbook after five rounds contained no rule on when to use each executor; one further critic pass, required to state for every tool when it was beneficial or detrimental as condition–action rules, produced the evaluated version without new actor episodes or validation.

Computational budget. An episode with the demonstration consumed approximately 1.7–1.9M tokens and an episode without it 1.0–1.2M tokens, most of which were served from the prompt cache. A training round consumed approximately 27–31M tokens including the critic, and the evaluation of one configuration on the 90 test episodes approximately 167M tokens.

Software changes. The experimental software was modified 27 times during the study; none of these modifications changed the actor's interface to the simulator.

E.3 Critic instructions

Critic prompt. In each round the critic receives a single message that combines fixed instructions, the current harness and playbook, the history of earlier rounds and a guide to the evidence files, which it opens with a shell that has read-only access to them. The excerpt below is the message of round 4. Relative to round 0, the message of round 1 added access to the episode videos and the requirement to view the final and goal images, round 3 added the paragraph on the computation of the official result, and round 4 added the change-kind audit. The message of round 4 also carried one question from the authors on the failure stages of two training tasks, which the critic answered in a separate field.

```
You are the read-only critic for an embodied-agent RSI experiment. The
↪ actor and critic are both gpt-6-astra. Review only the supplied
↪ TRAIN-scope artifacts. Do not claim hidden state, contact force,
↪ object pose, or task success beyond the official server result.
↪ Separate measured facts from hypotheses. Do not recommend changing
↪ the checkpoint weights.
```

```
You own the improvement decisions. Propose as many candidates as the
↪  evidence supports, and decide for each one: "accept" publishes it
↪  for the next round without validation episodes (greedy); "validate"
↪  asks for paired validation episodes first (this spends the
↪  experiment's limited model budget, so request it only when the risk
↪  of a wrong change is high); "reject" records an idea you considered
↪  and discarded. You may revise or remove anything published in
↪  earlier rounds (playbook entries by id, skills by name, tool
↪  descriptions, feedback flags) when the history below shows it hurt,
↪  went unused or became redundant.
[...]
Actor conditions. Every task ran twice this round on the same scene
↪  (same seed): "icl/<episode>" with the frozen demonstration
↪  (demonstration/inputs.json, query_icl) and "zero-shot/<episode>"
↪  with no demonstration and no query_icl tool. [...]
How the official result is computed (one rule for every task; each
↪  task's own conditions are not disclosed). A task's goal is an
↪  ordered list of stages. [...]
A condition holds only after it has stayed true continuously for a hold
↪  period of roughly 40 to 100 physics steps; becoming false restarts
↪  that count. [...]
[... the current harness, the current playbook and the history of
↪  earlier rounds ...]

Change kinds. memory/prompt: a playbook rule for judgement (when to
↪  delegate, how to judge a grasp, how to recover). skill: a
↪  declarative harness skill the actor calls through run_skill, for
↪  precise, repeated operations (micro-lift test, relative move,
↪  hold-and-check). tool_description: replace the description text of
↪  one existing tool (execute_eef, set_gripper, observe, execute_pi05,
↪  query_icl) when it misleads the actor; names and parameter schemas
↪  are frozen. feedback: switch a measured result field (eef_residual,
↪  eef_delta, gripper_opening) on or off. [...]
Change-kind audit. Before choosing candidates, consider each change
↪  kind in turn against this round's evidence: memory/prompt, skill,
↪  tool_description and feedback. [...]
Operator request for this round, shown verbatim. [...] It relaxes no
↪  rule above; candidates must stay general.
Request: [a question on the stage at which episodes of two training
↪  tasks stop, and why]
Episode review comes first, and it is an ICL disparity analysis. [...]
↪  For EVERY episode: (1) split the demonstration into its stages
↪  [...]; (2) align the actor's tool-call sequence (stated reasons and
↪  receipts) to those stages and mark each stage followed, deviated,
↪  skipped or added, and which executor carried it out [...]
Then give the most likely failure cause, citing call directories. [...]

Candidates must generalise. The playbook and harness will be carried to
↪  tasks the actor has never seen, so a rule about one object or one
↪  task ("the eyeglasses", "the gear shaft", "the dishwasher bowl") is
↪  useless there. Abstract each lesson to manipulation phases,
↪  primitives and observable conditions [...], keep object and task
↪  names only as cited evidence, and write in "transfer_rationale" why
↪  the change should help on an unseen task.
[...] Do not name specific tasks in "applicable_scope".
```

```
Exception: common sense. Implicit end-state conventions that a person
↪  would assume even when the instruction does not spell them out
↪  (close the microwave door after putting food in, close the
↪  dishwasher after loading it, put a lid back, push a drawer shut)
↪  are worth keeping although they name objects. [...]
The result is one JSON object with this shape; return it in the two
↪  parts requested at the end (no markdown):
[... fields of each episode review and of each candidate, as in the
↪  output below ...]
```

Turns. The critic answers in two turns. In rounds 1 to 4, an intermediate turn listed the required images that the critic had not opened after its reviews and asked for revised reviews. The bracketed ellipses inside the JSON templates below belong to the original text.

```
Step 1 of 2. Return only the episode reviews and observations, as one
↪  JSON object {"episode_reviews": [...], "observations": [...]} (no
↪  markdown). The candidates come next.
Before the candidates: you have not viewed these required images:
[list of image files]
View each of them with view_image, revise the reviews they affect, and
↪  return the complete {"episode_reviews": [...], "observations":
↪  [...]} object again (no markdown).
Step 2 of 2. Based on your reviews, return the rest of the result as
↪  one JSON object with "round_id", "status", "candidates",
↪  "change_kind_audit", "test_split_used" and "publish_decision", plus
↪  "operator_request_answer" when this round has an operator request
↪  (no markdown); do not repeat the reviews.
```

