

SO-101 Bench: Accessible Spatial Manipulation

Truman Hickok

Independent

hickoktruman@gmail.com

 Website ·  GitHub ·  Hugging Face

Low-cost arms such as the SO-101 have become the most common entry point to robot learning, yet no released policy performs spatially constrained, language-conditioned tasks on them out of the box, and existing SO-101 benchmarks run only on physical hardware with simple tasks and few held-out objects. We introduce SO-101 Bench, a simulated benchmark for language-conditioned spatial manipulation built on an Isaac Lab digital twin of a real SO-101 setup. It has four tasks scored as five families, 138 objects in seen, unseen-instance and unseen-class splits, 916 evaluation episodes with subsets for held-out appearances, rotate-to-fit planning and occlusion memory, and 3,629 simulated demonstrations on which every learned policy is fine-tuned under an equal compute budget. The twin ranks policies as the real robot does: five policies trained without real data, four VLAs and an LLM agent, keep their exact ranking when the same weights run on the physical arm in an unseen environment with unseen objects (Spearman $\rho = 1.00$, MMRV = 0.000), although simulation overestimates success. Across nine VLAs and world-action models, the best learned policies score near 61%, with spatial placement, unseen object classes and rotate-to-fit planning remaining open. An LLM agent that writes code and hands grasping to a learned policy scores 83.2%, and an ablation shows that this lead comes from the choice of harness and the choice of language model.

Keywords: Robot Manipulation, Benchmarks, Sim2Real

1 Introduction

Low-cost arms have become the most common entry point to robot learning. The open-source SO-101 [1, 2] is a 3D-printed arm driven by six serial-bus servos; a follower arm costs about \$120–150 in parts [2, 3], against roughly \$4,900 for one ALOHA arm [4]. LeRobot [5] supports it from teleoperation to deployment, and SO-family arms produce most of the community robot datasets on the Hugging Face Hub.¹ Open models now train on these data: SmolVLA was pretrained on 481 community datasets [6], and MolmoAct2 has a variant post-trained on 38k curated SO-100/101 episodes [7]. These models, however, are starting points for fine-tuning on a user’s own demonstrations: MolmoAct2-SO100/101 demonstrates out-of-the-box performance on basic tasks, and to our knowledge no released policy performs spatially constrained, language-conditioned tasks on an SO-101 out of the box in a new environment with new objects. Evaluation, too, has not kept pace: the SO-101 benchmarks released this year run only on physical hardware [8, 9, 3, 10]. Their tasks include insertion, pouring, wiping and spatial grounding; Spa-Bench tests relational placement, referential descriptions, ordinal position and counting, with familiar constituents in its primary evaluation and a separate novel-object pilot [10]. They are also costly: real-robot studies typically afford tens of rollouts per condition [11], and results depend on the particular build (servo offsets differ between arms [9]; one arm farm reports a misaligned camera and objects that wore down during evaluation [3]).

Simulation removes these costs, but a simulated benchmark is useful only if its rankings hold on hardware. Widely used suites target other embodiments [12, 13, 14], the one simulated SO-101 task set we know of contains eight language-free cube and can tasks [15], and the most cited suite is saturated: a 0.54M-parameter

¹559 of 976 (57%) randomly sampled LeRobot-tagged datasets declare an SO-100/101 `robot_type` in their metadata (our count, 26 October 2026).

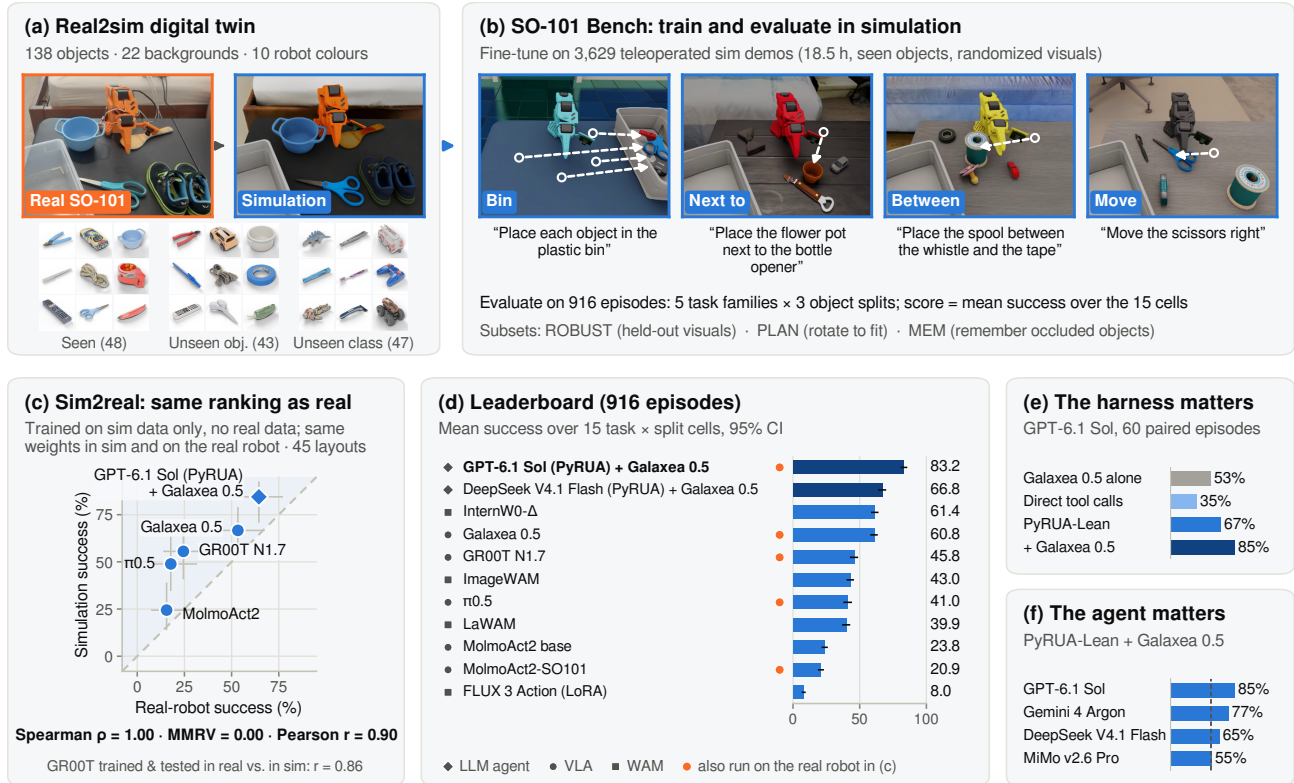


Figure 1: **SO-101 Bench at a glance.** (a) Real SO-101 and its digital twin. (b) The four tasks. (c) Sim2real: five policies trained without real data keep their ranking on the real robot. (d) Leaderboard. (e) Harness ablation. (f) The language model matters.

policy reaches 95.1% on LIBERO [16], where clean averages for three protocol-faithful vision-language-action model (VLA) and world-action model (WAM) evaluations span only 1.36 points [17]. Real2sim evaluation [18, 19, 20, 21] shows that a carefully built digital twin can predict real-world rankings, but the prediction is specific to the simulator, policies and setup on which it was validated: in a five-policy study spanning 1,115 real rollouts across three simulators, SIMPLER showed weak ranking agreement with the real robot [22]. These studies also mostly run in one direction, placing policies trained on real data inside a twin. As policies are increasingly trained on simulated data alone [23, 24], a benchmark should also show that its ranking survives when the same sim-trained weights run on a real robot. Finally, it should measure what aggregate success hides: a policy can complete a task without understanding it: on LIBERO, a probe that replaces instruction text with a task-ID embedding nearly matches the best published results [25], and on a real SO-101, VLAs that manipulate reliably can fail to adapt to OOD spatial relations [10].

We introduce SO-101 Bench, a simulated benchmark for language-conditioned spatial manipulation with a single SO-101 arm, built so that its rankings carry over to the real robot (Fig. 1). Its core is an Isaac Lab [26] digital twin of a real SO-101 setup with 138 household objects in three generalization splits: seen (in the training dataset), unseen object of a seen class, and unseen class, with 37 unseen-class objects kept private. Four tasks (place every object in a bin; place an object next to another; place it between two others; move it in a direction until it reaches the boundary object) are scored automatically from simulator state. They require the policy to pick out the named object among same-class distractors, grasp unfamiliar geometry and place it under spatial constraints. The primary evaluation has 916 feasibility-checked episodes over 15 task-family × split cells, with subsets for held-out appearances (ROBUST), rotating an object to fit a tight space (PLAN) and remembering occluded objects (MEM). Every learned policy is fine-tuned on the same 3,629 simulated teleoperated demonstrations of seen objects under an equal 36 h × 4 × A100 compute budget.

