

starRoboHarness: Reliable Evaluation and Efficiency Analysis for Policy-Reasoner Robot Control

starRoboHarness contributors

Code: github.com/starVLA/StarHarness | [Project page](#)

Abstract Multimodal reasoners can supervise robot policies, but execution cadence and inference cost can change the measured outcome. We present starRoboHarness, which connects policy proposals and bounded reasoner decisions to verified native outcomes. The main panel covers six configurations on 50 saved RoboDojo scenes across ten tasks. QwenPI_v3 with GPT-6 Astra completes 17 cases versus nine for the pure policy. The $\pi_{0.5}$ hybrid completes 20 cases versus 18 for pure $\pi_{0.5}$ with five-step replanning, with six hybrid-only and four policy-only successes. Thirteen hybrid successes contain no corrections, while 29 corrected failures consume 78% of its reported tokens. In three matched SimplerEnv QwenPI pairs, Astra made 22 valid reviews, changed zero actions, and preserved the control outcome and action hashes in every pair. A three-task direct Astra pilot reached native truncation without success in all cases. A corrected BEHAVIOR direct Astra radio pilot also failed at native step 3,225 with Q-score 0. Four completed matched RoboDojo pilots all failed to complete: `organize_table` scored 0.25 versus 0 after 140 Astra intervention steps, `build_tower` tied at 0.1 after 270 intervention steps, and `classify_objects` scored 0.4 versus 0 in two independent pairs after 250 and 205 intervention steps. Two BEHAVIOR native/pass-through pairs likewise retained identical failure outcomes and step counts. Across seven completed policy-supervision pairs, Astra produced no completion gain. The RoboDojo partial-score effect was mixed across three scenes, while the `classify_objects` increase repeated under the same frozen scene and protocol. Two completed BEHAVIOR VLAAct diagnostics scored 0/2 on public radio instances 301 and 302, each after 3,225 native steps. The separately collected RoboDojo panel remains descriptive and cannot identify an independent causal effect of Astra.

1. Introduction

Vision-language-action policies propose action sequences from images, language, and robot state. A multimodal reasoner can assess task progress and decide whether to continue a policy proposal or intervene. The resulting system is difficult to evaluate: a change in action cadence, reset identity, or retry policy may alter the outcome independently of the reasoner’s judgment. Provider and infrastructure failures further confound success rates if they are treated as native robot failures or silently replaced.

We develop starRoboHarness to make the boundary between proposal, decision, and physical execution clear. The harness binds each proposal to an observation, checks decision freshness and action limits, assigns execution to one controller, and records acknowledgements and native terminal outcomes. Policy, reasoner, and environment implementations live behind adapters. QwenPI_v3 and $\pi_{0.5}$ policies, GPT-6 Astra, and RoboDojo form the reference integrations studied here. Model weights and simulator assets remain external to the public package.

We evaluate six configurations on 50 saved RoboDojo scenes across ten tasks: QwenPI_v3 with and without Astra, three pure $\pi_{0.5}$ action cadences, and $\pi_{0.5}$ with Astra. QwenPI with Astra completes 17 scenes versus nine for the pure policy. The $\pi_{0.5}$ hybrid completes 20 versus 18 for fixed-5 pure policy,

with six hybrid-only and four policy-only successes. These scene crossovers show why an aggregate count needs a task-level account of completion and failure.

We also use BEHAVIOR and SimplerEnv as external validation tracks. These runs test whether the same proposal, validation, and receipt machinery survives different action spaces and evaluators. They are reported separately from the RoboDojo panel because the checkpoints, reset protocols, and task coverage are different.

The traces expose two recurring evaluation problems. Native partial credit can increase while a task remains incomplete. Prolonged recovery can consume most of the reasoner’s inference without producing a completion. In the $\pi_{0.5}$ hybrid, 29 corrected failures account for 78% of reported tokens, while thirteen successful episodes contain no corrections. Understanding this allocation requires both the executed action trace and the native terminal outcome.

Our contributions are a proposal-to-outcome control contract, validity-aware evidence for 300 native terminal observations across six configurations, and scene-level analyses of completion, recovery, and inference cost. Sanitized case rows and digest-checked table generation support the analysis. Each configuration has one separately collected rollout per scene. The study therefore establishes the observed outcomes and resource allocation, while a supervision effect requires execution cadence and collection conditions to be held fixed. Code and the project overview are linked on the title page.

2. Related Work

Robot policies and action cadence. RT-1 and RT-2 established language-conditioned robot control at scale, while Open X-Embodiment, Octo, and OpenVLA developed cross-embodiment datasets and generalist policy interfaces [1, 2, 3, 4, 5]. Flow and diffusion policies, including π_0 , $\pi_{0.5}$, Diffusion Policy, and RDT-1B, use different approaches to continuous action generation [6, 7, 8, 9]. VIMA studies multimodal prompts for manipulation, and BAKU studies efficient multi-task policy learning [10, 11]. Action Chunking Transformers and later real-time chunking work show why the executed prefix and inference cadence belong in the method definition [12, 13]. BridgeData V2 and DROID broaden the training-data base, while LeRobot provides a reusable policy interface [14, 15, 16]. StarVLA supplies the modular policy backbone and the QwenPI_v3 implementation used in our integration, while VLAct studies continued pretraining for vision-language-action policies [17, 18].

Reasoning, supervision, and recovery. SayCan grounds language plans in skill affordances, while Inner Monologue uses environment feedback to revise plans [19, 20]. Code as Policies, PaLM-E, Vox-Poser, and ManipLLM connect language or multimodal reasoning to executable robot behavior at different levels of abstraction [21, 22, 23, 24]. Closed-loop reasoning, VLM predictive control, reflective planning, and reasoning-to-decision systems make feedback and replanning explicit [25, 26, 27, 28]. InstructVLA integrates instruction understanding with action generation rather than placing a separate reasoner above a policy [29]. AHA studies vision-language failure detection. RACER and LIBERO-RECOVER address recovery after errors [30, 31, 32]. Runtime monitoring with interactive imitation learning provides another intervention model [33]. GPT-as-Policy compares direct reasoning control with specialist-policy supervision, and Agent as Policy studies a general-purpose agent in direct manipulation control [34, 35]. Our system focuses on the auditable boundary between a policy proposal, a reasoner decision, and the native outcome. Results from these different controllers cannot be transferred across protocols.

Benchmarks and evaluation validity. RLBench, LIBERO, CALVIN, ManiSkill2, RoboCasa, and RoboMimic provide complementary manipulation suites and datasets [36, 37, 38, 39, 40, 41]. The

Colosseum, VLABench, and RoboTwin 2.0 extend evaluation toward generalization, long-horizon reasoning, and bimanual robustness [42, 43, 44]. RoboDojo spans simulation and real-world tasks, and our ten subtasks with five cases each form a selected subset of its evaluation space [45]. SimplerEnv evaluates real-robot policies in visually matched simulation, while BEHAVIOR specifies task-trained policy baselines and a public evaluation protocol [54, 55, 56]. Benchmark audits and LIBERO-PRO show how task selection and shortcuts affect the meaning of a reported score [46, 47]. UMI-Bench and RB2 emphasize repeatable real-world evaluation and controlled benchmarking [48, 49]. Recommendations for empirical policy evaluation, imperfect simulators, and shared evaluation interfaces motivate our receipt and validity rules [50, 51, 52]. Metamorphic testing provides a related approach to systematic input checks [53]. We retain exact case identity, native receipts, validity rules, and a separate account of excluded attempts. These controls support auditability. The cohorts remain non-contemporaneous and are not a randomized trial.

These studies imply three constraints for a policy–reasoner evaluation. Policy outputs have a temporal granularity, so approving a proposal changes when the system observes and replans. Reasoner interventions consume time and tokens, so quality and cost must be measured on the same cases. A benchmark outcome is meaningful only after the evaluator distinguishes a native timeout from a provider failure. Our case-level record ties these boundaries together and leaves the remaining confounds visible.

3. System Architecture

Figure 1 summarizes the system architecture from task and world state through method selection, execution, adapters, and audit. The task specification, visual observations, robot state, and constraints are distinct inputs. The evaluated method profiles are policy-only execution, a reasoner supervising policy proposals, and direct reasoner control. All three pass through the shared execution and evidence boundary before reaching an environment adapter. Monitoring and native evaluation read the resulting trace.