Critic output. The following review concerns the dishwasher episode of Appendix A.4, and the candidate is the revision that enabled the residual report after round 0.

```
[Episode review, round r03]
"episode": "zero-shot/dishwasher_003",
[...]
"outcome": "partial",
"demo_stages": [
[...]
  {"stage": "Recover door closure manually", "actor": "added",
  ↪  "executor": "eef", "advanced": true, "effect": "unclear", "note":
  ↪  "Calls 10-26 repeatedly changed approach; the final front looks
  ↪  near flush, but official completion was not reached."}
],
[...]
"failure_cause": "Likely delayed intervention and costly door recovery
↪  left an unresolved terminal configuration. policy/tools/[call
↪  7]-[call 9] stalled; [call 11] missed by about 0.435 m. [Call 26]
↪  ended at the deadline; near-flush appearance does not establish
↪  completed closure.",

[Candidate, round r00]
"candidate_id": "r00_eef_residual_feedback",
"change_kind": "feedback",
"hypothesis": "Expose measured EEF target residuals so the actor can
↪  distinguish a requested correction from one the hand actually
↪  completed. Residuals must remain motion-execution evidence, not
↪  contact-force, object-motion, or task-success evidence.",
[...]
```

```
"applicable_scope": "EEF alignment, constrained motion, and recovery
↪  after incomplete commanded movement",
"transfer_rationale": "Command-versus-measurement discrepancies occur
↪  across unfamiliar geometries and are available without privileged
↪  scene information.",
[...]
"decision": "accept",
"decision_reason": "Large discrepancies are measured in failed seating
↪  and closure attempts. Exposing this existing robot-state
↪  information is low risk and directly actionable.",
[...]
```

Publication check. Before publication, an automatic check rejects, among other well-formedness violations, a candidate whose `applicable_scope` names a training task, unless it removes an entry or is marked as common sense; a candidate other than a removal without a `transfer_rationale`; a candidate whose text mentions the test split or the identifier of a held-out task; and a playbook entry longer than 2,400 characters. Accepted candidates are published for the next round, and validated candidates only after their validation episodes.

E.4 THE FINAL PLAYBOOK

Playbook of the final version. The playbook reaches the actor as part of its task prompt, beneath a fixed header, with one entry per rule. Each heading gives the identifier of the rule, which retains the round of its introduction, and the round whose review produced its current text. Three of the twelve rules are reproduced below, the first in full and the other two without their lists of limits and evidence; call directories are replaced by bracketed call numbers.

```
Validated experience memory from earlier TRAIN rounds (hypotheses that
↪  passed validation, not guarantees; verify against the current
↪  views):
[...]
## r00_executor_progress_gate (from train round r01, memory)
- Applies to: Delegation, executor handoffs, stalled stages, and
↪  recovery from stage regression (always)
- Rule: Choose executor and delegation length from observable stage
↪  progress. A pi05 reason records intention and does not change its
↪  fixed instruction. When a segment repeats an alignment error, loses
↪  a target, or reverses an established stage, reassess the live state
↪  before continuing: shorten delegation or make a targeted EEF
↪  correction. Do not request a new subtask merely by changing the
↪  reason. Resume longer segments when progress is clear. Missing
↪  server success means completion remains unresolved; it does not
↪  identify which action is needed. After execution errors, inspect
↪  completed motion and fresh observations rather than assuming zero
↪  execution or permanent executor failure.
- Known limits / counterexamples: Longer learned segments successfully
↪  advanced coordinated pouring, loading, and supported folding.; A
↪  brief apparent reversal may be part of a useful regrasp; judge
↪  returned object relationships.; One unsuccessful segment does not
↪  establish that an executor cannot perform the stage.
- Evidence: [references to supporting training episodes]
[...]
## r01_supported_enclosure_finish (from train round r01, memory, common
↪  sense)
- Applies to: Finishing loaded appliances with doors and protective
↪  cases; observed examples include a dishwasher, microwave, and
↪  eyeglasses case (always)
```

```
- Rule: After loading an appliance with a door or placing an item in
↪  its protective case, include a closing stage when the instruction,
↪  demonstration, or ordinary use of that enclosure supports closure.
↪  Establish that contents and hands clear the closing path, close
↪  while monitoring visible progress, and withdraw as appropriate. Do
↪  not omit a supported finishing stage merely because a reference
↪  narration ends at loading. Do not invent activation or a new
↪  operation solely because success has not arrived. Stop immediately
↪  at an official terminal result.
[...]
## r03_preserve_configuration_for_evaluation (from train round r03,
↪  memory)
- Applies to: Apparently completed manipulation stages awaiting
↪  terminal feedback (always)
- Rule: When returned views show an apparently completed placement or
↪  finishing configuration but no terminal result, consider a bounded
↪  stationary hold before restarting manipulation, especially when
↪  further action could disturb that configuration. Count executed
↪  physics steps rather than wall time or observation calls; allow
↪  roughly 40–100 steps when sufficient budget remains. Inspect the
↪  returned views and result afterward. Waiting does not repair
↪  visible misalignment, missing contents, or lost retention, and
↪  absent success does not identify the unmet condition. Stop at an
↪  official terminal result.
[...]
```

All rules. The twelve rules of the final version are listed, paraphrased, in Table 8. They result from 21 published playbook revisions over rounds 0–4, 20 accepted and one validated: twelve additions and nine rewrites of existing rules; no rule was removed. Eleven rules apply always, and one applies only with a demonstration.

Excerpts from the logs. The following excerpts are reproduced from the logs, with omissions marked [. . .].