Two studies test whether the twin can stand in for the real robot. Independently trained real-only and sim-only GR00T N1.6 policies find the same conditions hard (Pearson $r = 0.86$ across the 15 cells) and fail in the same ways (Sec. 4). Five policies that never saw real data from our setup, four VLAs and an LLM agent, then run with identical weights in the twin and on the physical SO-101 in an environment unseen in training, with objects from all three splits; SO-101 Bench policies work out of the box in unseen real environments with unseen objects. Simulation ranks them exactly as the real robot does (Spearman $\rho = 1.00$, MMRV = 0.000, 95% CI [0.000, 0.169]), although it overestimates every policy’s success by 9 to 31 percentage points (Sec. 5). On the full leaderboard of nine VLAs and WAMs, the best learned policies, InternW0- Δ [27] and Galaxea 0.5 [28], are statistically tied near 61; spatial placement and unseen classes are where they fail, and none exceeds 17% on PLAN. The strongest entry is an LLM agent: GPT-6.1 Sol [29] writing code in the PyRUA-Lean harness [30] and handing grasps to Galaxea 0.5 scores 83.2 and solves 75% of PLAN. Its lead comes largely from the harness. On 60 paired episodes, Sol succeeds on 35% (21/60) with direct tool calls, 67% (40/60) with PyRUA-Lean and 85% (51/60) with the Galaxea 0.5 hand-off, against 53% (32/60) for Galaxea 0.5 alone, and the lead holds on the real robot, 64% (29/45) against 53% (24/45).

Our contributions are:

- **SO-101 Bench**, a digital-twin benchmark for language-conditioned spatial manipulation on a low-cost arm, with instance- and class-level object splits, ROBUST, PLAN and MEM subsets and automatic scoring, released with the `so101_bench_sim` (3,629 simulated episodes) and `so101_bench_real` ($\sim 4,400$ real episodes) datasets.
- **Evidence that the twin ranks policies as the real robot does**: condition- and failure-level correspondence between independently trained real-only and sim-only policies, and an exact `sim2real` ranking of five policies, including an LLM agent, run with identical weights and no real data from our setup; these policies perform language-conditioned tasks out of the box in an unseen real environment with unseen objects.
- **A leaderboard of nine VLAs and WAMs and two LLM agents**, with every learned policy fine-tuned on the same data and compute, showing that spatial placement, unseen object classes and rotate-to-fit planning remain open.
- **Harness and model ablations** showing that, in our setting, a code-as-policy harness that hands grasping to a learned policy outperforms either on their own; Direct action generation by the agent proves expensive and ineffective, and 6.1 Sol compares favorably to other agents.

2 Related Work

Low-cost hardware and SO-101 benchmarks. Cheap hardware has repeatedly widened who can do robot learning. ALOHA showed that a low-cost bimanual rig can learn fine manipulation [31], and 2026 platforms push prices further, from a \$1,688 bimanual mobile manipulator [32] to the SO-101, at about a fortieth of the price of an ALOHA arm [4]. LeRobot [5] and community data have made the SO-100/101 a training platform for open VLAs [6, 7]. Four 2026 benchmarks evaluate VLAs on physical SO-101 arms. Yu and Qiu [8] analyse failure and recovery on four tasks, VLA-REPLICA is a sub-\$1,100 kit reproduced across sites [9], ArmnetBench scores 2,518 rollouts of seven policies on an arm farm [3], and Spa-Bench probes recombinations of familiar objects, spatial concepts and actions in its primary evaluation, with a separate novel-object pilot [10]. All four use real robot demonstrations for task-specific adaptation and evaluate on physical hardware. In simulation, the SO-101 Task Set contains eight language-free tasks for visual reinforcement learning with `sim2real` transfer [15], and LeIsaac provides SO-101 teleoperation and data-collection tooling in Isaac Lab [33]. To our knowledge, SO-101 Bench is the first language-conditioned SO-101 benchmark with held-out object splits whose simulated policy ranking is validated on the real arm. It also represents one of, if not the most, challenging task suites handled by an SO-101 up to this point.

Real2sim evaluation and `sim2real` transfer. Whether simulated evaluation predicts real performance was studied early for navigation [34]; for manipulation, SIMPLER [18] showed that an aligned simulator can rank

real-world policies and introduced the MMRV metric. Twins built from scans, neural reconstructions or generated assets now report strong score or ranking agreement within their validated settings [19, 20, 35, 36, 37, 38]; SimFoundry reaches Pearson 0.911 and MMRV 0.018 over five policy architectures [21]. Two 2026 results temper this. Validity does not carry across setups: re-tested against new real rollouts, SIMPLER’s agreement with the real-robot ranking is weak [22]. Reconstruction quality also matters: an authored object-and-scene pipeline achieved sim–real correlation of 0.90 versus 0.51 for a default reconstruction in a ten-cell study [39], consistent with our tuning of each reconstructed object against its physical counterpart (Appendix E). Learned world models are an alternative evaluator [40], but released models often predict success on verified failures [41]. In parallel, policies trained only on simulated data now transfer to real robots: MolmoB0T reaches 79.2% on real pick-and-place without real demonstrations [23], OASIS pairs image-to-3D assets with teleoperation inside simulation [24], and LEGS and Splat2Act fine-tune pretrained VLAs using only synthetic robot demonstrations [42, 43]. We deploy five sim-trained policies, including an LLM agent, with identical weights and test whether their ranking survives.

LLM agents for manipulation. Language models have long planned over learned skills [44] or written programs that compose perception and control [45, 46], and hierarchical systems pair high-level language reasoning with low-level action generation [47, 48]. In 2026 such agents became competitive with end-to-end policies: coding agents are now benchmarked on manipulation directly [49, 50, 51]; GPT-6 Astra with direct end-effector tool calls ranked above all forty public-policy entries in its September 2026 RoboDojo comparison [52, 53]; RoboICL reports higher aggregate progress than its strongest listed baseline on a thirty-task subset [54], and PyRUA-Lean exceeds the best public entries in its reported leaderboard snapshot [30]. A parallel line treats the VLA as a tool: Harness VLA exposes a frozen VLA as a retryable contact-rich primitive [55], VoLo steers one as an interruptible tool [56], Gemini Robotics ER 2 hands motor execution to any VLA [57], and a systematic study of such hierarchies finds that model choice and interface jointly shape performance [58]. Harness effects are large. With the same planner and primitives, one study reports that writing code instead of calling tools raises success from 63.1% to 71.7% [30]; a better harness lets a weaker model beat a stronger one [59]; and outside robotics the scaffold alone moves accuracy by up to 28 points within one model [60]. These studies usually compare agents with published entries trained under each entry’s own protocol. Simulated agent evaluations, ours included, commonly pause the simulator while the model computes, which hides its real-time cost [61], and frontier models controlling physical arms directly are not yet precise or efficient enough to finish tasks within a practical action budget [62]. SO-101 Bench compares agents and learned policies on the same episodes after training every learned policy on the same data and compute, ablates harness and model separately, and runs the agent on the real robot.

Manipulation benchmarks and evaluation practice. Simulated benchmarks for generalist policies [12, 13, 14, 53, 63] face saturation and train–test overlap. LIBERO instruction conditioning largely selects among memorized tasks [16]; policies above 90% there fall below 30%, or to zero, under modest perturbations [64, 65]; and an audit of five suites finds shortcut solvability, state-of-the-art gains not established as statistically significant, creeping overfitting and dependence on the training-data source [25]. Responses include generalization suites [66, 63], capability suites for memory of occluded objects [67] and spatial grounding [10]. Real-world evaluation has moved to distributed pairwise comparison [68], autonomous evaluation cells [69], and statistical methods that separate policies with few trials [11, 70]. SO-101 Bench adopts several of these lessons: it separates unseen instances from unseen classes and keeps part of the latter private, fine-tunes every policy on the same seen-object data under one compute budget, selects checkpoints on a separate validation set, and reports confidence intervals over 916 episodes. It also scores two spatial skills that, to our knowledge, few benchmarks test in closed loop: placing an object between two referents, and rotating a grasped object to fit (PLAN).

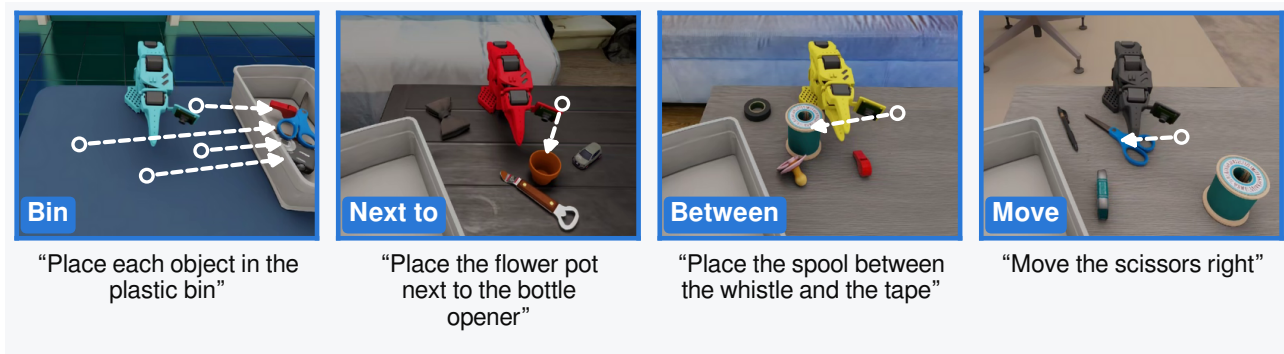


Figure 2: **The four SO-101 Bench tasks.** Bin, next-to, between and directional move, with the target paths overlaid on successful simulation episodes. The task panels repeat Fig. 1b.