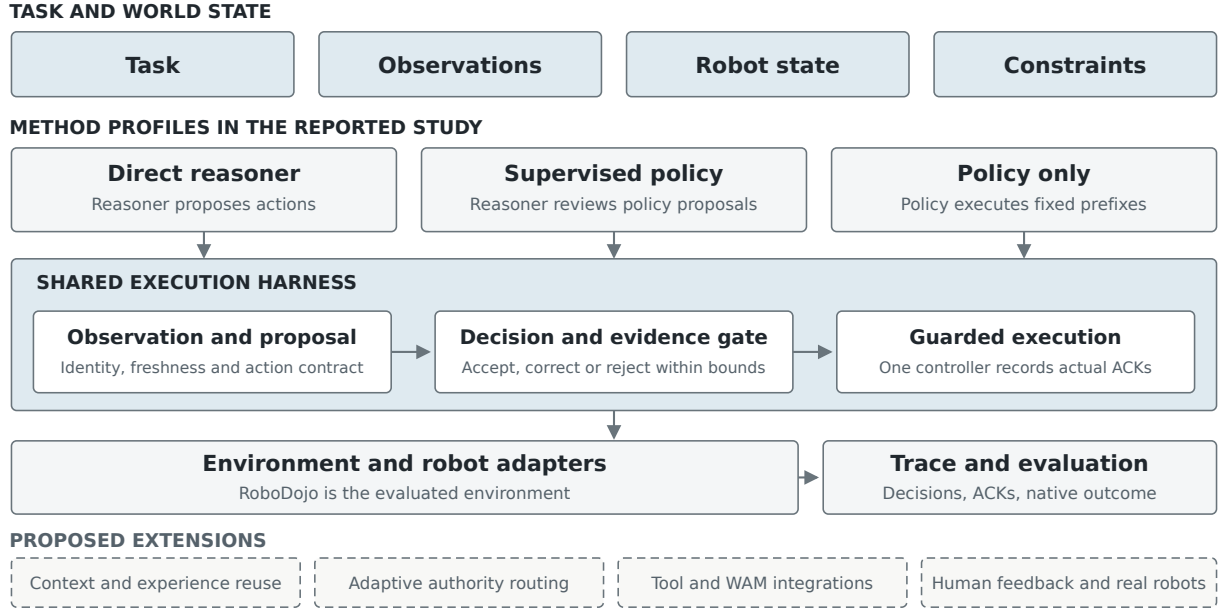


Figure 1 | Framework used in the reported study. The task specification and current observations enter the shared harness. Policy-only execution, reasoner supervision, and direct control use the same observation, proposal, decision, and execution boundary. Environment adapters record acknowledgements and native outcomes. Proposed extensions are shown separately and do not contribute to the reported scores.

The figure separates the evaluated paths from extensions that are outside the current results. QwenPI_v3 and $\pi_{0.5}$ connect to the shared contract through benchmark adapters. Section 5 reports the RoboDojo panel and the completed external validation runs.

3.1. Execution contract

Each proposal is bound to an exact observation and carries policy and action-contract identity. The adapter checks shape, freshness, bounds, units, gripper semantics, and request identity. A reasoner reviews the previous execution and next proposal using visible images, proprioception, and public trace evidence. Hidden reward, privileged object state, and future simulator truth are outside that interface. One controller owns state-changing execution and records acknowledgements. The native RoboDojo terminal outcome determines success and score. An attempt with uncertain acknowledgement remains in the ledger and is not silently replayed under the same identity.

Both reference policies use three RGB views and dual-arm proprioception to propose 50-action sequences. QwenPI_v3 uses a 50-by-14 absolute-joint proposal with the pinned `arx_x5` normalization. The RoboDojo-fine-tuned $\pi_{0.5}$ checkpoint uses its released OpenPI configuration [7]. Policy-only execution takes a specified prefix before requesting a fresh prediction. In the hybrid integrations, Astra chooses a prefix or issues a local correction within the adapter’s action bounds.

4. Experimental Design

4.1. Configurations and saved scenes

The evaluation covers six configurations: QwenPI_v3 with and without Astra, pure $\pi_{0.5}$ with 50-, 15-, and 5-action execution prefixes, and $\pi_{0.5}$ with Astra. Both hybrid integrations use GPT-6 Astra and bounded corrections of at most five steps. Their accepted policy prefixes are adaptive, up to 15 steps. Pure QwenPI executes 16 steps per prediction. Pure $\pi_{0.5}$ uses the native RoboDojo client with one simulator environment. These method identities distinguish supervision from execution cadence. The separately collected runs do not isolate either factor’s causal effect, so the six-configuration panel cannot be read as an Astra-only comparison.

Each configuration has a native terminal outcome on the same 50 saved scenes: ten tasks with five cases per task. Matching checks task, variant, reset layout, seed, case identifier, and scene-file digest. All 50 $\pi_{0.5}$ layout files match the QwenPI scene manifest. Packing includes three standard and two random layouts. Task groups A and B each contain five tasks and retain the provenance of the QwenPI data (Tables 3 and 4). Group labels describe data provenance rather than distinct benchmark difficulty levels. Supplementary development and excluded runs are excluded from the 50-scene denominators.

4.2. Outcome definitions and evidence

The unit of analysis is a native terminal episode. A row enters success-rate and score denominators only when `complete=true`, `valid_for_success_rate=true`, and a simulator terminal receipt exists. Timeouts at the native step limit are valid failures with their native partial score. Table generation checks scene coverage, outcome identity, and source digests before aggregating results. Appendix G describes the receipt and reset rules.

For a method m and a task containing $N = 5$ valid cases, success rate is $N^{-1} \sum_i s_{i,m}$, where $s_{i,m}$ is the native binary outcome. Mean score is $N^{-1} \sum_i r_{i,m}$, where $r_{i,m}$ is the native score on a 0–100 scale. Scene-level comparisons count successes unique to each configuration and score increases, decreases, and ties. Resource measures include native steps, policy requests, correction steps, elapsed controller time, and reported model tokens. Pure $\pi_{0.5}$ request counts are reconstructed from native steps and executed prefix lengths. Token snapshots include cached input and are not billing records.

4.3. Scope of comparison

Each configuration has one separately collected rollout per scene. The three pure $\pi_{0.5}$ cadences share a checkpoint and native client. The hybrid adds a reasoner and adaptive execution prefixes. QwenPI and $\pi_{0.5}$ also differ in policy interface and serving stack. Hardware, provider sessions, collection period, and recovery history vary across configurations. We report paired scene outcomes and resource use as descriptive observations. Exact scene matching permits trajectory comparison. It does not create a randomized supervision trial or a controlled latency study.

5. Results

5.1. Overview of the six configurations

Table 1 summarizes 300 native terminal observations on the 50 saved scenes. QwenPI with Astra completes 17 scenes versus nine for pure QwenPI. Pure $\pi_{0.5}$ completes 10, 12, and 18 at the three execution cadences. Its Astra hybrid completes 20. The means and success counts describe the recorded configurations under the comparison conditions in Section 4. They are not isolated effects of either the policy backbone or supervision.

Configuration	Policy prefix	Success	Mean score
QwenPI_v3	16	9/50	23.7
QwenPI_v3 + Astra	Adaptive	17/50	47.0
$\pi_{0.5}$, original	50	10/50	31.8
$\pi_{0.5}$, fixed-15	15	12/50	32.0
$\pi_{0.5}$, fixed-5	5	18/50	41.4
$\pi_{0.5}$ + Astra	Adaptive	20/50	55.3

Table 1 | Six configurations on the same 50 saved scenes. Policy prefix is the number of proposed actions executed before replanning. Adaptive hybrid prefixes are at most 15 steps, with corrections bounded to five steps. Success counts native completions, and mean score uses the native 0–100 scale. Each configuration has one separately collected rollout per scene.

5.2. Cross-benchmark validation

The cluster runs provide a portability check for the execution contract. They are not pooled with the RoboDojo denominators because the task sets, action spaces, checkpoints, and reset protocols differ. Table 2 records only runs with a native terminal receipt and a corresponding audit artifact.

Benchmark	Policy-only evidence	Supervised or direct path	Evidence-backed reading
RoboDojo	QwenPI 9/50, pure $\pi_{0.5}$ fixed-5 18/50	QwenPI + Astra 17/50, $\pi_{0.5}$ + Astra 20/50	Complete descriptive panel. The separately collected arms do not identify an Astra effect.
BEHAVIOR VLAct Qwen-Dual	0/2, Q-score 0.0 on instances 301 and 302	n/a	Native R1Pro diagnostics completed 3,225 steps each. This is a two-instance contract check, not a full BEHAVIOR estimate.
BEHAVIOR radio	Task-trained $\pi_{0.5}$ 2/10, GR00T N1.7 0/10 on public instances	$\pi_{0.5}$ + Astra: 0/2 audited trials, 16 valid reviews, 14 replans. Direct Astra: 0/1, 13 calls, normalized 23D	These are completed single-instance or two-instance diagnostics. They are not causal estimates.
SimplerEnv Google Robot	QwenPI 10/15, RT-1-X 12/15, official pinned GR00T 10/15	QwenPI + Astra: 2/3 seed-0 trials, 22 valid reviews, 0 edits. Direct Astra: 0/3, 40 calls	The supervised pilot preserves native actions. Direct Astra reaches native truncation on all three tasks.

Table 2 | Evidence coverage across benchmarks. Counts are native successes over completed episodes. External validation runs use the 4090 cluster. GR00T uses its official pinned SimplerEnv checkout and a different episode horizon from the QwenPI and RT-1-X runs, so these counts do not establish a policy ranking. BEHAVIOR supervised trials review policy proposals. The BEHAVIOR direct pilot uses the normalized R1Pro action space and one public instance. SimplerEnv also includes a three-task direct Astra pilot. The runs use different action interfaces and are not a policy ranking.

The external tracks use different action spaces and checkpoints. BEHAVIOR provides task-trained radio checkpoints and a public evaluation protocol [55, 56]. SimplerEnv-specific NVIDIA GR00T checkpoints are available [57]. The completed BEHAVIOR native run confirms that the R1Pro adapter and receipt path operate on a public instance. Its single-instance scope and the differing action contracts keep it separate from the RoboDojo panel.

The matched cross-benchmark evidence is operational. On SimplerEnv, the harness pass-through preserves every seed-0 QwenPI action hash, and the Astra pilot preserves the same hashes while completing 22 valid reviews. All three tasks retain the same native outcome, so the review path adds cost without changing control in these cases. On BEHAVIOR, two fresh native/pass-through pairs each validate 3,225 actions and retain failure with Q-score 0. These checks show that the contract and receipt path transfer across evaluators. They do not establish a population-wide absence of benefit. The BEHAVIOR pairs contain no reasoner intervention. The direct pilots use one BEHAVIOR instance and three SimplerEnv tasks.