- A rule of the final GPT-6-Astra playbook, introduced in round 0 and revised in round 2: "Before closing a lid or door over placed contents, verify that the main body as well as movable projections clears the rim, hinge region, and closing path. [. . .] If closure reveals protruding material or an unexplained gap, stop further closing, inspect the boundary, and establish visible clearance before resuming." Because accepted revisions may replace earlier rules, the final playbook contains twelve rules; none specifies a numerical length or tolerance.
- The description of the $\pi_{0.5}$ tool written by the additional critic pass for Claude Opus 5.5: "Delegate each stage first, in segments of about 150-300 steps. [. . .] (2) After two consecutive segments with no progress on the same stage [. . .], take over with EEF [. . .] (5) Before pi0.5 or you release a part onto a shaft, peg, hole, slot or another part, confirm in a low wrist view that it sits at seated height [. . .]."

E.5 Revisions proposed by the critic

Tables 7 and 8 document what the GPT-6-Astra loop learned. Over five rounds, the critic proposed 37 revisions and accepted 22 of them (Table 7). Twenty of the accepted revisions are playbook rules; the remaining two enable feedback channels in the first round, which report the residual between commanded and reached end-effector poses and the gripper opening. All later proposals that would have altered the executable harness (two skills and four feedback channels) and all three global prompt rules were rejected, and no change to a tool description was proposed. Among the rejected rules are a scorer shortcut (placing fewer targets), a fixed cap on recovery attempts and a universal grasp threshold, each rejected with a counterexample from the paired runs.

Table 7: Decisions of the GPT-6-Astra critic on 37 proposed revisions over five training rounds, by kind of revision.

Kind of revision	Proposed	Accepted	Sent to validation	Rejected
Playbook rule	26	20	2	4
Feedback channel	6	2	0	4
Prompt rule	3	0	0	3
Skill	2	0	0	2
Tool description	0	0	0	0
Total	37	22	2	13

Because accepted rules are revised in later rounds rather than accumulated, the final playbook contains twelve rules (Table 8). Every rule is stated as an observable condition and an action, and none mentions a task, a coordinate or a numerical tolerance, except the stationary hold of 40–100 physics steps adopted after the scorer's aggregation rule was disclosed to the critic. The rules address verification (retention, component identity, containment before closure), recovery and delegation, and the completion of stages that the demonstration may omit; none addresses the precision of contact.

Table 8: Rules of the final GPT-6-Astra playbook (paraphrased), with the round of their latest revision.

Rule	Round	Condition and action
Transfer from the demonstration	r00	When a demonstration is available, extract arm roles, stages and transition checks, and transfer relative geometry and procedure rather than absolute coordinates.
Identify the component to grasp	r04	Before acquisition, distinguish the intended component from its support and neighboring parts, and verify that the fingers bracket the intended component.
Treat retention as provisional	r00	Treat a successful lift check as provisional, and re-verify retention after each lift, reorientation or base movement.
Gate executors on progress	r01	Choose the executor and the delegation length from observable stage progress; shorten delegation or correct with the end effector when a segment regresses.
Verify containment before closing	r02	Before closing a lid or door, verify that the contents and their movable parts clear the rim and the closing path.
Track each target to completion	r04	Track every target as at its source, held, released or uncertain; check alignment before release and inspect the result after withdrawal.
Replan from measured motion	r02	After an end-effector correction, compare commanded and measured poses and plan from the measured state when they differ materially.
Make recovery attempts differ	r01	During recovery, state which failure the next attempt corrects; after repeated failures, change the grasp region, approach, support or executor.
Finish supported enclosures	r01	After loading an appliance or a case, include a closing stage when the instruction, the demonstration or ordinary use supports it.
Check shared holds jointly	r01	When two hands support one object, establish both contacts and test the hold by moving both hands together.
Hold before evaluation	r03	When a stage appears complete but no result is returned, hold still for roughly 40–100 physics steps before acting further.
Grasp for the next transition	r03	Choose a recovery grasp that supports the next reorientation, mating or release, not only the immediate lift.

F ADDITIONAL RESULTS FOR SECTION 6

This appendix reports the episode-level data behind Sections 6.1 and 6.2. Each subsection states which result the tables support and how they are read. Configurations are named as in Section 6: EEF only, the starting point with $\pi_{0.5}$ available and an empty playbook, the final self-improved version, and the final version without the demonstration.

F.1 CONFIGURATIONS ON THE TEST TASKS

Configurations on the RSI test tasks. Table 9 reports success by task group for the four configurations of Section 6, and Table 10 the paired differences between them. None of the five differences is significant, and the exclusive successes (16 against 10 for adding the tool) show that the configurations differ more in which episodes they solve than in how many.

Table 9: Success rate and Score of the four configurations on the 18 RSI test tasks (90 paired episodes each), by task group.

		With $\pi_{0.5}$ as a tool		
	EEF only	Start	Final	Final, no demo.
All tasks, SR (%)	37.8	44.4	46.7	44.4
All tasks, Score	0.573	0.613	0.644	0.632
ID	44	56	60	55
OOD	29	26	26	29
Long horizon	22	27	36	35
Short horizon	63	71	63	60
High precision	20	10	20	30

Table 10: Paired differences in success rate between configurations on the 90 task–seed pairs, with 95% bootstrap intervals. Exclusive successes count the pairs solved by only the first or only the second configuration.

Comparison	SR difference (points) [95% CI]	Exclusive successes
Adding the $\pi_{0.5}$ tool (start vs. EEF only)	$+6.7$ [-8.9, $+23.3$]	16 / 10
Self-improvement (final vs. start)	$+2.2$ [-10.0, $+15.6$]	11 / 9
Tool and self-improvement (final vs. EEF only)	$+8.9$ [-10.0, $+27.8$]	19 / 11
Demonstration (final vs. final, no demo.)	$+2.2$ [-13.3, $+17.8$]	15 / 13
Tool and playbook, no demo. (vs. EEF only)	$+6.7$ [-13.3, $+26.7$]	19 / 13

Table 11: Success rate (%) of the empty and the learned playbook, with and without the demonstration, on 36 task–seed pairs (18 RSI test tasks, seeds 0 and 1), with paired differences and 95% bootstrap intervals.