3 SO-101 Bench

SO-101 Bench evaluates language-conditioned tabletop manipulation with a single SO-101 arm [1]. It is designed to test whether a policy can turn semantic understanding into precise physical execution: identifying the right object among distractors, grasping objects of unfamiliar geometry, and placing them under (often precise) spatial constraints. Every component below is shared by all evaluated policies.

3.1 Tasks

The benchmark comprises four language-conditioned tasks (Fig. 2), scored as five task families.

- **Place each object in the plastic bin.** The pure grasping task: the policy must identify, grasp, transport and deposit every object on the table. The bin moves between episodes. We evaluate a 1-object and a 4-object condition as separate families.
- **Place [object 1] next to [object 2].** The policy must place the target close to a referent without disturbing other objects, adding semantic grounding and basic spatial placement.
- **Place [object 1] between [object 2] and [object 3].** The target must end centred between two referents, which makes the task sensitive to object extent, orientation and fine placement; it is the most demanding task.
- **Move [object 1] [direction].** The policy must push or carry the target along a straight path in a commanded direction until it reaches the boundary formed by the next object in its way.

The bin task stresses grounding, grasp strategy and grasp precision; the three spatial tasks additionally require relational grounding, spatial reasoning, fine-grained placement and robustness to partial occlusion. To keep language grounding nontrivial, distractors of the same class or colour are included so that class identity alone is insufficient, and colour words are omitted from instructions unless necessary (i.e when two objects of the same class are on the table).

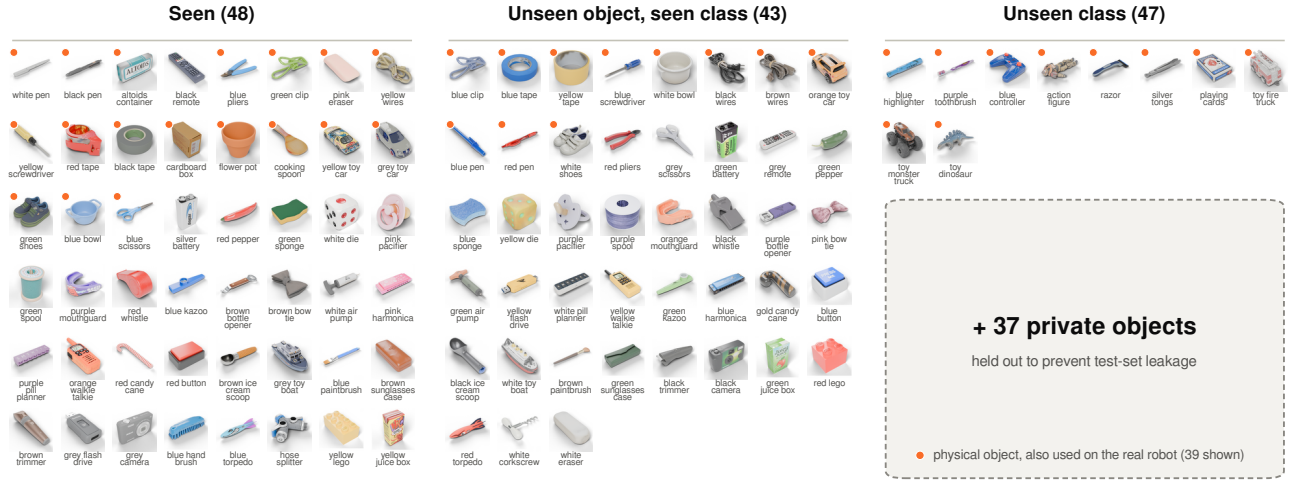


Figure 3: **Simulation objects** by generalization split. Orange dots mark physical objects that are also used in the real-robot evaluation. The 37 private unseen-class objects are held out to prevent test-set leakage and are not shown.

3.2 Objects and Generalization Splits

The benchmark uses 138 household objects in three generalization splits (Fig. 3): **seen** (48 objects that appear in training), **unseen object, seen class** (43 objects whose class, e.g. bowl or pen, appears in training but whose instance does not), and **unseen class** (47 objects of entirely novel classes). The splits separate transfer across *instances* from transfer across *categories*. Objects were chosen to maximize diversity of geometry and of domain (e.g. office, kitchen, sports, toys), with priority on irregular shapes. Every object was verified through teleoperation to be graspable by the SO-101, so that failures reflect the policy rather than the hardware. Of the unseen-class objects, 37 are kept private to prevent test-set leakage.

All objects were created using the image-to-3D tools in 3D AI Studio [71]. The ~ 60 physical objects were reconstructed from photographs of the real items and their simulated appearance and physical properties were tuned to match them; the remaining objects were generated from internet or AI-generated images and passed through the same pipeline (Appendix E). All simulated objects are rigid bodies.

3.3 Digital Twin

All evaluation runs in a digital twin of the original real-world setup, built in Isaac Lab [26] from a scan of the room and the object assets of Sec. 3.2. Physics runs at 240 Hz and the policy acts at 30 Hz. Policies observe an overhead camera and a wrist camera, both at 640×480 , and the arm’s joint positions. The twin includes 22 background environments: 18 are used in training and in the varied evaluation episodes, and 4 are held out (Appendix F). The robot is rendered in one of 6 training colours or one of 4 held-out colours. The real arm’s joint calibration is aligned to the simulated kinematics so that the same policy weights can run in both domains (Sec. 5).

3.4 Scoring

Episodes are scored automatically from simulator state. Positions are measured in the tabletop plane from where objects rest at the start of the episode. A success condition must hold continuously for 5 s; a failure condition ends the episode once it has held for 5 s, so briefly undoing a mistake is allowed. The robot is considered to touch an object when any link presses on it with more than 0.05 N. Episodes time out after 30 s (75 s for 4-object bins).

- **Bin:** every object ends with its reference point (the centre of its top-down bounding box) inside the bin’s footprint and untouched by the robot. Fails if the bin moves more than 0.5 in.

PLAN: rotate a boat to fit between referents



"Place the toy boat between the pill planner and the camera"

MEM: move towards an occluded boundary



"Move the tape backwards"

Figure 4: **PLAN and MEM examples** (start, middle and end of successful primary-evaluation episodes). Left: GPT-6.1 Sol with PyRUA-Lean and Galaxea 0.5 rotates the toy boat to fit between the pill planner and the camera (episode 529), which all nine learned policies fail. Right: ImageWAM moves the tape backwards (episode 300), retaining the boundary object position while it is occluded during the move; four of the nine learned policies solve this episode. All objects and backgrounds shown are public.

- **Next to:** the target’s top-down outline ends within 2 in of the referent’s outline.
- **Between:** the target’s reference point ends within 1.75 in of the line through the two referents’ reference points, and closer to each referent than the referents are to each other.
- **Move:** with an object ahead of the target, the target’s leading edge ends within 2 in of that object’s facing edge; otherwise the target advances at least 2 in. The target must stay within 2 in of the straight path, and the boundary object must not move more than 0.5 in along the direction.

In all three spatial tasks the target must move at least 1 in and must not be touched by the robot at the end, and the episode fails if any non-target object (referents included) moves more than 1 in, or if the target or a held object stays in contact with another object for more than 5 s.

3.5 Training Data

All learned policies are fine-tuned on `so101_bench_sim`: 3,629 teleoperated episodes (18.5 h at 30 fps) recorded in the digital twin using seen objects only. It contains 1,892 bin episodes (including bin instructions that name a single target among other objects), 1,150 next-to and between episodes, and 587 move episodes; 3,004 episodes have four objects on the table. Most episodes are rendered with randomized lighting, camera intrinsics and extrinsics, background and robot colour; the canonical setup remains in the training distribution. A variant, `so101_bench_sim_WM`, adds the episode’s initial overhead and wrist frames for policies that accept them. We also release `so101_bench_real`, approximately 4,400 real-robot teleoperated episodes (~20 h) across approximately 35 objects.

3.6 Primary Evaluation

The primary evaluation contains 916 scored episodes spanning the 15 task-family $\times$ split cells, with every object appearing in each family it supports. Feasibility (spacing between objects) is checked for every layout so that a poor arrangement cannot make success impossible. About half the episodes use the canonical appearance and half vary the background and robot colour. We report the **benchmark score**, the mean success rate over the 15 cells, with a 95% bootstrap confidence interval that resamples episodes within cells. Three overlapping subsets probe specific capabilities: **ROBUST** (178 episodes with held-out backgrounds and robot colours), **PLAN** (48 episodes in which the grasped object must be rotated to fit into a tight space), and **MEM** (51 episodes in which the policy must remember the position and 3D footprint of an object that becomes occluded). Figure 4 shows a successful example of each.