5.3. Task group A: five paired subtasks

Task	Policy succ.	Hybrid succ.	Policy score	Hybrid score
Organize table	0/5	0/5	10.0	65.0
Classify by language	0/5	0/5	0.0	0.0
Imitate sorting	0/5	1/5	1.0	36.0
Arrange largest number	0/5	1/5	2.0	37.0
Fold clothes	1/5	4/5	32.0	80.0
Overall (25)	1/25	6/25	9.0	43.6

Table 3 | Task group A. Each task has five exact case pairs. Success is shown as completed cases over valid cases. Score is the mean native score on a 0–100 scale.

Hybrid success is 6/25 (24%) versus 1/25 (4%) for the pure policy. Native mean score is 43.6 versus 9.0 (Table 3). Across exact binary pairs there are five hybrid-only wins, no policy-only wins, and 20 ties. Native score improves in 16 pairs, declines in one, and ties in eight. Fold clothes contributes three of the five additional successes. Its success count rises from one to four and its mean score rises from 32 to 80. Language classification is unsuccessful for both methods and has zero mean score under both methods. The aggregate gain combines a task-specific improvement with a task showing no measurable progress. Different student execution cadences and stochastic visual observations prevent attributing the paired difference solely to corrections.

5.4. Task group B: five paired subtasks

Task	Policy succ.	Hybrid succ.	Policy score	Hybrid score
Put bottles in bin	3/5	4/5	64.0	82.0
Classify objects	0/5	0/5	6.0	22.0
Build tower	4/5	4/5	82.0	86.0
Make Kong	1/5	3/5	20.0	60.0
Pack objects in box	0/5	0/5	20.0	2.0
Overall (25)	8/25	11/25	38.4	50.4

Table 4 | Task group B. Each task has five exact case pairs. Pure-policy counterparts were collected after the fixed hybrid cases. Success is shown as completed cases over valid cases.

Group B gives 11/25 (44%) hybrid and 8/25 (32%) pure-policy successes, with mean native scores 50.4 and 38.4 (Table 4). The 25 binary pairs contain four hybrid-only wins, one policy-only win, and 20 ties. Score changes are more mixed: seven positive, three negative, and 15 tied pairs. Make

Kong accounts for two extra successes, while bottles accounts for one. Build tower has 4/5 successes under both methods. The gain is concentrated in a few tasks.

Packing is the largest regression. Both methods fail all five cases, but mean native partial score falls from 20 to 2. The hybrid cases reach the native 1,300-step limit with scores 0, 0, 0, 0, and 10, so all five are valid scored failures. In one trace, a verified toy-car grasp is followed by repeated five-step transfer corrections, four accepted 15-step policy segments, and a later takeover after an open-gripper hover. The trace includes a verified grasp, so grasping alone does not explain that failure. It motivates analysis of post-grasp transfer and intervention timing, without explaining the other four failures. Classification likewise remains at 0/5 success for both methods despite a higher hybrid partial score.

Figure 2 separates completion from partial progress. Eight tasks have a higher hybrid mean score, one is tied, and packing is lower. In group B classification, score rises from 6 to 22 while success remains 0/5 under both methods. Build tower retains 4/5 successes with a four-point score increase. A larger score can indicate progress toward a goal without a completed task.

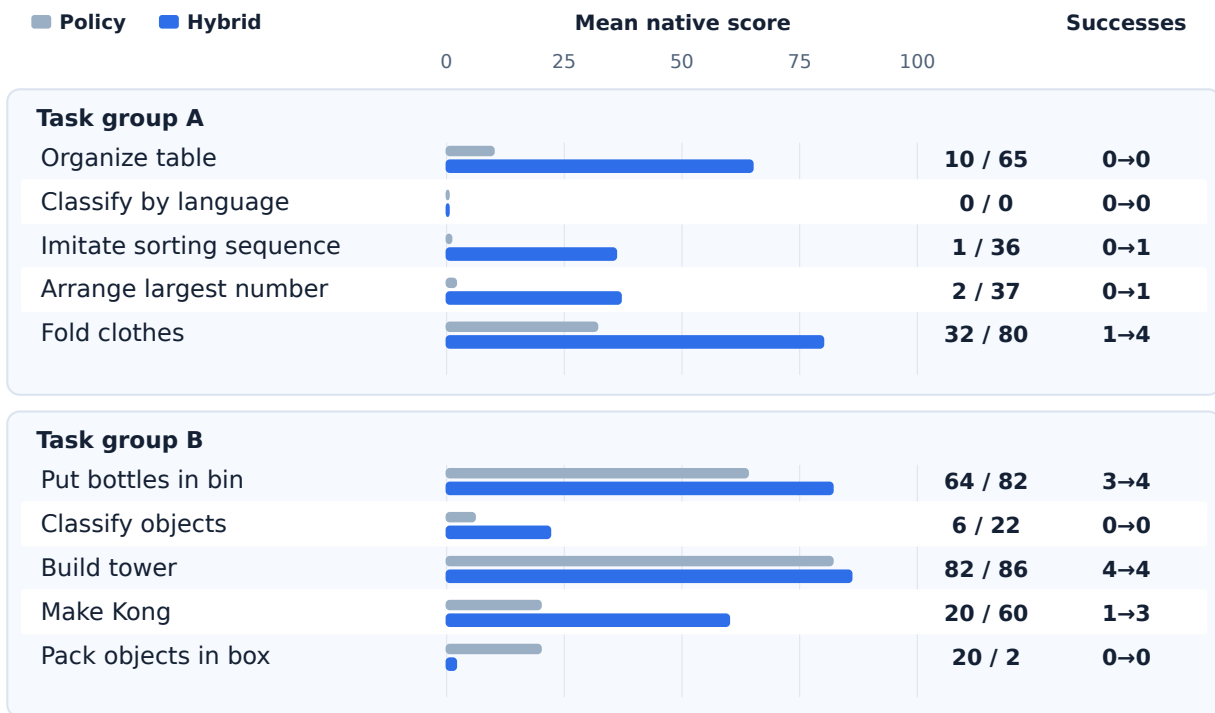


Figure 2 | Task-level mean native score and success count on five matched cases per task. Gray bars show pure policy, blue bars show hybrid, and the rust-colored bar marks the packing regression. Numbers at right give policy/hybrid mean score and policy-to-hybrid success counts. Groups A and B were collected in different periods.

Ten-task descriptive index. Across both groups, the hybrid completes 17/50 cases (34%) and has mean native score 47.0. The corresponding policy cases complete 9/50. Because group B policy cases were collected later than its hybrid cases, these pooled counts describe the observed task set. Group-level paired estimates remain the basis for interpretation.

5.5. Policy choice and replanning cadence

At original cadence, pure $\pi_{0.5}$ completes 10 scenes versus nine for QwenPI. The one-case difference hides six scenes completed only by $\pi_{0.5}$ and five only by QwenPI, with four shared successes. Their union covers 15 scenes. This is retrospective complementarity, not the performance of a runnable selector. The task-level comparison is given in Table 14.

For pure $\pi_{0.5}$, shortening the executed prefix from 50 to 15 to five steps gives 10, 12, and 18 completions, respectively (Table 5). Both shorter cadences complete three folding scenes and all five bottle-placement scenes. Fixed-5 additionally completes two packing scenes and four build-tower scenes, but neither short cadence completes a Make Kong scene. Language classification and sorting have no completions at any of the three pure-policy cadences. The aggregate trend conceals these task differences.

Task	Orig. succ.	15 succ.	5 succ.	Orig. score	15 score	5 score
Organize table	0/5	1/5	1/5	35	40	35
Classify by language	0/5	0/5	0/5	0	0	2
Imitate sorting	0/5	0/5	0/5	0	0	0
Arrange largest number	0/5	0/5	1/5	3	4	26
Fold clothes	0/5	3/5	3/5	16	64	64
Put bottles in bin	3/5	5/5	5/5	76	100	100
Classify objects	1/5	1/5	2/5	26	36	51
Build tower	3/5	2/5	4/5	68	46	82
Make Kong	3/5	0/5	0/5	60	0	0
Pack objects in box	0/5	0/5	2/5	34	30	54
Complete panel	10/50	12/50	18/50	31.8	32.0	41.4

Table 5 | Complete exact-scene pure $\pi_{0.5}$ cadence panel. Each row contains five saved layouts. Original, 15, and 5 denote executed actions per fresh policy prediction. Scores are mean native RoboDojo scores on a 0–100 scale. All three methods have valid native terminals on all 50 selected layouts. Each has one rollout per scene, collected separately. No method uses a reasoner.

5.6. $\pi_{0.5}$ with Astra supervision

The $\pi_{0.5}$ hybrid completes 20/50 scenes with mean score 55.3, compared with 18/50 and 41.4 for fixed-5 pure policy. Fourteen scenes succeed under both, six only under the hybrid, and four only under fixed-5. The hybrid has higher native score on 19 scenes, lower score on seven, and equal score on 24. The six unique hybrid completions comprise two language-classification scenes, two random folding scenes, one object-classification scene, and one Make Kong scene. Fixed-5 alone completes an organize-table scene, a build-tower scene, and two packing scenes.

Table 6 reports task completion and resource use together. The hybrid completes all folding and bottle-placement scenes. It completes two language-classification scenes, which none of the three pure-policy cadences completes. It also fails every packing scene, including the two completed under fixed-5. The higher aggregate score and completion count coexist with scene-level regressions. The adaptive hybrid prefixes and separate collection conditions preclude attributing the 13.9-point mean-score difference to Astra alone.