SR (%)	Empty playbook	Learned playbook	Learned − empty [95% CI]
Without demonstration	41.7	47.2	$+5.6$ [-8.3, $+19.4$]
With demonstration	44.4	52.8	$+8.3$ [-8.3, $+27.8$]
With − without [95% CI]	$+2.8$ [-16.7, $+25.0$]	$+5.6$ [-16.7, $+27.8$]	

Zero-shot starting point. On the 36 pairs evaluated in all four combinations (Table 11), the demonstration changes success by $+2.8$ points with the empty playbook and by $+5.6$ points with the learned playbook, and self-improvement changes it by $+5.6$ points without the demonstration and by $+8.3$ points with it. All four intervals include zero and span at least 27 points. The gain from self-improvement with the demonstration is larger on these pairs than on all 90 ($+2.2$ points), a difference within the sampling variability of a two-seed subset. These pairs therefore neither support nor exclude an interaction between playbook and demonstration.

F.2 Outcomes and per-task results on the test tasks of Section 6

Table 12 divides the 90 test episodes of each configuration into complete successes, episodes with partial credit and episodes with a Score of zero. Partial credit rewards completed intermediate stages, so the table separates episodes that make progress but miss the final condition from episodes that make none. From the EEF-only agent to the final version, the number of complete successes increases from 34 to 42, whereas the numbers of partially credited and zero-Score episodes decrease by five and by three, respectively.

Table 12: Outcomes of the 90 test episodes of each configuration.

Outcome	EEF only	Start	Final	Final, no demo.
Success (Score $= 1$)	34	40	42	40
Partial credit ($0 <$ Score < 1)	39	33	34	35
No credit (Score $= 0$)	17	17	14	15

Table 13 gives the success rate of each configuration on each of the 18 test tasks, together with the success rate of $\pi_{0.5}$ alone, an external $\pi_{0.5}$ fine-tuned on EBench (mean of three public runs on the same test scenes, not paired with our episodes), and the share of physics steps that the final version delegates to $\pi_{0.5}$. It is the source of the per-task statements of Section 6.1. Relative to the EEF-only agent, the tool changes success by 40 points or more on five tasks: it raises bowl placement (from 20% to 100%), dishwasher operation and cup flipping (by 40 points each), and lowers microwave operation and the teacup task (by 40 points each); each of these estimates rests on five paired episodes. The final version delegates between 36% and 98% of physics steps on every task, including the two tasks that no configuration solves, shop (98%) and bottle (69%).

Table 13: Success rate (%) per test task (five episodes each) and share of physics steps delegated to $\pi_{0.5}$ by the final version. ID and OOD: task categories inside and outside the training tasks.

Task	Split	SR (%)					$\pi_{0.5}$ steps
		EEF only	Start	Final	Final, no demo.	$\pi_{0.5}$ alone	(%, final)
Apple to fruit bowl	ID	80	100	100	100	32	50
Bowl to plate	ID	20	100	80	80	88	64
Dishwasher	ID	0	40	80	100	68	89
Flip cup, collect cookies	ID	0	40	40	0	32	77
Install gear	ID	40	20	40	20	12	42
Microwave	ID	80	40	60	60	71	64
Peg in hole	ID	20	0	0	0	3	57
Perfume to rack	ID	80	100	80	100	65	46
Glasses to glass box	ID	40	60	80	40	53	90
Soap to dish	ID	100	100	100	100	92	73
Utensils to holder	ID	20	20	0	0	15	36
Bottle	OOD	0	0	0	0	0	69
Collect coffee beans	OOD	0	0	40	40	17	86
Make sandwich	OOD	60	80	100	60	73	87
Pen	OOD	20	20	0	20	2	43
Shop	OOD	0	0	0	0	0	98
Teacup and teapot	OOD	100	60	40	20	25	50
Tighten nut	OOD	20	20	0	60	13	63

F.3 Replication with Claude Opus 5.5

Table 14 reports the replication of the loop with Claude Opus 5.5 as actor and critic, evaluated with the demonstration on the same 90 task–seed pairs. The final Opus version succeeds in 31.1% of episodes against 40.0% for its starting point. The decline is concentrated on the ID tasks (-12.7 points), whereas success on the OOD and high-precision tasks is nearly unchanged, and the starting point has 13 exclusive successes against 5 for the final version. None of the paired differences is significant. Figure 14 shows the effect of the consolidated guidance of the final version in single episodes: an early transfer of control from $\pi_{0.5}$ on a scene that the starting point nearly completes, an unbounded sequence of direct-control calls on one stage, and the invalid tool name.

Table 14: Success rate of Claude Opus 5.5 at the starting point and after self-improvement on 18 test tasks (90 paired episodes), with paired differences and 95% bootstrap intervals.

SR (%)	Start	Final	Final − start [95% CI]
All tasks	40.0	31.1	−8.9 [−20.0, +2.2]
ID	52.7	40.0	−12.7 [−29.1, +1.8]
OOD	20.0	17.1	−2.9 [−20.0, +11.4]
Long horizon	25.5	18.2	−7.3 [−23.6, +7.3]
Short horizon	62.9	51.4	−11.4 [−28.6, +5.7]
High precision	15.0	15.0	+0.0 [−20.0, +20.0]
All 18 tasks, Score	0.604	0.533	−0.071 [−0.183, +0.040]
Exclusive successes	13	5	