3.7 Fine-Tuning Protocol

Every learned policy is fine-tuned with an equal compute budget of 36 h on 4 $\times$ A100-80GB GPUs, following its authors’ recommended recipe where one exists (Appendix C). Checkpoints are evaluated on a held-out 39-episode validation set, and the best checkpoint is used for the primary evaluation. Three policies

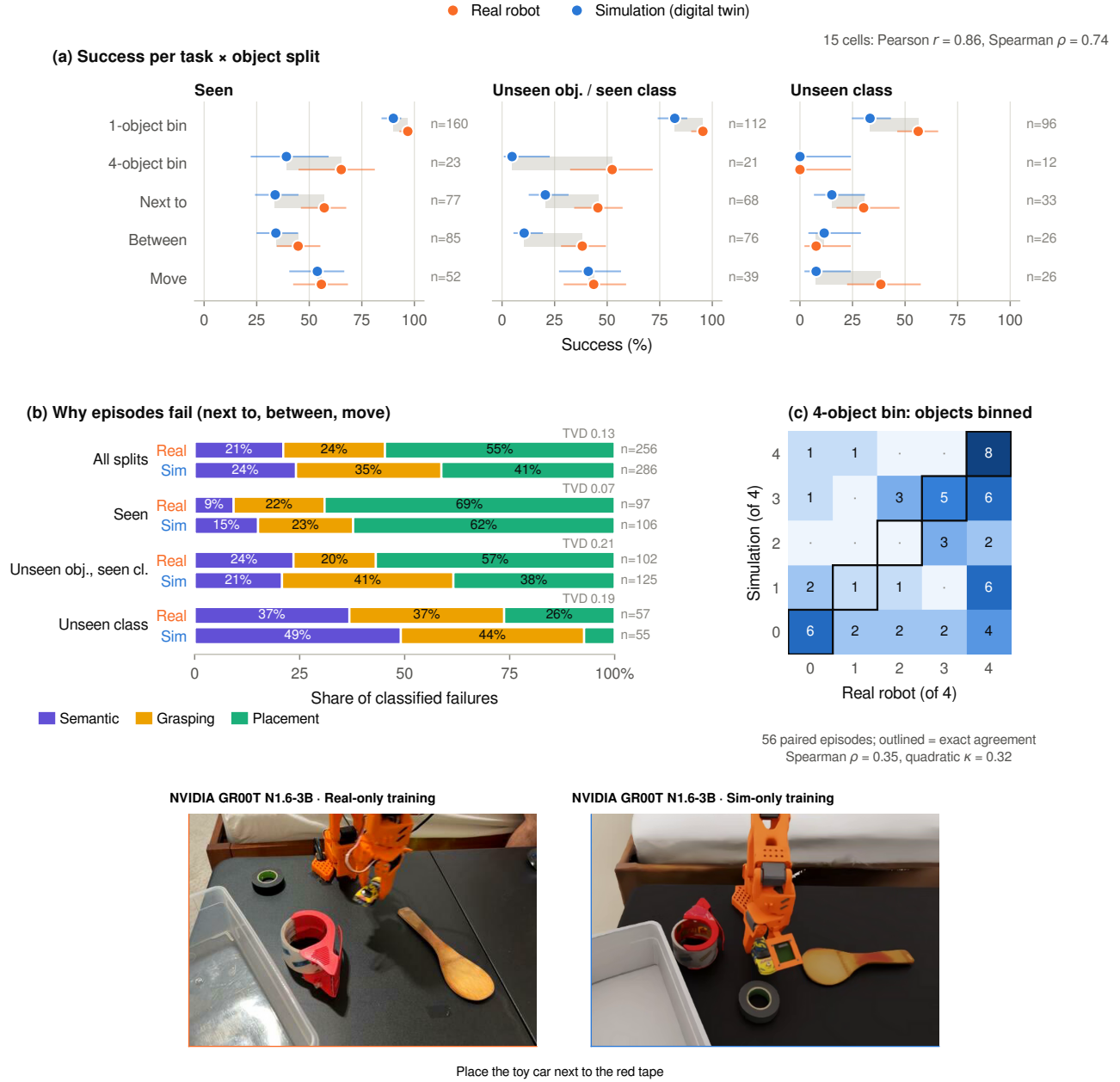


Figure 5: **Sim–real correspondence of independently trained GR00T N1.6 policies.** (a) Success of a real-only policy on the physical SO-101 and a sim-only policy in the digital twin, per task and object split (95% Wilson intervals). (b) Composition of hand-annotated failures (next to, between, move): both domains shift from placement to semantic and grasping failures as object novelty increases. (c) Objects binned in 4-object bin episodes, real vs. simulation (56 paired episodes; outlined cells are exact agreement). Below: overhead views of independently trained NVIDIA GR00T N1.6-3B policies executing the same instruction with the same objects (real trial 52, matched task row 466). Poses and frame times differ.

(GR00T N1.7 [72, 73], $\pi_{0.5}$ [48] and MolmoAct2 [7]) additionally receive the episode’s initial frames at each timestep.

4 Real-Only vs. Sim-Only Policy Correspondence

Before measuring sim2real transfer (Sec. 5), we ask a coarser question: does the digital twin reproduce the *difficulty structure* of the real benchmark? If a policy trained in simulation and evaluated in the twin finds the same tasks, splits and failure modes hard as a policy trained and evaluated in the real world, the twin is a useful proxy for real-world evaluation even before any weights cross domains.

Setup. We fine-tune two GR00T N1.6-3B policies [72, 74] independently: a *real-only* policy on real tele-operated demonstrations (`so101_bench_real`), evaluated on the physical SO-101, and a *sim-only* policy on demonstrations collected in the digital twin (a subset of `so101_bench_sim` containing 3D scanned objects from `so101_bench_real`, with 50% of the data volume of `so101_bench_real`), evaluated in simulation. We compare them on 906 episodes matched across domains by task, instruction, ordered object set and object split. Object poses are not identical across domains, so individual episodes are not exact replicas; the primary comparison is therefore at the level of the 15 task $\times$ split conditions and of failure composition, with episode-level agreement reported as a secondary measure. Real-world outcomes and failure types were labelled by hand; simulated failures were labelled the same way and both were mapped to three categories, *semantic* (wrong object or referent(s)), *grasping* (grasp, lift or release failed) and *placement* (correct object handled, imprecise/incorrect placement).

Difficulty is preserved across conditions. The sim-only policy succeeds on 45% (409/906) of episodes and the real-only policy on 61% (552/906), a gap of -15.8 pp (95% CI $[-19.9, -11.8]$). Despite this offset, the two systems order the 15 conditions similarly: Pearson $r = 0.86$ $[0.71, 0.91]$ and Spearman $\rho = 0.74$ $[0.52, 0.91]$ across cells (Fig. 5a). In both domains success is highest on the 1-object bin and seen objects and falls with task difficulty and object novelty, with the 4-object bin and the between task hardest on unseen classes.

Failure modes are preserved. The composition of failures on the spatial tasks follows the same trend in both domains (Fig. 5b). On seen objects most failures are placement errors (69% real, 62% sim; total variation distance 0.07); on unseen classes most are semantic or grasping errors (74% real, 93% sim). Across all splits the two failure mixes differ by a total variation distance of 0.13. On the 4-object bin, the real-only policy bins 67% of objects and the sim-only policy 47%; per-episode progress is weakly correlated across domains, likely due to the small size of the sim dataset for this experiments (Spearman $\rho = 0.35$; Fig. 5c).