Task	Valid	Hybrid succ.	Fixed-5 succ.	Hybrid score	Corr. steps	Tokens (M)
Organize table	5/5	0/5	1/5	55.0	1,340	117.52
Classify by language	5/5	2/5	0/5	50.0	819	80.17
Imitate sorting	5/5	0/5	0/5	26.0	879	98.25
Arrange largest number	5/5	1/5	1/5	42.0	890	107.94
Fold clothes	5/5	5/5	3/5	100.0	15	22.36
Put bottles in bin	5/5	5/5	5/5	100.0	5	44.24
Classify objects	5/5	3/5	2/5	66.0	423	100.40
Build tower	5/5	3/5	4/5	62.0	680	92.85
Make Kong	5/5	1/5	0/5	20.0	730	89.89
Pack objects in box	5/5	0/5	2/5	32.0	1,825	156.42
Observed total	50/50	20/50	18/50	55.3	7,606	910.02

Table 6 | Task-level outcomes on 50 exact saved scenes with native-valid $\pi_{0.5}$ -Astra terminals. Fixed-5 success counts come from a separately collected pure-policy campaign on the same scenes. Scores use RoboDojo’s native 0–100 scale. Corrections combine edited and recovery steps. Reported tokens include cached input and are not billing.

5.7. Inference cost and efficiency

The cost of supervision includes more than the task outcome. The reasoner can add policy replans, correction actions, and model calls. Quality and resource use must therefore be examined on the same cases. Group B has exact receipts for both methods, but its wall-time and usage counters remain operational telemetry because hardware and provider routes were not matched.

Metric, group B (25 cases)	Pure policy	Hybrid
Mean control-loop wall time (s)	330.5	4,999.8
Mean policy calls / episode	54.4	85.6
Total correction steps	0	4,627
Total input tokens	0	462,535,681
Cached input fraction	n/a	93.95%
Total output tokens	0	1,563,270

Table 7 | Runtime telemetry on group B exact cases. Token counts are the last recorded asynchronous usage snapshots, not billing. Hardware, transport, and collection period differ, so wall-time values are descriptive rather than a controlled slowdown estimate.

Table 7 shows the scale of reasoning used by the group B hybrid. Input-token caching is high, yet 462.5 million reported input tokens and 4,627 correction steps were recorded across 25 cases. The roughly 15-fold difference between mean control-loop times is descriptive because machines and transports differ. Packing shows that additional reasoning and corrections need not improve a native outcome. A controlled comparison must measure quality and cost on matched cases, hardware, and provider routes.

The completed pure $\pi_{0.5}$ cadence panel supplies a within-policy quality and cost check on the same 50 saved scenes. Native client logs imply 884 policy requests over 43,770 steps at original cadence, 2,819 over 42,049 steps at fixed-15, and 8,313 over 41,525 steps at fixed-5. Completion rises from 10 to 12 to 18 cases, and mean native score is 31.8, 32.0, and 41.4. Fixed-5 uses 2.95 times as many requests as fixed-15 for six additional observed completions, and 9.40 times as many as original cadence for eight additional completions. Language classification, sorting, and Make Kong together

consume 2,665 fixed-5 requests, 32% of that method’s total, without a completion. Four other task families account for all six extra fixed-5 completions over fixed-15. Requests are reconstructed from control steps and executed prefix lengths. They do not measure GPU time, energy, or billing. These are single rollouts at each cadence.

Resource use in the 50 native-valid $\pi_{0.5}$ –Astra cases is concentrated in unsuccessful recovery. Twenty-nine failed episodes with corrections account for 7,296 of 7,606 correction steps and 713.53 of 910.02 million reported tokens. The seven corrected successes use 310 correction steps and 85.64 million tokens. Thus 96% of observed correction steps and 78% of valid-case reported tokens fall in corrected failures. Harder scenes can require both more recovery and more inference, so these descriptive groups do not establish that corrections caused failure.

The ten scenes with discordant hybrid and fixed-5 success show this pattern. Two of six hybrid-only successes use no corrections. The other four use 15, 40, 55, and 165 correction steps. In all four fixed-5-only successes, the hybrid fails after 290, 380, 450, or 470 correction steps. Correction volume therefore does not track the observed completion advantage on these scenes. The rollouts were collected separately, so task difficulty, action timing, and trajectory variation remain possible explanations.

The concentration also appears by task (Table 6). Organize table, sorting, and packing each have five native-valid hybrid cases and no success. Their fifteen cases use 372.18 million reported tokens and 4,044 correction steps, or 41% of tokens and 53% of correction steps in the complete panel. Mean native partial scores are 55, 26, and 32, respectively. These tasks show that costly partial progress can coexist with zero completion.

On all 50 scenes, the hybrid records 3,952 student-policy predictions and the same number of Astra decisions over 42,539 native steps. Fixed-5 pure policy requires 8,313 reconstructed policy requests over 41,525 native steps on those saved scenes. The hybrid makes 52% fewer policy requests, but adds a reasoner decision for every proposal and reports 910.02 million tokens. This is an operational tradeoff between policy request frequency and reasoner use. Different trajectories and control horizons prevent interpreting the request ratio as an isolated speed or compute gain.

The supplementary development cases and direct-control case are reported separately in Appendix B. Neither contributes to the paired estimates.

6. Discussion

6.1. Delegation and correction explain different parts of a trajectory

A hybrid controller can change the duration of a delegated policy segment and the action itself. In the $\pi_{0.5}$ hybrid, thirteen of twenty successful episodes contain no corrections, including four of five folding successes and four of five bottle-placement successes. Those episodes still require reasoner decisions and consume 105.74 million reported tokens. A hybrid completion is insufficient evidence of a successful recovery intervention.

The six successes absent from fixed-5 pure policy clarify this distinction. Two contain no corrections. The other four use 15, 40, 55, and 165 corrected steps. Random folding layout 0 localizes one possible recovery: three short corrections move the gripper from a book toward the garment, after which the policy completes the fold. In language classification, the supervisor redirects objects away from an incorrect basket before resuming policy execution. These traces identify actions and subsequent progress within successful runs. Appendix F supplies the native frames and case details. Because intervention is selected in response to the observed state, comparing corrected and

uncorrected episodes cannot determine a correction’s causal effect.

6.2. Execution cadence changes which scenes are completed

Pure $\pi_{0.5}$ completes 10, 12, and 18 scenes at 50-, 15-, and 5-step prefixes. The task pattern is uneven. Both shorter prefixes complete three folding scenes and all five bottle-placement scenes, while original cadence completes none and three, respectively. Make Kong shows the opposite ranking: original cadence completes three scenes, and neither shorter prefix completes any. Build tower has three, two, and four completions. These outcomes make execution cadence part of the method definition rather than an incidental serving choice.

The hybrid’s 20 completions versus fixed-5’s 18 also combine gains and losses. The hybrid completes two language-classification scenes and both random folding scenes absent from fixed-5. Fixed-5 completes two packing scenes absent from the hybrid. In build tower, a hybrid failure spends 380 correction steps on displaced supports and board placement, while original and fixed-5 pure policy each complete the same saved layout. A successful recovery on one task does not predict the task-level ranking elsewhere. These single-rollout scene crossovers should guide repeated controls and do not establish a general controller ordering.

6.3. Native partial credit differs from final task state

RoboDojo’s transition scorers preserve the last reached stage even when subsequent actions disturb object poses. The hybrid sorting task has mean score 26 with no completion in five scenes. Organize table has mean score 55 and no completion. These scores record credited progress, not a certificate of the terminal arrangement.

Packing illustrates the gap between an agent’s assessment and the benchmark’s geometric checks. In layout 1, the controller reports that all four objects are at the box, with orientations unverified. It then spends 310 of the remaining 392 steps correcting a toothbrush. Native credit never advances beyond the one-object stage. The score requires specified orientation, box alignment, containment, and open grippers. A grasp or apparent placement alone is insufficient. Organize table exposes a related mismatch: its instruction includes drawer opening, while the native completion check has no separate drawer-opening condition. One trajectory uses 105 steps attempting that subgoal. An audit must therefore keep visible motion, language-specified progress, scored stages, and native completion distinct.

6.4. Unsuccessful recovery accounts for most recorded inference

Twenty-nine corrected failures consume 78% of valid-case reported tokens and 96% of correction steps. Three tasks with no hybrid completion account for 41% of those tokens. The concentration is not explained by intervention count alone: a corrected Make Kong success uses 40 recovery steps, while a failed trajectory uses 120. In another Make Kong failure, 105 correction steps revisit an unrelated tile that remains face-up at termination. Packing contains a similarly extended sequence without additional native credit.

These observations motivate monitoring recovery progress alongside cost. A useful record links each correction to its precondition, the executed action, the next observation, and any native progress. Repeated visits to an unresolved state can then be distinguished from a sequence that advances the task. The present data identify where inference accumulated. They do not establish that reducing those corrections would improve completion. The inference ledger records resource use only for episodes with comparable receipts. The reported token totals describe the completed scored cases.

6.5. Implications for evaluating a supervisor

The audited panel supports three conclusions about evaluation. A hybrid completion is not evidence that a correction was necessary: thirteen of the twenty successful $\pi_{0.5}$ hybrid episodes contain no corrections. More reasoning is not evidence of effective recovery: fifteen hybrid episodes in organize table, sorting, and packing consume 372.18 million reported tokens without a completion. Finally, exact scene matching does not establish reproducibility of an outcome: fixed-5 pure policy completes the same packing scene in two of three runs. Each conclusion follows from a counterexample or an audited count, without assuming that the reasoner caused the observed outcome.

These observations define separate questions for a controlled evaluation. Native completion tests whether the full task was achieved. Matched execution windows test whether supervision adds value beyond the policy’s replanning cadence. Repeated paired runs quantify outcome variability, while the inference ledger measures the resources spent on both completed episodes and interrupted attempts. Reporting one of these quantities cannot substitute for the others.