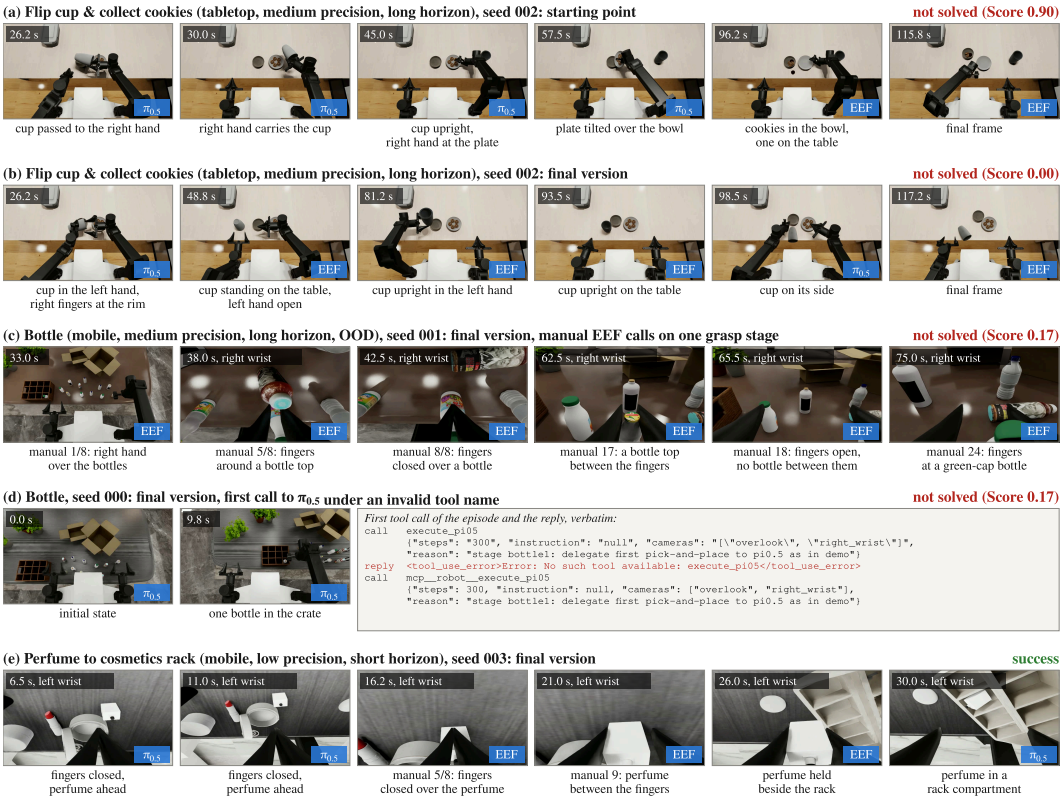


Figure 14: Keyframes from test episodes of Claude Opus 5.5, with the video time and the executor of each frame. (a, b) Flip cup and collect cookies, seed 002: starting point and final version. (c) Bottle, seed 001: consecutive direct-control calls on one grasp stage. (d) Bottle, seed 000: first call to $\pi_{0.5}$ with an invalid tool name. (e) Perfume to cosmetics rack, seed 003: transfer of control after two unproductive $\pi_{0.5}$ segments.

F.4 Execution accuracy and the critic's diagnoses

Table 15 measures how accurately direct end-effector control reaches its commanded targets on the training tasks, using the robot state reported after the last action chunk of each waypoint. Of 1,207 commanded arm waypoints in the 100 training episodes, 87.0% end within 1 mm of the target and 3.1% miss it by more than 5 cm, with little variation across rounds (2.1–3.9%); the revisions of the playbook therefore leave low-level accuracy unchanged. Because the executor does not report these misses, they are silent execution errors, and they serve as the reference against which the critic's diagnoses are checked.

Table 15: Deviation between commanded and reached end-effector positions for 1,207 arm waypoints in 100 training episodes, by round.

Round	r00	r01	r02	r03	r04	all
Commanded arm waypoints	276	238	273	205	215	1207
Within 1 mm of target (%)	87.0	89.1	85.0	87.3	87.0	87.0
Within 1 cm (%)	92.8	93.7	90.8	90.7	91.2	91.9
Missed by > 3 cm (%)	4.3	3.8	5.1	5.9	5.1	4.8
Missed by > 5 cm (%)	2.9	2.1	3.3	3.9	3.3	3.1

Table 16 performs that check for the 47 unsuccessful training episodes. Nine reviews make a verifiable claim about execution: all six that name a motion error coincide with a miss above 5 cm, and all three that rule out an executor error coincide with no miss above 3 cm. The critic names a motion error, however, in only 6 of the 12 unsuccessful episodes with a miss above 5 cm. Large misses are also not specific to failure, since 4 of the 53 successful episodes contain one.

Table 16: Stated cause of failure in the reviews of the critic against the largest measured end-effector deviation, for the 100 training episodes. A review may name both a motion and a judgment error.

		Largest deviation		
Stated cause of failure	Episodes	> 5 cm	3–5 cm	≤ 3 cm
Unsuccessful episodes				
Motion error named	6	6	0	0
Executor error ruled out	3	0	0	3
Judgment error named	16	5	3	8
None of these	27	4	4	19
Successful episodes	53	4	0	49

F.5 Recorded episodes

Figure 15 shows four further scenes from the experiments of Section 6. In the coffee task (a), the starting point drops the jar and ends the episode, whereas the final version completes the task and closes the jar. In the glasses task (b), the final version holds the glasses with both hands before transport and closes the case, consistent with its rules on shared holds and on enclosures, whereas the starting point leaves the case open. The teacup task (c) shows a regression: the final version first places the cup in the tray intended for the teapot. In the first training round (d), the same microwave scene is solved with direct EEF control when the demonstration is available and with $\pi_{0.5}$ when it is not, which illustrates that the demonstration determines the executor strategy before any playbook is learned. Figure 14 shows corresponding episodes of Claude Opus 5.5.

Figure 16 shows two test scenes of the experiments in Section 6, on which the agent without $\pi_{0.5}$ and the final self-improved version differ in opposite directions.