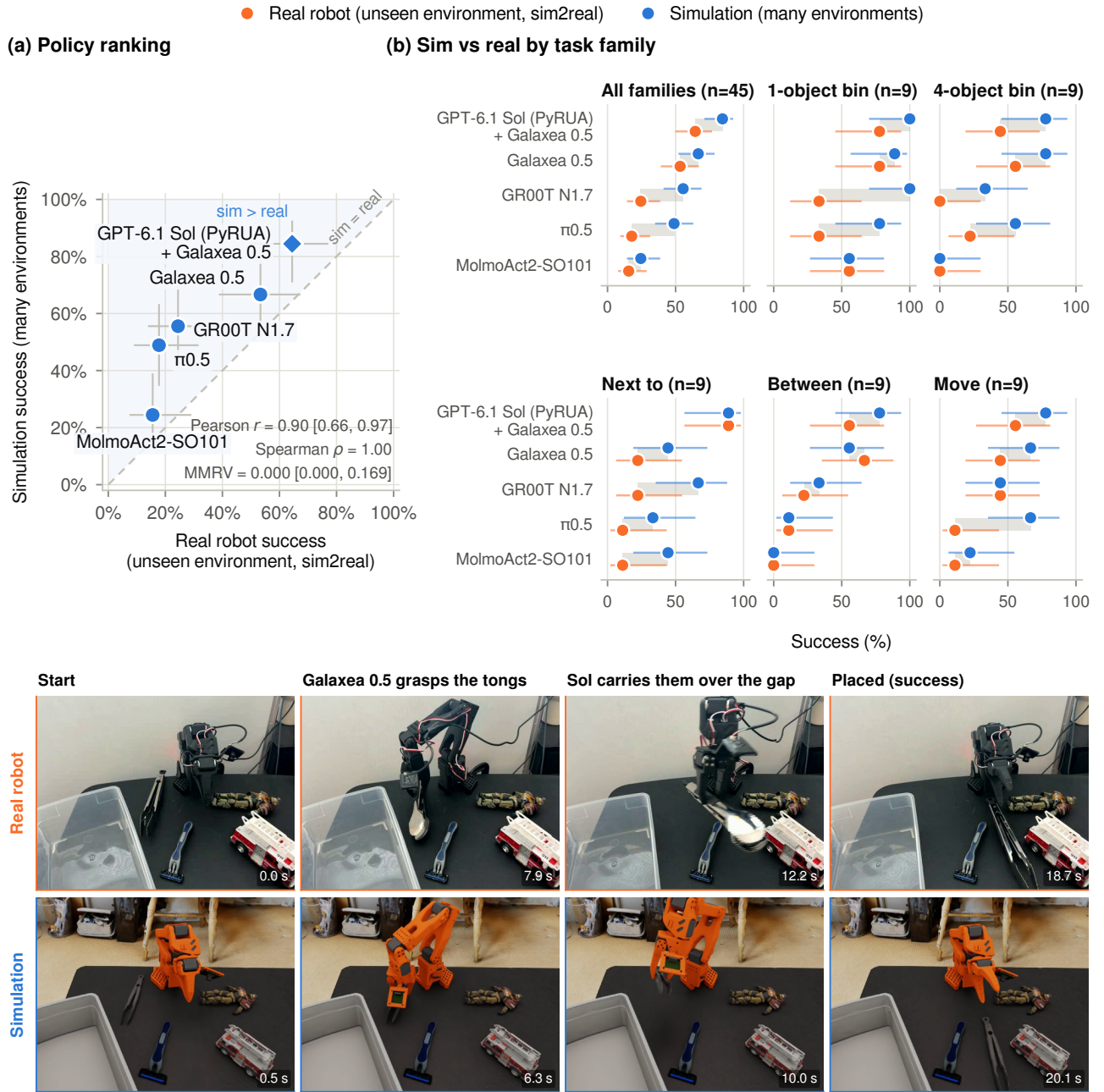
Episode-level agreement is moderate. Paired episodes have the same outcome 66% of the time (MCC 0.36), and where both failed the failure type matches in 47% of cases (Cohen’s $\kappa = 0.17$). This is expected given differing training dataset scales, initial poses and stochastic policies, and is why we emphasize condition-level correspondence. Full results are in Appendix A.

Overall, because the two policies differ in training data as well as in evaluation domain, this experiment cannot attribute the absolute gap to the simulator; the lower sim success reflects both policies and domains together. What it does show is that the twin reproduces which conditions are hard and why policies fail there. Section 5 isolates the simulator by running identical weights (with sim training only) in both domains.

5 Sim2Real with No Real Data

A simulated benchmark is only useful if it tells us which policies are better in the real world. We test this directly: five policies that never saw real data from our setup are evaluated, with identical weights, both in the digital twin and on the physical SO-101.

Setup. We evaluate four learned policies, Galaxea 0.5 [28], GR00T N1.7 [73], $\pi_{0.5}$ [48] and MolmoAct2-SO101 [7], and the strongest LLM agent, GPT-6.1 Sol (PyRUA) + Galaxea 0.5, with the same checkpoints and harness as the leaderboard (Sec. 6). The learned policies were fine-tuned only on simulated demonstrations; the agent pairs a frozen language model with the same sim-trained Galaxea 0.5 checkpoint, to which it hands



Trial 41: "Place the tongs between the razor and the toy fire truck" (unseen class). Robot time shown; the real episode took 9.2 min of wall-clock time.

Figure 6: **Sim2real**. (a) Policy-level success on the real robot (unseen environment) vs. simulation (many environments) on the same 45 layouts: the ranking of all five policies is preserved exactly (Spearman $\rho = 1.00$, MMRV = 0.000) and every policy scores higher in simulation. Diamond: the LLM agent; circles: learned policies fine-tuned only on simulated demonstrations. (b) Success in simulation and on the real robot per policy, overall (45 episodes) and per task family (9 episodes), with 95% Wilson intervals. Bottom: the agent on matched trial 41, a between episode with unseen-class objects that none of the four learned policies solved on the real robot. Galaxea 0.5 grasps the tongs, then the agent carries and places them, in both domains.

off grasping. The evaluation set has 45 episodes: 3 for each of the 15 task-family $\times$ split cells, using only the 40 benchmark objects we still own physically (18 seen, 11 unseen object, 11 unseen class), with every object used at least once. Each layout is reproduced on the real table by projecting the simulated object footprints into the overhead camera view, so both domains start from matched object positions and poses. The real

robot operates in an environment that does not appear in the training data, and real episodes are scored by the operator under the benchmark’s success and failure rules and the same time limits. As in simulation, the agent’s time limit counts control steps only: the arm holds still while the model computes. Median elapsed wall time to rollout stop is 197 s for the agent, compared with 38 s for Galaxea 0.5, 38 s for GR00T N1.7, 59 s for $\pi_{0.5}$ and 56 s for MolmoAct2-SO101 over the same 45 layouts, including successes and failures. These times come from timestamped control traces checked against the original recordings and include inference waits and within-rollout stalls, but exclude model/server startup, scene resets and subsequent operator scoring (Appendix B). Lighting and camera placement are fixed for the whole study. The four learned policies are evaluated in alternating blocks so that slow drift does not favour any one of them, and the agent in a later session in the same environment; before every episode the overhead view is compared with a reference frame (across all 225 real episodes the camera moved by at most 2.6 px and brightness changed by at most 7%). In simulation each layout is run once with the canonical appearance.

Simulation preserves the policy ranking. Success in simulation and on the real robot ranks the five policies identically (Fig. 6a): GPT-6.1 Sol (PyRUA) + Galaxea 0.5 84% (38/45) vs. 64% (29/45), Galaxea 0.5 67% (30/45) vs. 53% (24/45), GR00T N1.7 56% (25/45) vs. 24% (11/45), $\pi_{0.5}$ 49% (22/45) vs. 18% (8/45), and MolmoAct2-SO101 24% (11/45) vs. 16% (7/45). The Spearman correlation is 1.00 and the mean maximum rank violation (MMRV [18]) is 0.000 (95% CI [0.000, 0.169], resampling episodes within each policy); the Pearson correlation is 0.90 [0.66, 0.97]. The ranking spans two kinds of system, an LLM agent that plans and places through code and four end-to-end policies, so it does not depend on the policies sharing an architecture. With five policies these intervals remain wide, and we treat the result as evidence that simulation ranks policies correctly rather than as a precise calibration.

The agent’s lead transfers. On the real robot the agent succeeds on 29 of 45 episodes against 24 for Galaxea 0.5, the policy it calls for grasping: it solves 10 episodes that Galaxea 0.5 fails and fails 5 that Galaxea 0.5 solves (exact McNemar $p = 0.30$; in simulation 10 against 2, $p = 0.04$). Its advantage lies where it, rather than Galaxea 0.5, does the work. It succeeds on 8 of 9 real next-to episodes, against 2 for Galaxea 0.5, and on 7 of 15 unseen-class episodes, against 4. Eight real episodes were solved by the agent and by none of the four learned policies, including the between episode with unseen-class objects in Fig. 6 (bottom). None of the agent’s 16 real failures involved the wrong object: 7 were on the bin tasks (5 knocked the lighter real bin), 3 moved a non-target object, and the remaining 6 missed the placement criteria (for example, by setting the target on top of a referent).

Simulation is optimistic, by different amounts per policy. Every policy succeeds more often in simulation, by 9 to 31 percentage points (20 for the agent). Paired episodes agree on the outcome 56–91% of the time, and almost all disagreements are episodes that succeed in simulation and fail on the real robot (57, against 10 the other way). The gap is largest on the bin tasks for GR00T N1.7 and on the bin and move tasks for $\pi_{0.5}$ (Fig. 6b): for example, GR00T bins the single object in 9/9 simulated episodes but 3/9 real ones; each policy’s low real-world success rate on bin tasks is, however, largely explained by the far lesser mass of the real plastic bin (making it easier to displace). Many larger real objects were also scaled down to 0.7–0.9 times their real-world size in simulation. Overall, absolute success rates in simulation should be read as an upper bound on real-world performance, while comparisons between policies transfer. Per-family and per-split results are in Appendix B.

6 Leaderboard

We evaluate nine learned policies and two LLM agents on the full primary evaluation (Fig. 7).