6.6. What the matched audits establish

The external matched audits support a limited result. In the three seed-0 SimplifierEnv QwenPI pairs, Astra completed 22 valid reviews and selected the student action every time. The action hashes, step counts, and native outcomes matched the byte-preserving control in all three pairs. The review path therefore added inference latency without changing the executed control or task result in this sample. In the two fresh BEHAVIOR native/pass-through pairs, both arms ran 3,225 steps, received Q-score 0, and failed the task. Those pairs validate gateway preservation and do not contain a reasoner intervention.

Four completed RoboDojo matched pilots provide direct intervention evidence. On `organize_table`, both arms failed at 1,000 steps. The control scored 0.25 and Astra scored 0 after 140 eef intervention steps. On `build_tower`, both arms failed at 1,050 steps and scored 0.1. Astra used 780 student steps, 260 eef steps, and 10 edit steps. The original `classify_objects` pair ended at 1,100 steps with scores 0 and 0.4 after 235 eef steps and 15 edit steps. A fresh pair on the same frozen scene and protocol again ended at 1,100 steps with scores 0 and 0.4. This Astra arm used 160 eef and 45 edit steps. Scene identity, reset, cadence, native receipts, and immutable artifact manifests passed the audit for all four pairs. These are exact within-pair results, not an estimate of the average effect across RoboDojo.

The `classify_objects` trace shows what the partial-score increase means. Astra intervened when the student trajectory carried a watch toward the vehicle basket, redirected a pen away from the watch basket, and corrected the final vehicle’s transport toward the white basket. The controller’s final assessment records all three categories in their intended baskets, but the native evaluator still reports a timeout and score 0.4 because the full terminal condition was not reached. The audit therefore supports a concrete claim about intermediate recovery in this case. It does not support a claim that the intervention policy reliably completes the task.

The matched SimplifierEnv telemetry shows the efficiency tradeoff. The review path used 1,149.5 seconds of recorded reasoner time and 1,188.7 seconds of wall time across the three tasks, compared with 36.5 seconds for the byte-preserving control. This is a descriptive 32.55-fold wall-time ratio for this serving path. It is not a hardware-normalized latency estimate. Zero action edits still required inference cost.

The audit supports the following trace-level conclusion. In the three SimplifierEnv pairs, Astra made no action changes and did not change the native result. In the four completed RoboDojo matched

pilots, Astra did not improve completion: all seven policy-supervision pairs ended with the same binary outcome in both arms. Partial scores fell on `organize_table`, tied on `build_tower`, and rose on both `classify_objects` runs. The two `classify_objects` contrasts were 0 to 0.4 after 250 and 205 Astra intervention steps. Astra added inference in all seven pairs and physical intervention in the RoboDojo pilots, while the completion endpoint did not improve.

The audits support an exact statement about these nine evidence pairs. Every native receipt, scene identity, execution mode, and artifact manifest required by the audit passed. The result is therefore definitive for the recorded traces: adding Astra did not change completion in the seven policy-supervision pairs, and the four RoboDojo pairs contain a decrease, a tie, and two increases. The result does not describe scenes, policies, or serving conditions outside the registered pairs.

The three `SimplerEnv` pairs, the four RoboDojo intervention pairs, and the two `BEHAVIOR` gateway pairs have zero discordant binary outcomes. Their exact two-sided McNemar p-values are 1.0. The RoboDojo partial-score contrasts contain one decrease, one tie, and two increases. The repeated `classify_objects` increase is tied to one frozen scene. For the tested cases, attaching Astra did not improve completion, while its effect on partial score was not uniform. The 50-scene RoboDojo panel remains separately collected and descriptive.

Table 8 separates the claims supported by each evidence source. A valid receipt does not turn two separately collected trajectories into a causal comparison.

Evidence source	Matched units	Observed contrast	Licensed conclusion
RoboDojo matched pilots	4 pairs	<code>organize_table</code> : 0.25 vs 0, <code>build_tower</code> : 0.1 vs 0.1, <code>classify_objects</code> : 0 vs 0.4 in two pairs, all failed	Astra has no completion gain in these four audited pairs. Partial-score effects are mixed. The repeated increase is scene-specific. No population effect estimate.
<code>SimplerEnv</code> QwenPI review	3 pairs	22 reviews, 0 edits, equal outcomes and hashes	No observed gain in these tested tasks. Review cost is measurable.
<code>SimplerEnv</code> direct Astra	3 episodes	0/3 success, 40 calls, all native truncated	One-seed direct-control result, with no causal contrast.
<code>BEHAVIOR</code> fresh pair	2 pairs	Native and pass-through both fail at 3,225 steps and Q-score 0	Gateway preservation only. No reasoner effect is tested.
RoboDojo six-configuration panel	50 scenes per configuration	Separately collected single trajectories	Descriptive task and resource patterns only. No independent Astra effect.

Table 8 | Evidence boundary for the claims in this report. “Matched” means that the comparison preserves the registered scene, reset, checkpoint, and execution contract. The external matched samples are too small for a population claim.

6.7. Limits of the evidence

The study covers ten selected RoboDojo tasks with five scenes per task. Each configuration has one separately collected rollout per scene, and hardware, serving, provider sessions, and cadence differ. Exact scene files do not guarantee identical trajectories: supplementary fixed-5 packing repeats complete the same layout in two of three runs, while two frozen stack-bowls runs change from success to failure. The reported counts establish the observed outcomes and cost allocation. Task-level rates are imprecise, and the results describe the recorded embodiments, tasks, and serving conditions.

Identified estimand. For the matched pilots, we report the within-pair native outcome difference $Y_i(\text{Astra}) - Y_i(\text{control})$ for the same saved scene, policy checkpoint, execution cadence, reset identity, and serving conditions. The six-configuration panel is descriptive because its arms were collected separately. Episodes without a verified native terminal receipt are excluded from outcome denominators. The reported pairs pass scene identity, arm identity, cadence, provenance, and terminal receipt audits.

7. Conclusion

starRoboHarness makes policy proposals, reasoner decisions, guarded actions, and native outcomes auditable. QwenPI with Astra completes 17/50 scenes versus 9/50 for QwenPI alone. Across the same saved scenes, pure $\pi_{0.5}$ completes 10, 12, and 18 cases at original, fixed-15, and fixed-5 cadences. The $\pi_{0.5}$ hybrid with Astra completes 20/50. The hybrid has six successes absent from fixed-5, including two language-classification scenes, while fixed-5 alone succeeds on four scenes, including two packing layouts. Thirteen hybrid successes use no corrections, and 29 corrected failures account for 78% of the 910.02 million reported tokens in valid hybrid episodes. Three zero-success tasks consume 372.18 million of those tokens. The cross-benchmark runs confirm that the same evidence boundary transfers to BEHAVIOR and SimplerEnv. In the seven completed matched policy-supervision pairs, Astra produces no completion gain: three SimplerEnv pairs preserve every action and outcome, while all four RoboDojo pairs fail in both arms. Partial scores change from 0.25 to 0 on `organize_table`, remain at 0.1 on `build_tower`, and rise from 0 to 0.4 in two `classify_objects` pairs after 250 and 205 intervention steps. The six-configuration RoboDojo panel remains descriptive because its methods are separately collected single trajectories. The two completed BEHAVIOR VLAct diagnostics record 0/2 native success after 3,225 steps each. This small diagnostic is reported separately from the RoboDojo panel. The matched pairs support a trace-level conclusion, not a population-average effect estimate.

References

- [1] Anthony Brohan et al. *RT-1: Robotics Transformer for Real-World Control at Scale*. [arXiv:2212.06817](#), 2022.
- [2] Anthony Brohan et al. *RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control*. [arXiv:2307.15818](#), 2023.
- [3] Open X-Embodiment Collaboration. *Open X-Embodiment: Robotic Learning Datasets and RT-X Models*. [arXiv:2310.08864](#), 2023.
- [4] Octo Model Team. *Octo: An Open-Source Generalist Robot Policy*. [arXiv:2405.12213](#), 2024.
- [5] Moo Jin Kim et al. *OpenVLA: An Open-Source Vision-Language-Action Model*. [arXiv:2406.09246](#), 2024.
- [6] Kevin Black et al. π_0 : *A Vision-Language-Action Flow Model for General Robot Control*. [arXiv:2410.24164](#), 2024.
- [7] Physical Intelligence et al. $\pi_{0.5}$: *a Vision-Language-Action Model with Open-World Generalization*. [arXiv:2504.16054](#), 2025.
- [8] Cheng Chi et al. *Diffusion Policy: Visuomotor Policy Learning via Action Diffusion*. [arXiv:2303.04137](#), 2023.