F.6 Case analyses

Two cases from the experiments of Section 6 are reconstructed call by call and read as in Appendix D: a test scene evaluated with the starting point and with the final version, and two training episodes together with the critic's reviews of them.

Self-improvement adds a finishing stage that the agent executes itself (dishwasher, seed 002). Table 17 lists the calls behind Figure 5b. Both versions delegate the loading to $\pi_{0.5}$, which executes 85% of the physics steps of the starting point and 71% of those of the final version, and both verify each bowl grasp with a 1 cm EEF lift, so the difference between the two episodes arises after both bowls are in the rack. The starting point delegates the remaining stage without naming it and ends its session with the door open, after which the robot is held for the remaining 1290 steps and the episode receives a Score of 0.5. The final version names the closing stage in its reason for $\pi_{0.5}$, registers that $\pi_{0.5}$ returns toward the countertop instead, retrieves the last two keyframes of the demonstration and

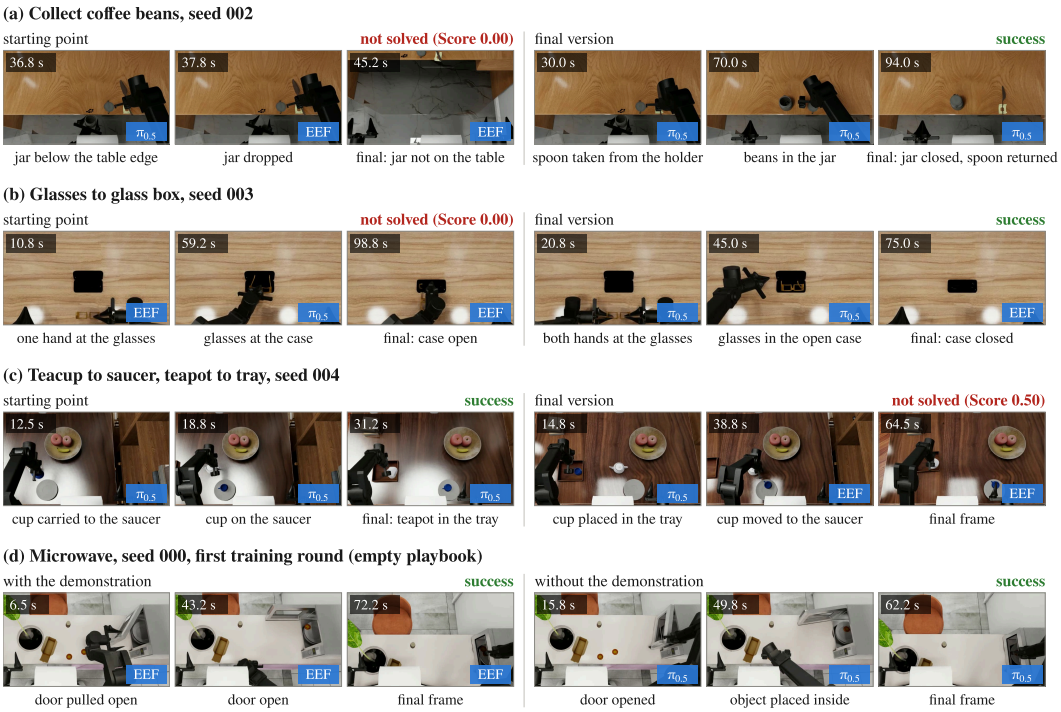


Figure 15: Keyframes from four further scenes, with the video time and the executor of each frame. (a–c) Starting point and final self-improved version on the same test scene. (d) First training round with and without the demonstration.

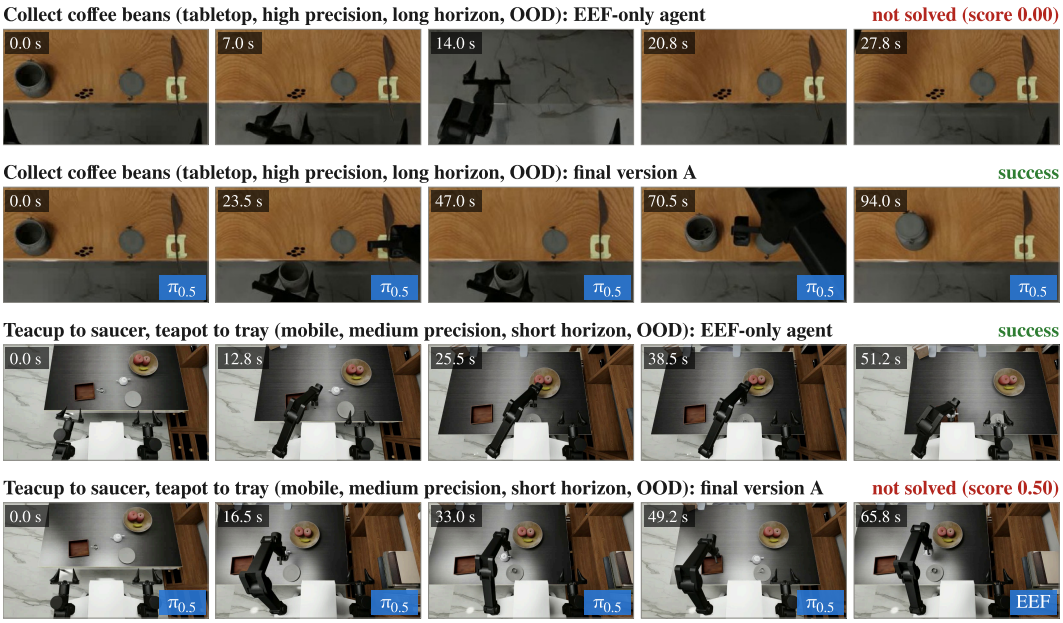


Figure 16: EEF-only agent and final version on the same test scenes (seed 2); frame labels give the executor of each action chunk. Coffee beans: the EEF-only agent's view turns away from the table (Score 0), whereas the final version delegates all 378 chunks to $\pi_{0.5}$ and succeeds. Teacup: the EEF-only agent succeeds, whereas the final version delegates 188 of 265 chunks and reaches a Score of 0.5.