Policies. The learned policies cover two families. *Vision-language-action models* (VLAs) map images and language directly to actions: GR00T N1.7 [73], $\pi_{0.5}$ [48], Galaxea 0.5 [28], and MolmoAct2 [7], which we fine-tune from two starting points, the general MolmoAct2 checkpoint (*base*) and its SO-100/101 post-trained variant (*SO101*). *World-action models* (WAMs) additionally model future observations: InternW0- Δ [27] and FLUX 3 Action [75] jointly model video and actions, ImageWAM [76] uses an image-editing backbone, and

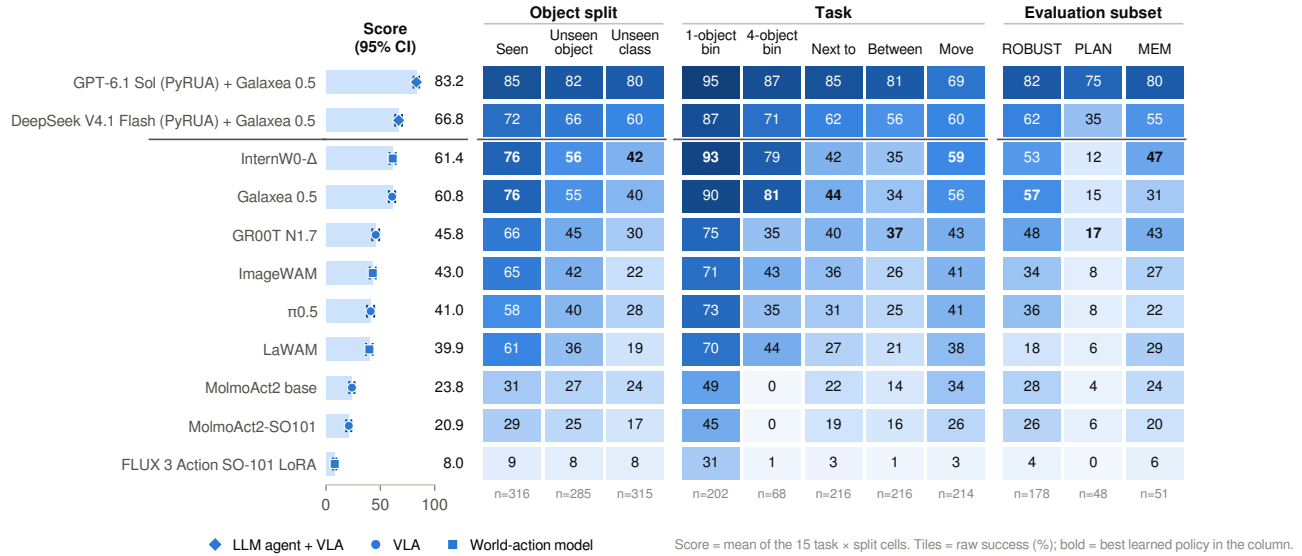


Figure 7: **SO-101 Bench leaderboard** (916 primary-evaluation episodes). Score is the mean success over the 15 task × object-split cells with a 95% bootstrap CI; tiles give raw success by object split, task and evaluation subset (ROBUST: held-out backgrounds and robot colours; PLAN: rotate to fit; MEM: remember occluded objects). Bold marks the best learned policy in each column. Diamonds: LLM agents; circles: VLAs; squares: WAMs.

LaWAM [77] predicts DINOv3 [78] latent states. All learned policies are fine-tuned on `so101_bench_sim` under the protocol of Sec. 3.7; FLUX 3 Action is fine-tuned with LoRA [79], following its released recipe, and the rest are fine-tuned in full or with their recommended frozen components (Appendix C). The two *LLM agents* pair a frozen language model (GPT-6.1 Sol [29] or DeepSeek V4.1 Flash-1017 [80]) with a code-as-policy harness [45], PyRUA-Lean [30], that hands grasping to Galaxea 0.5; we analyse them in Sec. 7.

Learned policies fall short of the benchmark. The best learned policies, InternW0-Δ (61.4, 95% CI [58.5, 64.1]) and Galaxea 0.5 (60.8 [57.9, 63.6]), are statistically tied, followed by GR00T N1.7 (45.8), ImageWAM (43.0), $\pi_{0.5}$ (41.0) and LaWAM (39.9). Both MolmoAct2 variants score below 25 and FLUX 3 Action scores 8.0. Starting MolmoAct2 from its SO-100/101 checkpoint does not help (20.9 vs. 23.8 for the base model, with overlapping intervals), so prior exposure to the embodiment (in its current form) is no substitute for the benchmark’s task and object coverage under this protocol.

Training scale appears to drive performance. Under the equal 36 hr budget, the number of fine-tuning samples each policy sees varies by a factor of 30 (Appendix C), and the amount of robot data in its pretraining varies widely as well. The strongest learned policies combine large robot pretraining corpora with at least a full epoch of fine-tuning (InternW0-Δ 1.2 epochs, Galaxea 0.5 1.6), whereas the three weakest saw well under one epoch: MolmoAct2 0.5–0.6 epochs and FLUX 3 Action 0.2; both appear under-trained. The MolmoAct2 training loss was still falling steeply when the budget ran out, and its published simulation recipe uses about five times as many samples [7]; FLUX 3 Action’s LoRA run used less than a tenth of the samples in its released full fine-tuning recipe [75]. These results therefore rank policies at a fixed compute budget, and the low scores of MolmoAct2 and FLUX 3 Action should not be read as their capability ceiling.

Spatial tasks and novel objects are the bottleneck. Grasping familiar objects is largely solved: the strongest learned policies bin a single object in 90–93% of episodes and place 90% of objects in 4-object bins. The spatial tasks are far harder, with next-to and between success of 34–44% for the same policies. Object novelty compounds this. On the spatial tasks, InternW0-Δ falls from 72% on seen objects to 18% on unseen classes for next to, and from 56% to 19% for between; Galaxea 0.5 falls from 71% to 19% and from 63% to 12%. Pooled

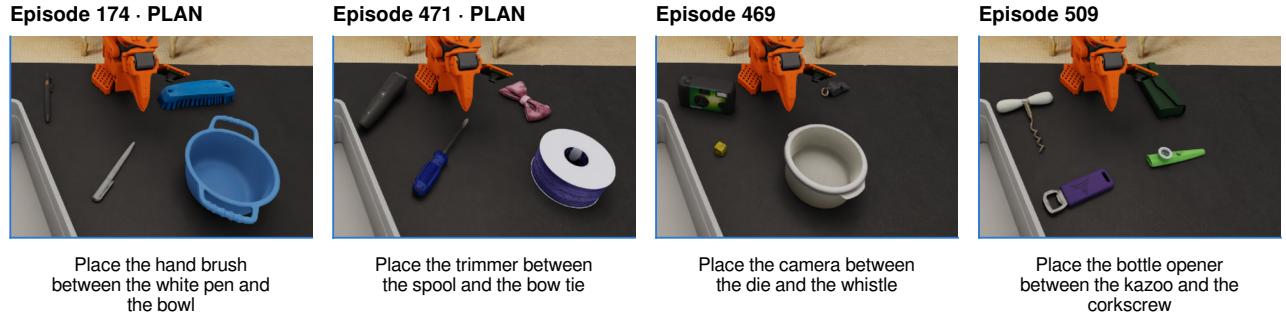


Figure 8: **Episodes solved only by GPT-6.1 Sol + PyRUA-Lean v9.6 + Galaxea 0.5.** Initial overhead views, cropped to the task workspace, with the exact instructions. All ten other leaderboard entries failed each episode. Episodes 174 and 471 belong to PLAN.

over tasks, the six strongest learned policies lose 30–43 points between seen objects and unseen classes; the three weakest lose less only because their seen-object success is already low.

Subsets isolate distinct weaknesses. PLAN is the clearest gap: no learned policy exceeds 17% when the grasped object must be rotated to fit (GR00T N1.7 is best), while the agent reaches 75%. On MEM, learned policies reach at most 47% (InternW0- Δ , which has native memory [27]). ROBUST separates policies by visual robustness rather than skill: Galaxea 0.5 and GR00T N1.7 keep their overall success under held-out backgrounds and robot colours, whereas LaWAM drops from 39% to 18% and ImageWAM from 43% to 34%.

LLM agents lead. Both agents outperform every learned policy, including the Galaxea 0.5 policy they call: GPT-6.1 Sol reaches 83.2 [80.7, 85.7] and DeepSeek V4.1 Flash-1017 66.8 [63.7, 69.9]. The agent’s advantage is largest where learned policies are weakest: Sol’s success varies by only five points across object splits (85/82/80%) and it leads on PLAN and MEM. Sol also outperforms Galaxea 0.5 on the real robot (64% vs. 53%, Sec. 5). Section 7 analyses where this advantage comes from.

7 LLM Agents: Harness and Model Ablations

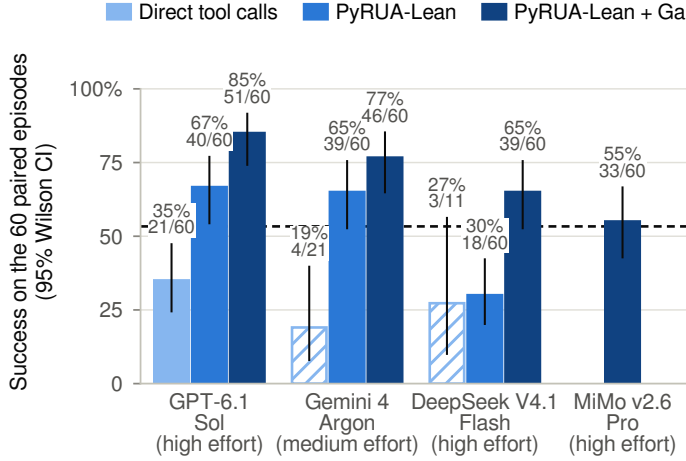
The LLM agents lead the leaderboard (Sec. 6). We ask what drives this: the harness that connects the language model to the robot, or the language model itself.