- [9] Songming Liu et al. *RDT-1B: a Diffusion Foundation Model for Bimanual Manipulation*. [arXiv:2410.07864](#), 2024.
- [10] Yunfan Jiang et al. *VIMA: General Robot Manipulation with Multimodal Prompts*. [arXiv:2210.03094](#), 2022.
- [11] Siddhant Haldar et al. *BAKU: An Efficient Transformer for Multi-Task Policy Learning*. [arXiv:2406.07539](#), 2024.
- [12] Tony Z. Zhao et al. *Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware*. [arXiv:2304.13705](#), 2023.
- [13] Kevin Black et al. *Real-Time Execution of Action Chunking Flow Policies*. [arXiv:2506.07339](#), 2025.
- [14] Homer Walke et al. *BridgeData V2: A Dataset for Robot Learning at Scale*. [arXiv:2308.12952](#), 2023.
- [15] Alexander Khazatsky et al. *DROID: A Large-Scale In-The-Wild Robot Manipulation Dataset*. [arXiv:2403.12945](#), 2024.
- [16] Remi Cadene et al. *LeRobot: An Open-Source Library for End-to-End Robot Learning*. [arXiv:2602.22818](#), 2026.
- [17] StarVLA Community. *StarVLA: A Lego-like Codebase for Vision-Language-Action Model Developing*. [arXiv:2604.05014](#), 2026.
- [18] S. Yang et al. *Beyond Data Scaling: Representation-Centric Continued Pre-training for Vision-Language-Action Models*. [arXiv:2608.27550](#), 2026. Code: <https://github.com/starVLA/VLAct>.
- [19] Michael Ahn et al. *Do As I Can, Not As I Say: Grounding Language in Robotic Affordances*. [arXiv:2204.01691](#), 2022.
- [20] Wenlong Huang et al. *Inner Monologue: Embodied Reasoning through Planning with Language Models*. [arXiv:2207.05608](#), 2022.
- [21] Jacky Liang et al. *Code as Policies: Language Model Programs for Embodied Control*. [arXiv:2209.07753](#), 2022.
- [22] Danny Driess et al. *PaLM-E: An Embodied Multimodal Language Model*. [arXiv:2303.03378](#), 2023.
- [23] Wenlong Huang et al. *VoxPoser: Composable 3D Value Maps for Robotic Manipulation with Language Models*. [arXiv:2307.05973](#), 2023.
- [24] Xiaoqi Li et al. *ManipLLM: Embodied Multimodal Large Language Model for Object-Centric Robotic Manipulation*. [arXiv:2312.16217](#), 2023.
- [25] Michal Nazarczuk et al. *Closed Loop Interactive Embodied Reasoning for Robot Manipulation*. [arXiv:2404.15194](#), 2024.
- [26] Wentao Zhao et al. *VLMPC: Vision-Language Model Predictive Control for Robotic Manipulation*. [arXiv:2407.09829](#), 2024.

- [27] Yunhai Feng et al. *Reflective Planning: Vision-Language Models for Multi-Stage Long-Horizon Robotic Manipulation*. [arXiv:2502.16707](#), 2025.
- [28] Yifu Yuan et al. *From Seeing to Doing: Bridging Reasoning and Decision for Robotic Manipulation*. [arXiv:2505.08548](#), 2025.
- [29] Shuai Yang et al. *InstructVLA: Vision-Language-Action Instruction Tuning from Understanding to Manipulation*. [arXiv:2507.17520](#), 2025.
- [30] Jiafei Duan et al. *AHA: A Vision-Language-Model for Detecting and Reasoning Over Failures in Robotic Manipulation*. [arXiv:2410.00371](#), 2024.
- [31] Yinpei Dai et al. *RACER: Rich Language-Guided Failure Recovery Policies for Imitation Learning*. [arXiv:2409.14674](#), 2024.
- [32] Lin Liu et al. *LIBERO-RECOVER: Beyond Task Success Towards Failure Recovery in Robotic Manipulation Models*. [arXiv:2609.05178](#), 2026.
- [33] Huihan Liu et al. *Model-Based Runtime Monitoring with Interactive Imitation Learning*. [arXiv:2310.17552](#), 2023.
- [34] GPT-as-Policy contributors. *GPT as Policy*. <https://github.com/anonymous-report-421/GPT-as-Policy>, 2026.
- [35] Mengzhao Jia et al. *Agent as Policy for Robotic Manipulation*. [arXiv:2609.12541](#), 2026.
- [36] Stephen James et al. *RLBench: The Robot Learning Benchmark & Learning Environment*. [arXiv:1909.12271](#), 2019.
- [37] Bo Liu et al. *LIBERO: Benchmarking Knowledge Transfer for Lifelong Robot Learning*. [arXiv:2306.03310](#), 2023.
- [38] Oier Mees et al. *CALVIN: A Benchmark for Language-Conditioned Policy Learning for Long-Horizon Robot Manipulation Tasks*. [arXiv:2112.03227](#), 2021.
- [39] Jiayuan Gu et al. *ManiSkill2: A Unified Benchmark for Generalizable Manipulation Skills*. [arXiv:2302.04659](#), 2023.
- [40] Soroush Nasiriany et al. *RoboCasa: Large-Scale Simulation of Everyday Tasks for Generalist Robots*. [arXiv:2406.02523](#), 2024.
- [41] Ajay Mandlekar et al. *What Matters in Learning from Offline Human Demonstrations for Robot Manipulation*. [arXiv:2108.03298](#), 2021.
- [42] Wilbert Pumacay et al. *THE COLOSSEUM: A Benchmark for Evaluating Generalization for Robotic Manipulation*. [arXiv:2402.08191](#), 2024.
- [43] Shiduo Zhang et al. *VLABench: A Large-Scale Benchmark for Language-Conditioned Robotics Manipulation with Long-Horizon Reasoning Tasks*. [arXiv:2412.18194](#), 2024.
- [44] Tianxing Chen et al. *RoboTwin 2.0: A Scalable Data Generator and Benchmark with Strong Domain Randomization for Robust Bimanual Robotic Manipulation*. [arXiv:2506.18088](#), 2025.
- [45] Tianxing Chen et al. *RoboDojo: A Unified Sim-and-Real Benchmark for Comprehensive Evaluation of Generalist Robot Manipulation Policies*. [arXiv:2607.04434](#), 2026.

- [46] Tianchong Jiang et al. *What Are We Actually Benchmarking in Robot Manipulation?*. [arXiv:2606.04233](https://arxiv.org/abs/2606.04233), 2026.
- [47] Xueyang Zhou et al. *LIBERO-PRO: Towards Robust and Fair Evaluation of Vision-Language-Action Models Beyond Memorization*. [arXiv:2510.03827](https://arxiv.org/abs/2510.03827), 2025.
- [48] Shi Jin et al. *UMI-Bench 1.0: An Open and Reproducible Real-World Benchmark for Tabletop Robotic Manipulation with UMI Data*. [arXiv:2606.10382](https://arxiv.org/abs/2606.10382), 2026.
- [49] Sudeep Dasari et al. *RB2: Robotic Manipulation Benchmarking with a Twist*. [arXiv:2203.08098](https://arxiv.org/abs/2203.08098), 2022.
- [50] Hadas Kress-Gazit et al. *Robot Learning as an Empirical Science: Best Practices for Policy Evaluation*. [arXiv:2409.09491](https://arxiv.org/abs/2409.09491), 2024.
- [51] Apurva Badithela et al. *Reliable and Scalable Robot Policy Evaluation with Imperfect Simulators*. [arXiv:2510.04354](https://arxiv.org/abs/2510.04354), 2025.
- [52] XPolicyLab Community. *XPolicyLab: A Unified Standard and Open Ecosystem for Robot Policy Evaluation and Deployment*. [arXiv:2608.09892](https://arxiv.org/abs/2608.09892), 2026.
- [53] Pablo Valle et al. *Metamorphic Testing of Vision-Language Action-Enabled Robots*. [arXiv:2602.22579](https://arxiv.org/abs/2602.22579), 2026.
- [54] Xuanlin Li et al. *Evaluating Real-World Robot Manipulation Policies in Simulation*. [arXiv:2405.05941](https://arxiv.org/abs/2405.05941), 2024. Code: <https://github.com/simpler-env/SimplerEnv>.
- [55] BEHAVIOR contributors. *2026 BEHAVIOR Challenge Baselines*. <https://behavior.stanford.edu/challenge/baselines.html>, 2026.
- [56] BEHAVIOR contributors. *2026 BEHAVIOR Challenge Evaluation and Rules*. <https://behavior.stanford.edu/challenge/evaluation.html>, 2026.
- [57] NVIDIA. *Isaac-GR00T: An Open Foundation Model for Generalist Humanoid Robot Policies*. <https://github.com/NVIDIA/Isaac-GR00T>, 2025.

A. Ten-task descriptive hybrid index

Group	Task	Hybrid succ.	Hybrid score
A	Organize table	0/5	65.0
A	Classify by language	0/5	0.0
A	Imitate sorting	1/5	36.0
A	Arrange largest number	1/5	37.0
A	Fold clothes	4/5	80.0
B	Put bottles in bin	4/5	82.0
B	Classify objects	0/5	22.0
B	Build tower	4/5	86.0
B	Make Kong	3/5	60.0
B	Pack objects in box	0/5	2.0
Total	50 cases	17/50	47.0

Table 9 | All 50 selected hybrid terminal cases by task group. Success is shown as completed cases over valid cases. This descriptive index is not a separate experiment.

B. Supplementary development cases

Dev task	Native result	Score	Steps	Policy calls	Corrections
Classify by language	Timeout	10	1,100	146	481
Arrange largest number	Success	100	487	34	0
Make Kong	Success	100	381	27	0
Total / mean	2/3 success	70	1,968	207	481

Table 10 | Supplementary development validation: one standard g0/10 layout per task. Score in the final row is the three-case mean. Other final-row quantities are totals.

The development evaluation completes two of three standard g0/10 tasks (Table 10). All 481 correction steps occurred in the failed language-classification episode. Both successes required none. The receipts report 30.91 million input tokens, including 29.83 million cached input tokens, and 72.6 thousand output tokens. No simultaneous pure-policy comparator was run, so this small development sample cannot establish a correction benefit.