closes the door in eight EEF calls over 573 physics steps; success registers at step 2197. Neither the instruction ("Put all of the bowls into the dishwasher.") nor the demonstration, whose annotation states "Closing or starting a wash cycle is not established by these views.", asks for the door to be closed. The stage comes from a playbook rule accepted after round 1, in which both training runs on this task succeeded after $\pi_{0.5}$ had closed the door, and whose text includes "Do not omit a supported finishing stage merely because a reference narration ends at loading." On the five test seeds of this task, success rises from 2 to 4, while the EEF-only agent attains a Score of 0.5 on all five. The gain is a change of strategy rather than of precision: the largest EEF miss of the final version is 1.0 cm, and the added stage is executed by the agent's own control, not by the policy to which it delegates. Delegation remains a liability elsewhere: on the teacup scene of Figure 16, the final version delegates 188 of 265 action chunks and attains a Score of 0.5, where the EEF-only agent succeeds.

The critic's reviews are correct in both directions, but neither yields an effective revision. Table 18 follows two unsuccessful training episodes without the demonstration from the actor's calls to the critic's review and proposals; both appear in Table 16. In the dishwasher episode of round 3, the review's stated miss of "about 0.435 m" at call 11 equals the measured 43.5 cm, and the critic's observations for the round report the 22.1 cm miss of the paired run with the demonstration with the same accuracy. The commanded target of call 11 lies 31.4 cm below the hand's starting height with the fingers horizontal; the hand moves 4.5 cm, the tool reports success, and the actor states in its next reason that the pose "was not reached", later that the "arm endpoint falls short". The review nevertheless attributes the failure to "delayed intervention and costly door recovery", the two proposals that cite the episode, to remove the closing rule and to prefer long delegation, are both rejected, and none of the six proposals of the round mentions reach, joint limits or inverse kinematics; the playbook in force during that round instead states that residuals "do not identify joint limits, obstruction, contact force, object rotation, or task success". In the peg-insertion episode of round 0, the review rules out an executor error, and the measurement agrees: no EEF waypoint misses by more than 0.07 mm, and the actor's own reasons name the "peg base" and "its block" as grasp targets. The rule derived from this review was accepted, entered the playbook of round 1 and was replaced by refined versions after rounds 1 and 4, yet no peg-insertion episode succeeds in any later training round or in the ten test episodes of the starting point and the final version. The two cases illustrate the pattern of Section 6.2: the per-episode attribution is correct in both directions, but a correct motion diagnosis yields no revision of the harness, and a correct judgment diagnosis yields a rule that does not change the outcome.

Table 17: Dishwasher, seed 002, RSI test scene with the demonstration: starting point (top, Score 0.5) and final version (bottom, Score 1.0).

Call	Time (s)	Step	Executor	Stated reason (verbatim)	What followed
Starting point					
1–5	0.0	0	$\pi_{0.5}$, gripper, EEF	"Test dark bowl rim grasp with a slow one-centimetre vertical lift and settle." (call 4)	The first bowl is lifted under EEF control (calls 4–5).
6–8	9.5	284	$\pi_{0.5}$	"With dark bowl lifted and supported, reposition and operate dishwasher front using right arm."	Call 9: "Approach remaining white patterned bowl on countertop [. . .]"
9–15	31.5	944	$\pi_{0.5}$, gripper, EEF	"Transport the securely lifted second bowl to the open dishwasher and load free rack space." (call 15)	The second bowl is carried to the rack (44.25 s).
16	49.5	1482	EEF	"Hold withdrawn arms and let both placed bowls settle in the dishwasher rack for verification."	The second bowl is in the rack (50.5 s).
17	51.0	1530	$\pi_{0.5}$	"Both bowls appear released in rack; complete remaining dishwasher manipulation and withdraw."	The door is open in the last frame (56.75 s).
end	–	1710	–	Final message: "Both bowls are in the open dishwasher rack, and the countertop is clear."	No Score yet; the robot is held for 1290 further steps; Score 0.5.
Final version					
1–5	0.0	0	$\pi_{0.5}$, EEF	"Door has begun opening; continue opening and first-bowl placement while checking retention." (call 5)	The first bowl is loaded; the EEF lift of call 2 tests retention.
6–10	28.0	842	$\pi_{0.5}$, EEF	"Patterned bowl retained at rack entrance; finish lowering and releasing into rack then withdraw for placement inspection." (call 10)	Call 11 reports both bowls inside the rack.
11	48.0	1444	$\pi_{0.5}$	"Both bowls visibly supported inside rack with hands withdrawn; proceed with supported dishwasher closing stage."	$\pi_{0.5}$ moves back toward the countertop; the door stays open.
12–14	54.0	1624	EEF, observe	"Policy returned to cleared source; reposition with hands clear to inspect open door for manual closure." (call 12)	Call 13 retrieves the last two keyframes of the demonstration.
15–19	57.5	1736	EEF, base	"Move closed hand beneath the door's front-right edge, then gently lift to initiate closure." (call 15)	The arm raises the door (63.5 s).
20	70.5	2152	EEF	"Door is nearly shut; gently complete inward push from measured hand height then withdraw clear."	Score 1.0 at step 2197; the door is closed (71.5 s).

Table 18: Two unsuccessful training episodes without the demonstration and the critic's response. Top: dishwasher, round 3, seed 003 (Score 0.83). Bottom: peg insertion, round 0, seed 000 (Score 0). Critic rows quote the review and the proposals that cite the episode.