Harnesses. All harnesses give the model the same overhead and wrist images, the same task rules and no privileged simulator state. *Direct tool calls* uses the *inspect-robots* harness [81], letting the model command end-effector motions directly through tool calls. *PyRUA-Lean* [30] is a code-as-policy harness [45]: the model writes Python that uses open-vocabulary segmentation (via SAM3 [82]), 3D localization and inverse kinematics to move the arm. *PyRUA-Lean + Galaxea 0.5* adds the fine-tuned Galaxea 0.5 policy as a tool: the model localizes objects, plans, and places, while handing grasping to Galaxea 0.5; the agent inspects Galaxea 0.5’s results periodically, and can initiate recoveries or take over. The simulator is paused while the model computes, so time limits apply to simulated time only.

Paired episodes. Running agents on all 916 episodes is costly, so the ablations use 60 fixed primary-evaluation episodes: 30 *easy* episodes (six per task family) and 30 *hard* episodes (six per family with the fewest successes among the nine learned policies, including 11 PLAN episodes). Both sets were selected before any agent was run.

The harness matters more than the model. With GPT-6.1 Sol, success rises from 35% (21/60) with direct tool calls to 67% (40/60) with PyRUA-Lean and 85% (51/60) with the Galaxea 0.5 hand-off (Fig. 9a), against

(a) Harness × LLM



(b) Sol per task vs Galaxea 0.5 alone

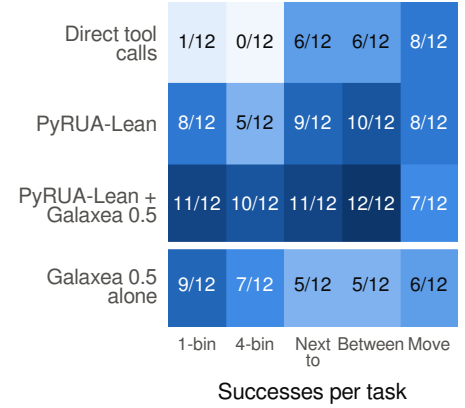


Figure 9: **Harness and LLM ablation** on the 60 paired episodes. (a) Success of each harness with three LLMs, and of the PyRUA-Lean v9.6 + Galaxea 0.5 harness with a fourth, MiMo v2.6 Pro (95% Wilson intervals); hatched bars were stopped before completing all 60 episodes. The dashed line marks Galaxea 0.5 alone. (b) Successes per task family for GPT-6.1 Sol with each harness and for Galaxea 0.5 alone.

Table 1: **Cost and token usage** of the PyRUA-Lean v9.6 + Galaxea 0.5 harness with each LLM, summed over the same 60 paired episodes. Output tokens include reasoning tokens (shown in parentheses). API cost is estimated from token counts at list prices (Gemini: gross Vertex rate).

LLM (effort)	Success	API cost	Output tokens (reasoning)	Input tokens (cached)
GPT-6.1 Sol (high)	85% (51/60)	\$14.68	0.32M (0.15M)	61.82M (96%)
Gemini 4 Argon (medium)	77% (46/60)	\$74.25	1.11M (0.86M)	42.30M (73%)
DeepSeek V4.1 Flash-1017 (high)	65% (39/60)	\$1.40	1.45M (1.23M)	87.33M (98%)
MiMo v2.6 Pro (high)	55% (33/60)	\$2.63	1.06M (0.82M)	75.33M (96%)
Galaxea 0.5 alone	53% (32/60)	\$0	–	–

53% (32/60) for Galaxea 0.5 alone. Direct tool calls have trouble grasping (1/12 single-object bins, 0/12 four-object bins) but handle the spatial tasks reasonably (Fig. 9b); writing perception and IK code unlocks grasping, and handing grasps to Galaxea 0.5 makes them reliable. The hand-off is also cheaper: Sol’s API cost falls from \$18.44 with PyRUA-Lean to \$14.68 with the hand-off over the 60 episodes. The same ordering holds for Gemini 4 Argon (65% to 77%) and DeepSeek V4.1 Flash-1017 (30% to 65%). The two policies are complementary: Galaxea 0.5 alone succeeds on 93% of easy episodes but only 13% of hard ones, whereas Sol with the hand-off reaches 93% and 77%. Move is the one family where the hand-off does not help (7/12, against 8/12 with PyRUA-Lean alone).

The model matters too. Holding the PyRUA-Lean + Galaxea 0.5 harness fixed, we compare four language models on the same 60 episodes (Fig. 9a, Table 1). GPT-6.1 Sol [29] reaches 85% (51/60), Gemini 4 Argon [83] 77% (46/60), DeepSeek V4.1 Flash-1017 [80] 65% (39/60) and MiMo v2.6 Pro [84] 55% (33/60); all four exceed Galaxea 0.5 alone. The models differ most on the spatial tasks (between: 4 to 12 of 12 episodes) and on the hard episodes, where they solve 23, 20, 14 and 11 of 30. The full benchmark confirms the ordering of the two models we ran on all 916 episodes (83.2 vs. 66.8). Cost and capability are only loosely related: DeepSeek V4.1 Flash-1017 costs \$1.40 for the 60 episodes, a tenth of Sol, while Argon costs the most (\$74.25) without matching Sol. Sol also generates the fewest tokens, 0.32M output tokens against 1.06–1.45M for the others (despite Gemini being at Medium effort).

8 Conclusion

SO-101 Bench makes evaluation of language-conditioned spatial manipulation on a low-cost arm both repeatable and predictive of the real robot. Its digital twin reproduces which conditions are hard and why policies fail there, and it ranks five policies trained without real data exactly as the physical SO-101 does; policies developed against the benchmark run out of the box in unseen real environments with unseen objects. The leaderboard shows where current methods stand: learned policies grasp familiar objects reliably but struggle with spatial placement, unseen object classes and rotate-to-fit planning, while an LLM agent that writes code and delegates grasping to a learned policy leads both in simulation and on the real robot, largely because of its harness.

Limitations. The sim2real study covers five policies and 45 layouts in one real environment, so its intervals are wide (MMRV 95% CI [0.000, 0.169]), and simulation overestimates absolute success by 9 to 31 percentage points; simulated success rates should be read as an upper bound and comparisons, not levels, as what transfers. The learned policies are compared at a fixed 36 h fine-tuning budget, which left some of them under-trained, so the leaderboard does not measure capability ceilings. The agents’ time limits count simulated or control time only: the simulator pauses, and the real arm holds still, while the language model computes, which hides their real-time cost. Finally, the benchmark covers rigid objects and a single arm. Extending the twin to deformable objects and bimanual setups, adding real-time constraints for agents, and using the twin’s ranking as a prior for cheaper real-robot evaluation are natural next steps.

A logical next step is to use repeated rollouts in the twin to refine control programs and accumulate validated, reusable skills across episodes, as in ASPIRE [85] and RoboRSI [86], while keeping the held-out evaluation fixed.