Supplementary build_tower case	Native result	Score	Steps	Policy calls
QwenPI_v3, 16-step	Timeout	10	1,050	66
QwenPI_v3, fixed-15 ablation	Timeout	0	1,050	70
Astra + QwenPI_v3	Success	100	761	52
Astra Direct	Timeout	0	1,050	0

Table 11 | Supplementary development evidence for one build_tower layout. Astra Direct issued 210 direct decisions. Its zero policy-call count reflects the absence of a student policy. No row belongs to the frozen panels.

In the supplementary build_tower case, Astra plus QwenPI_v3 succeeded with zero correction steps, whereas pure QwenPI_v3 and Astra Direct timed out (Table 11). This single layout checks native termination across execution paths but cannot establish a population effect.

C. $\pi_{0.5}$ diagnostic tasks and receipt accounting

Native task	Exploratory	Frozen run 1	Frozen run 2
Stack bowls	2/3 (71.7)	3/3 (100.0)	2/3 (71.7)
Stack blocks	0/3 (15.0)	0/3 (10.0)	0/3 (10.0)
Stack blocks by language	0/3 (6.7)	0/3 (0.0)	0/3 (0.0)
Pour liquid into cup	1/3 (33.3)	1/3 (33.3)	1/3 (33.3)
Pour by language	0/3 (0.0)	0/3 (0.0)	0/3 (0.0)

Table 12 | Supplementary pure $\pi_{0.5}$ diagnostics. Cells give native successes out of three and mean score. The frozen runs share 15 verified scenes and one checkpoint.

The two frozen runs have four and three successes on 15 exact scenes. Fourteen pairs retain their score and outcome. Stack-bowls layout 2 changes from success at 100 to failure at 15. This flip limits claims from one rollout per layout. Block stacking and language-conditioned pouring fail in both

runs, whereas ordinary pouring succeeds on one layout. These distinct tasks do not isolate language understanding. None enters the 50-case panel.

The accepted $\pi_{0.5}$ panel uses RoboDojo Pi_05 59999 in one simulator. All 50 scene hashes match the QwenPI panel. Unstable batches substituted two Make Kong scenes and one excluded sorting scene. We reran the selected layouts. Substitutions and five setup failures enter no denominator.

The full pure-policy cadence comparison appears in Table 5.

D. Complete $\pi_{0.5}$ -Astra matched cases

Table 13 lists all 50 selected hybrid cases and their native outcomes. Native timeouts remain valid failures.

Table 13 | 50 matched cases with native completion/score and hybrid resource use. Corrections combine edited and recovery steps. Tokens include cached input. Unlabeled task rows are standard layout 0. Rand. marks the random variant. Coverage is 50/50. All rates use native-valid terminal cases.

Task	Pure $\pi_{0.5}$	$\pi_{0.5}$ + Astra	Steps	Time (s)	Corr.	Tokens (M)
Organize table	Fail / 75	Fail / 50	1,000	4,320	330	28.47
Organize table (layout 1)	Fail / 50	Fail / 25	1,000	5,336	210	35.81
Organize table (layout 2)	Fail / 0	Fail / 50	1,000	2,635	470	17.61
Organize table (layout 3)	Fail / 25	Fail / 75	1,000	1,833	125	11.76
Organize table (layout 4)	Fail / 25	Fail / 75	1,000	3,003	205	23.87
Classify by language	Fail / 0	Success / 100	711	1,800	55	15.44
Classify by language (layout 1)	Fail / 0	Fail / 10	1,100	2,197	210	14.17
Classify by language (layout 2)	Fail / 0	Fail / 40	1,100	2,122	105	23.64
Classify by language (layout 3)	Fail / 0	Fail / 0	1,100	2,834	284	14.84
Classify by language (layout 4)	Fail / 0	Success / 100	942	1,659	165	12.09
Imitate sorting	Fail / 0	Fail / 50	1,564	7,878	385	44.78
Imitate sorting (layout 1)	Fail / 0	Fail / 0	442	632	0	5.12
Imitate sorting (layout 2)	Fail / 0	Fail / 50	1,600	2,515	113	18.14
Imitate sorting (layout 3)	Fail / 0	Fail / 30	1,600	3,290	370	22.77
Imitate sorting (layout 5)	Fail / 0	Fail / 0	536	940	11	7.44
Arrange largest number	Fail / 15	Fail / 15	1,050	3,038	200	28.68
Arrange largest number (layout 1)	Fail / 0	Success / 100	564	1,183	15	11.52
Arrange largest number (layout 2)	Fail / 0	Fail / 25	1,050	3,144	145	23.26
Arrange largest number (rand. 0)	Fail / 0	Fail / 30	1,050	4,430	220	26.41
Arrange largest number (rand. 1)	Fail / 0	Fail / 40	1,050	2,956	310	18.07
Fold clothes	Fail / 20	Success / 100	300	650	0	3.95
Fold clothes (layout 1)	Fail / 20	Success / 100	303	585	0	4.84
Fold clothes (layout 2)	Fail / 20	Success / 100	296	652	0	3.75
Fold clothes (rand. 0)	Fail / 20	Success / 100	304	813	15	5.41
Fold clothes (rand. 1)	Fail / 0	Success / 100	302	782	0	4.41
Put bottles in bin	Success / 100	Success / 100	477	642	0	9.38
Put bottles in bin (layout 1)	Success / 100	Success / 100	323	1,317	0	7.76
Put bottles in bin (layout 2)	Fail / 40	Success / 100	440	875	0	7.51
Put bottles in bin (layout 3)	Success / 100	Success / 100	419	780	0	6.84
Put bottles in bin (layout 4)	Fail / 40	Success / 100	700	1,514	5	12.74
Classify objects	Fail / 15	Success / 100	786	1,122	0	13.01
Classify objects (layout 1)	Success / 100	Success / 100	779	1,073	0	14.15
Classify objects (layout 2)	Fail / 0	Fail / 15	1,100	2,542	215	27.66
Classify objects (layout 3)	Fail / 0	Fail / 15	1,100	3,231	193	26.87
Classify objects (layout 4)	Fail / 15	Success / 100	890	1,567	15	18.71
Build tower	Fail / 30	Success / 100	723	931	0	6.09

Continued on next page

Task	Pure $\pi_{0.5}$	$\pi_{0.5}$ + Astra	Steps	Time (s)	Corr.	Tokens (M)
Build tower (layout 1)	Success / 100	Fail / 0	1,050	4,116	380	32.70
Build tower (layout 2)	Success / 100	Fail / 10	1,050	4,192	300	30.01
Build tower (layout 3)	Success / 100	Success / 100	717	1,219	0	11.48
Build tower (layout 4)	Fail / 10	Success / 100	714	1,271	0	12.57
Make Kong	Success / 100	Fail / 0	600	2,884	120	24.71
Make Kong (layout 1)	Fail / 0	Success / 100	407	1,159	40	9.72
Make Kong (layout 2)	Fail / 0	Fail / 0	600	1,887	120	13.51
Make Kong (layout 3)	Success / 100	Fail / 0	600	3,014	290	23.98
Make Kong (layout 4)	Success / 100	Fail / 0	600	2,380	160	17.96
Pack objects in box	Fail / 10	Fail / 25	1,300	6,082	450	39.37
Pack objects in box (layout 1)	Fail / 50	Fail / 10	1,300	2,635	380	21.62
Pack objects in box (layout 2)	Fail / 10	Fail / 25	1,300	3,136	265	18.50
Pack objects in box (rand. 0)	Fail / 50	Fail / 50	1,300	3,796	290	36.57
Pack objects in box (rand. 1)	Fail / 50	Fail / 50	1,300	4,734	440	40.37
50 valid cases					7,606	910.02

D.1. First matched pilot diagnostic

A matched pilot compares fixed-5 control with Astra supervision on organize-table layout 1. Both arms use the same checkpoint, reset identity, hardware, and 1,000-step horizon. Initial robot state, end-effector pose, instruction, and remaining horizon are bitwise equal. Both episodes time out after 200 decision windows. On the report’s 0–100 scale, control scores 25 and Astra scores 0, a difference of –25 points. Astra executes 140 correction steps, while control executes none. Terminal receipts, cadence, scene identity, and artifact hashes pass the audits. Camera arrays and the first policy proposals differ slightly across fresh processes, so matching the reset does not reproduce an identical policy trajectory. This pair documents a supervised timeout despite substantial correction. It does not estimate the average effect of supervision or the effect of an individual correction. The pilot is reported as a supplementary matched case and does not enter the six-configuration panel.

E. Policy complementarity by task

Task	QwenPI		QwenPI + Astra		$\pi_{0.5}$	
	Success	Score	Success	Score	Success	Score
Organize table	0/5	10.0	0/5	65.0	0/5	35.0
Classify by language	0/5	0.0	0/5	0.0	0/5	0.0
Imitate sorting	0/5	1.0	1/5	36.0	0/5	0.0
Arrange largest number	0/5	2.0	1/5	37.0	0/5	3.0
Fold clothes	1/5	32.0	4/5	80.0	0/5	16.0
Put bottles in bin	3/5	64.0	4/5	82.0	3/5	76.0
Classify objects	0/5	6.0	0/5	22.0	1/5	26.0
Build tower	4/5	82.0	4/5	86.0	3/5	68.0
Make Kong	1/5	20.0	3/5	60.0	3/5	60.0
Pack objects in box	0/5	20.0	0/5	2.0	0/5	34.0
All tasks	9/50	23.7	17/50	47.0	10/50	31.8

Table 14 | Three observed methods on the same 50 saved RoboDojo scene layouts. Success is the number of completed cases, and Score is the mean native score on a 0–100 scale. The $\pi_{0.5}$ rows come from a separate pure-policy campaign.