Call	Time (s)	Step	Executor	Stated reason or critic output (verbatim)	What followed
Dishwasher, round 3					
1–9	0.0	0	$\pi_{0.5}$	"Both bowls appear released inside rack, hands clear; monitor supported dishwasher finishing stage." (call 7)	1980 steps delegated; call 10: "Policy revisited empty source; [. . .]"
11	68.0	2044	EEF	"Approach outer dishwasher door edge from front, clear of bowls in rack."	The hand moves 4.5 cm; miss 43.5 cm; the tool reports success.
12	69.75	2100	EEF	"Previous low horizontal pose was not reached; use downward-oriented approach to outer door lip."	Miss below 0.1 mm.
13–25	71.5	2156	EEF	"Contact at the handle aperture slipped; place support farther beneath the solid door panel." (call 24)	Misses of 9.9, 18.4 and 8.2 cm in calls 18, 24 and 25.
26	97.0	2972	EEF, base	"Door is rising but arm endpoint falls short; advance base to maintain reachable support along closing arc."	The episode ends at the 3000-step limit; Score 0.83.
review	–	–	critic	"Likely delayed intervention and costly door recovery left an unresolved terminal configuration. [. . .] missed by about 0.435 m. [. . .]"	Names a motion error; the measured miss of call 11 is 43.5 cm.
proposals	–	–	critic	"Remove the supported enclosure-finishing rule because both loading episodes spent substantial budget attempting closure and still ended partial."; "Prefer long uninterrupted learned-policy segments across manipulation stages [. . .]"	Both rejected; no proposal of the round addresses reach.
Peg insertion, round 0					
6	17.0	512	$\pi_{0.5}$	"Finish centering and closing the right gripper on the peg base."	Call 9: "Both grasps need correction: left hand is visibly empty and peg remains near table; [. . .]"
20	62.25	1880	EEF	"Peg slipped during test; reopen and lower 2 cm to grasp deeper on its block rather than near the peg."	Regrasp with the right hand (calls 21–26).
23	65.75	1988	EEF	"Center fingers around peg block and deepen contact after detecting asymmetric pinch."	The closure of call 25 ends at an opening of 16.4 mm, unchanged through the 1 cm lift of call 26.
27–28	70.25	2124	$\pi_{0.5}$	"Both components now lift with their fingers and stable gaps; perform coordinated alignment for insertion." (call 27)	Score 0 at the 2500-step limit; the largest EEF miss of the episode is 0.07 mm.
review	–	–	critic	"Likely component-role confusion compounded by unstable acquisition. [. . .] No executor error appears."	Names a judgment error, in agreement with the measurement.
proposal	–	–	critic	"Before assembly or transfer, distinguish the movable target, its original support, the destination, and nearby fixed parts using the instruction and live views. [. . .]"	Accepted; replaced by refined versions after rounds 1 and 4; no later peg-insertion success.

G COMPARISON WITH PRIOR SYSTEMS

Table 19 relates the failure modes observed in this work to twenty prior self-improvement and harness systems; each entry is based on an explicit statement in the cited paper. No prior system reports executor selection biased by the training distribution of the delegated policy. Kinematic diagnostics capable of separating unreachable poses from judgment errors are built into KnowBody, ASPIRE, RHO and Agent as Policy, but none of these systems ablates them. Non-transfer of training gains is reported only for non-embodied harness evolution and for a loop that updates VLA weights, and, in a paired comparison with its own starting version, for no embodied agent harness in this set. Regression under self-evolution is additionally reported, in restricted form, for the RACaP baseline and for AdaHVLA.

Table 19: Failure modes observed in this work and their status in twenty prior self-improvement and harness systems. Columns: training gains that do not transfer; guidance that evolves while the executable interface does not; symptoms without identified causes; executor selection biased by the training distribution of the policy; failures beyond revised guidance; regression. ●: reported; ◐: reported in restricted form; ○: addressed by design; –: neither reported nor addressed; n/a: not applicable.

System	Non-transfer	Static interface	Undiagnosed cause	Executor bias	Beyond guidance	Regression
Embodied agents and harnesses						
Harness VLA / RPent (Zhang et al., 2026e; RLinf, 2026)	–	–	–	–	–	–
ENPIRE (Xiao et al., 2026)	–	○	–	–	–	–
ASPIRE (Lu et al., 2026)	○	–	○	n/a	–	–
RHO (Elmaaroufi et al., 2026)	○	○	○	n/a	●	–
Show-Harness (Chen et al., 2026)	n/a	n/a	○	n/a	–	n/a
Agent as Policy (Jia et al., 2026)	–	○	○	n/a	○	–
SHAPER (Wang et al., 2026b)	○	○	–	–	–	–
Zetta (Ding et al., 2026)	○	○	◐	–	○	○
RACaP (Li et al., 2026b)	○	○	◐	n/a	○	◐
AdaHVLA (Tang et al., 2026)	○	○	–	n/a	–	◐
RegenHarness (Wang et al., 2026a)	○	–	–	–	–	○
KnowBody (Lou et al., 2026)	–	–	○	n/a	–	○
Embodied-AAS (Zhou et al., 2026)	–	○	●	n/a	–	○
HarnessPAI (Wang et al., 2026c)	–	○	◐	○	●	–
BATON (Xu et al., 2026)	–	○	–	–	–	–
Zhang et al. (2026b), robotics	○	○	–	n/a	n/a	–
Policy-weight improvement						
AgenticRobotics (Yu, 2026)	●	–	–	–	–	●
Non-embodied harness evolution						
Yang et al. (2026)	–	●	●	n/a	●	○
Wang et al. (2026e)	●	–	–	n/a	–	●
SEAGym (Zheng et al., 2026)	●	–	–	n/a	–	●
This work	●	●	●	◐	●	◐