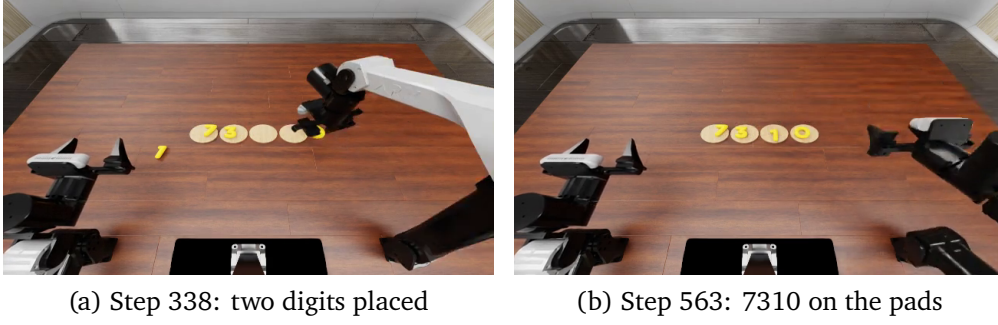


Figure 3 | Native first-camera frames from number-arrangement layout 1. The first precedes the reasoner’s first bounded correction. The second shows all four digits on their target pads shortly before the native success at step 564. The frames describe one trajectory and do not isolate the effect of supervision.

F. Intervention and failure trajectories

The cases below link recorded decisions to native observations and outcomes. They illustrate the executed mechanisms and unresolved failure states. They do not supply the counterfactual trajectory required to establish that a correction was necessary or sufficient.

F.1. Local recovery in folding and number arrangement

On random folding layout 0, head and wrist observations after the first 15 student steps place the left gripper over a yellow book instead of the white sleeve. Three five-step end-effector corrections move the hand inward. The next policy proposal approaches the cuff, and delegated actions fold and release the garment before native success at step 304. The episode has 289 student steps and 15 corrected steps. Three fixed-5 pure-policy runs, including the main-panel run, fail with score zero on the saved layout. On random layout 1, the hybrid instead succeeds without corrections at step 302, while all three pure cadences fail. The two hybrid folding successes therefore involve different control paths.

On number-arrangement layout 1, three five-step corrections align and lower digit 0 over the right-most pad and move a held digit 1 toward the third pad. Remaining policy actions produce native success at step 564. The episode reports 11.52 million tokens, including 11.28 million cached-input tokens. Figure 3 shows the arrangement before correction and near termination. Original and fixed-15 pure policy each fail at score zero, while fixed-5 also succeeds. That successful pure-policy run demonstrates that this saved scene admits a completed trajectory without Astra.

F.2. Semantic redirection in language classification

Language-classification layout 0 requires watches in the middle basket and wooden toys in the right basket. Eleven five-step corrections redirect a held watch and interrupt proposed transfers of a wooden toy toward the middle basket. The supervisor stages that toy on clear tabletop, then delegates 13 policy prefixes that complete placements and arm return. Figure 4 shows the visible sequence. All three pure cadences fail on the saved scene, while the hybrid receives native success and score 100.

On layout 4, the supervisor uses 145 corrected steps by step 260 to stage and redirect a pepper after proposals toward an incorrect basket. Later delegated actions complete the placements, with native

success at step 942 after 165 corrected steps in total and 12.09 million reported tokens. The hybrid completes two of five language-classification scenes, has mean score 50, and uses 819 corrected steps and 80.17 million tokens. These successful redirections coexist with three native failures, so the individual trace does not establish transfer across the task.

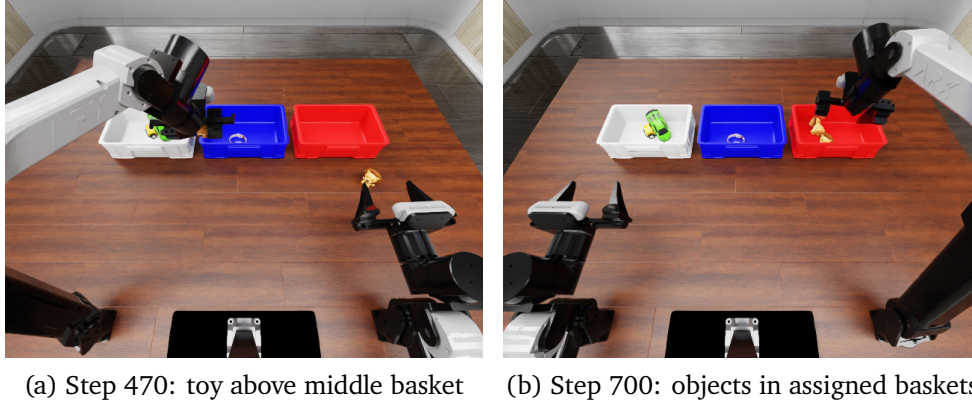


Figure 4 | High-camera observations from the native-successful $\pi_{0.5}$ -Astra language-classification layout. The task requires watches in the middle basket and wooden toys in the right basket. Between these views, the supervisor stages a wooden toy for regrasp before delegating the final placements. The sequence is trajectory evidence, not a controlled causal comparison.

F.3. Assembly failures and repeated tile recovery

Build-tower layout 1 reaches the 1,050-step limit with score zero after 380 corrected steps and 32.70 million reported tokens. Its 76 five-step corrections revisit displaced supports and board placement. Late observations show a board standing on its edge across the supports. The subsequent corrections do not produce a completed stack. Original and fixed-5 pure $\pi_{0.5}$ succeed on this saved scene. The hybrid’s three build-tower completions contain no corrections, while its two failed scenes contain 680 corrected steps in total.

Make Kong has one hybrid completion in five scenes, versus three at original pure-policy cadence and none at either shorter cadence. In the successful hybrid layout 1, eight corrections target a displaced upright matching tile. Later observations record all three matching tiles face-up before success at step 407. This episode uses 40 recovery steps and 9.72 million tokens. Layout 0 instead uses 120 recovery steps on a dropped matching tile and fails at the 600-step limit, reporting 24.71 million tokens.

Layout 2 isolates a repeated recovery sequence. Three matching South tiles are face-up by step 262, but an unrelated five-dot tile is knocked face-up during subsequent placement. From step 435 onward, 105 of the episode’s 120 corrected steps attempt to grasp, rotate, and restore that tile. It remains face-up in the final frame, and native score is zero. The trace identifies the unresolved disturbance and where the remaining horizon was spent.

F.4. Packing geometry and rollout variation

Packing layout 1 times out at 1,300 steps with score 10, after 920 student steps and 380 corrected steps. It makes 140 reasoner decisions and reports 21.62 million tokens. At step 908, the controller assesses that all four objects are at the box but leaves their orientations unverified. It then uses 310 of the remaining 392 steps on 62 corrections directed at a displaced toothbrush. Native credit never advances beyond one-object placement. Original and fixed-15 pure policy also fail on this scene, with

scores 50 and 25. The geometric checks, rather than the controller’s verbal assessment, determine benchmark progress.

On random packing layout 0, the hybrid fails with score 50 after 290 recovery steps and 36.57 million tokens. Three fixed-5 pure-policy runs on the saved scene yield two successes and one timeout at score 50. The repeated control exposes variability even with the scene and cadence fixed. On standard layout 0, fixed-5 succeeds while the hybrid fails at score 25 after 450 corrected steps. Across five packing scenes, the hybrid completes none and uses 1,825 corrections and 156.42 million tokens.

F.5. Persistent partial credit and instruction alignment

Sorting layouts 0 and 2 reach score 50 at the fourth ordered-prefix transition but fail the task. The hybrid uses 385 corrected steps and 44.78 million tokens on the first, versus 113 steps and 18.14 million tokens on the second. Layout 1 contains no correction: all 442 control steps are delegated, yet its 30 reasoner decisions report 5.12 million tokens and its native score is zero. The task has no hybrid completion, mean score 26, and 98.25 million reported tokens. Equivalent partial scores can therefore conceal different trajectories and costs.

Organize-table layout 2 uses 470 corrected steps and reports 17.61 million tokens before timeout at score 50. Of those corrections, 105 steps attempt to grasp and pull the drawer handle. The instruction asks for drawer opening, while the native completion check combines four object-placement conditions and arm return without a separate drawer-opening condition. Fixed-5 pure policy succeeds on this saved scene. Layout 4 reaches score 75 but fails after 205 corrected steps and 23.87 million tokens. That persistent stage records three of four item checks with open grippers. Completion separately requires all four checks and arm return.

Native scoring also distinguishes motion from credited placement. Language classification requires an entire category in the correct basket, no objects from other categories there, and open grippers. Sorting requires a correctly ordered prefix in the target basket with later objects outside it. Packing requires prescribed object orientation and containment, an upright aligned box, and open grippers. A zero score need not imply no object moved, and a nonzero stage need not certify the final arrangement.

G. Data provenance

The reported RoboDojo panel uses ten selected tasks with five saved scenes per task. The QwenPI and pure $\pi_{0.5}$ scene files share the same manifest and scene digests. The hybrid rows retain their source exports, case identities, reset metadata, native receipts, and artifact hashes. Group A and Group B keep their collection provenance separate because the policy counterparts in Group B were collected after the hybrid cases.

All rows entering a success or score denominator have a native terminal receipt. Native timeouts retain the score assigned by the evaluator. Provider failures, transport aborts, and episodes without a terminal receipt are excluded from the reported denominators. The completed replacement runs use fresh trial identities. The audit scripts check the source digests, scene identity, arm identity, and receipt fields before table generation.

The external validation rows use separate action contracts and checkpoints. The BEHAVIOR QwenDual head uses the 23-dimensional R1Pro contract. Its native receipt is listed in the repository evidence matrix and is not pooled with the RoboDojo panel.