References

- [1] Hugging Face. SO-101: LeRobot Robot Documentation, 2026. URL <https://huggingface.co/docs/lerobot/so101>. Official release or documentation; accessed 26 October 2026.
- [2] R. Knight, P. Kooijmans, R. Cadene, S. Alibert, M. Aractingi, D. Aubakirova, A. Zouitine, R. Martino, S. Palma, C. Pascal, and T. Wolf. SO-ARM100: Standard Open SO-100 & SO-101 Arms, 2024. URL <https://github.com/TheRobotStudio/SO-ARM100>. GitHub repository; accessed 26 October 2026.
- [3] P. Selvaraj, L. Uttini, and V. Kuosmanen. ArmnetBench v0.1: Parallel Real-World Evaluation of Manipulation Policies on a Low-Cost Arm Farm, 2026. URL <https://arxiv.org/abs/2607.24481>.
- [4] K. Kawaharazuka, Y. Obinata, H. Ishida, J. Oh, T. Suzuki, S. Inoue, K. Yoneda, A. Iwata, and K. Okada. MEVION: Low-Cost Open-Source Data Collection System for Powerful and High-Speed Dual-Arm Manipulation, 2026. URL <https://arxiv.org/abs/2607.17970>. Accepted to IEEE Robotics and Automation Practice.
- [5] R. Cadene, S. Alibert, F. Capuano, M. Aractingi, A. Zouitine, P. Kooijmans, J. Choghari, M. Russi, C. Pascal, S. Palma, D. Aubakirova, M. Shukor, J. Moss, A. Soare, Q. Lhoest, Q. Gallouédec, and T. Wolf. LeRobot: An Open-Source Library for End-to-End Robot Learning. In *International Conference on Learning Representations (ICLR)*, 2026. URL <https://openreview.net/forum?id=CiZMMAFQR3>.
- [6] M. Shukor, D. Aubakirova, F. Capuano, P. Kooijmans, S. Palma, A. Zouitine, M. Aractingi, C. Pascal, M. Russi, A. Marafioti, S. Alibert, M. Cord, T. Wolf, and R. Cadene. SmolVLA: A Vision-Language-Action Model for Affordable and Efficient Robotics, 2025. URL <https://arxiv.org/abs/2506.01844>.
- [7] H. Fang, J. Duan, D. Clay, S. Wang, S. Liu, W. Huang, X. Fan, W.-C. Tsai, S. Chen, Y. R. Wang, S. Xing, J. Cho, J. S. Park, A. Eftekhari, P. Sushko, K. Farley, A. Wadhwa, C. Harrison, W. Han, Y.-C. Lee, E. VanderBilt, R. Hendrix, S. Ellawela, L. Ngoo, J. Chai, Z. Ren, A. Farhadi, D. Fox, and R. Krishna. MolmoAct2: Action Reasoning Models for Real-world Deployment, 2026. URL <https://arxiv.org/abs/2605.02881>.
- [8] Y. Yu and X. Qiu. Benchmarking Vision-Language-Action Models on SO-101: Failure and Recovery Analysis, 2026. URL <https://arxiv.org/abs/2606.08881>.
- [9] A. S. Huang, J. Zhang, S. Tang, and Y. Xiang. VLA-REPLICA: A Low-Cost, Reproducible Benchmark for Real-World Evaluation of Vision-Language-Action Models, 2026. URL <https://arxiv.org/abs/2605.20774>.
- [10] J. Tien-Smith, M. Pidden, and R. A. Potamias. Spa-Bench: A Real-Robot Benchmark for Spatial Grounding in Vision-Language-Action Models, 2026. URL <https://arxiv.org/abs/2610.12696>.
- [11] S. Arkhangelskiy. PhAIL: A Real-Robot VLA Benchmark and Distributional Methodology, 2026. URL <https://arxiv.org/abs/2605.29710>.
- [12] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone. LIBERO: Benchmarking Knowledge Transfer for Lifelong Robot Learning, 2023. URL <https://arxiv.org/abs/2306.03310>.
- [13] S. Nasiriany, S. Nasiriany, A. Maddukuri, and Y. Zhu. RoboCasa365: A Large-Scale Simulation Framework for Training and Benchmarking Generalist Robots, 2026. URL <https://arxiv.org/abs/2603.04356>. International Conference on Learning Representations (ICLR), 2026.
- [14] T. Chen, Z. Chen, B. Chen, Z. Cai, Y. Liu, Z. Li, Q. Liang, X. Lin, Y. Ge, Z. Gu, W. Deng, Y. Guo, T. Nian, X. Xie, Q. Chen, K. Su, T. Xu, G. Liu, M. Hu, H. ang Gao, K. Wang, Z. Liang, Y. Qin, X. Yang, P. Luo, and Y. Mu. RoboTwin 2.0: A Scalable Data Generator and Benchmark with Strong Domain Randomization for Robust Bimanual Robotic Manipulation, 2025. URL <https://arxiv.org/abs/2506.18088>.

- [15] A. Almuzairee and H. I. Christensen. Squint: Fast Visual Reinforcement Learning for Sim-to-Real Robotics, 2026. URL <https://arxiv.org/abs/2602.21203>. Accepted to IEEE Robotics and Automation Letters (RA-L), 2026.
- [16] K. Sendai, T. Matsushima, and Y. Iwasawa. MINERVA: How Small Can a Manipulation Policy Be and Still Solve LIBERO?, 2026. URL <https://arxiv.org/abs/2609.03715>.
- [17] Y. Wang, Z. Rao, J. Zhang, X. Li, Z. Wu, Y. Duan, M. Zheng, F. Wang, S. You, and T. Cai. IndustrialVLA-Bench: A Traceable Multi-Axis Evaluation of Open Robot Policy Models, 2026. URL <https://arxiv.org/abs/2609.25562>.
- [18] X. Li, K. Hsu, J. Gu, K. Pertsch, O. Mees, H. R. Walke, C. Fu, I. Lunawat, I. Sieh, S. Kirmani, S. Levine, J. Wu, C. Finn, H. Su, Q. Vuong, and T. Xiao. Evaluating real-world robot manipulation policies in simulation. In *Conference on Robot Learning (CoRL)*, 2024.
- [19] A. Jain, M. Zhang, K. Arora, W. Chen, M. Torne, M. Z. Irshad, S. Zakharov, Y. Wang, S. Levine, C. Finn, W.-C. Ma, D. Shah, A. Gupta, and K. Pertsch. PolaRiS: Scalable Real-to-Sim Evaluations for Generalist Robot Policies, 2025. URL <https://arxiv.org/abs/2512.16881>.
- [20] M. Sedlacek, P. Yefanov, G. Ponimatkin, J. Bardhan, S. Pilc, M. Fourmy, E. Kazakos, C. G. M. Snoek, J. Sivic, and V. Petrik. REALM: A Real-to-Sim Validated Benchmark for Generalization in Robotic Manipulation, 2025. URL <https://arxiv.org/abs/2512.19562>.
- [21] N. Ranawaka, J. Wong, W.-L. Pai, W.-T. Chu, T. Dai, M. Moghani, H. Yin, Y. Jiang, W. Durbano, B. Huynh, Y. Fang, D. Xu, R. Zhang, L. Fei-Fei, L. Fan, B. Wen, A. Mandlekar, and Y. Zhu. SimFoundry: Modular and Automated Scene Generation for Policy Learning and Evaluation, 2026. URL <https://arxiv.org/abs/2606.28276>.
- [22] S. Wang, H. Xu, Y. Hu, F. Lin, and Y. Gao. A Practical Recipe Towards Improving Sim-and-Real Correlation for VLA Evaluation, 2026. URL <https://arxiv.org/abs/2606.10366>.
- [23] A. Deshpande, M. Guru, R. Hendrix, S. Jauhri, A. Eftekhari, R. Tripathi, M. Argus, J. Salvador, H. Fang, M. Wallingford, W. Pumacay, Y. Kim, Q. Pfeifer, Y.-C. Lee, P. Wolters, O. Rayyan, M. Zhang, J. Duan, K. Farley, W. Han, E. Vanderbilt, D. Fox, A. Farhadi, G. Chalvatzaki, D. Shah, and R. Krishna. MolmoB0T: Large-Scale Simulation Enables Zero-Shot Manipulation, 2026. URL <https://arxiv.org/abs/2603.16861>.
- [24] Z. Yu, J. Zheng, W. Xie, J. Shi, C. Zhang, C. Bai, and X. Li. OASIS: From Simulation Data Collection to Real-World Humanoid Loco-Manipulation, 2026. URL <https://arxiv.org/abs/2606.08548>.
- [25] T. Jiang, X. Tan, S. Wheeler, L. Sun, T. W. Ayalew, and M. Walter. What Are We Actually Benchmarking in Robot Manipulation?, 2026. URL <https://arxiv.org/abs/2606.04233>.
- [26] NVIDIA. Isaac Lab: A GPU-Accelerated Simulation Framework for Multi-Modal Robot Learning. *arXiv preprint arXiv:2511.04831*, 2025. URL <https://arxiv.org/abs/2511.04831>.
- [27] X. Miao, Z. Li, B. Fang, K. Song, T. Wang, H. Zhang, Y. Wang, X. Li, Y. He, X. Wei, C. Gao, X. Yang, Y. Xu, K. Ren, W. Guo, J. Zhou, X. Wang, W. Zhao, N. Yang, Z. Cai, Y. Xue, H. Li, Z. He, Y. Zhou, R. Fu, J. Zhang, S. Cui, F. Huang, Y. Zhou, X. Gao, Y. Yao, Q. Yu, K. Li, M. Zhou, M. Huang, X. Li, W. Cui, B. Jiang, X. Zhu, J. Dong, H. Guo, T. Lu, M. Yu, B. Zhou, B. Zhao, T. Xue, W. Zhang, and C. Shen. InternW0- Δ : A World Action Model Bridging Predictive Dynamics and Actions with 20K+ Hours of Open Data, 2026. URL <https://arxiv.org/abs/2609.31394>.
- [28] Y. Liu, Z. Dong, B. Ye, T. Yuan, T. Jiang, A. Yang, S. Cao, H. Liu, Y. Sun, Z. Guo, X. Liu, K. Dong, C. Pan, C. Wu, T. Cheng, X. Ren, X. Zhang, J. Cui, Z. Zhao, H. Zhang, K. Xu, H. Yang, B. Zhang, J. Niu, S. Zhu, S. Zhang, and H. Zhao. G0.5: One Autoregressive Stream for Robot Reasoning and Action, 2026. URL <https://arxiv.org/abs/2608.11739>.

- [29] OpenAI. Introducing GPT-6.1 Sol, 2026. URL <https://openai.com/index/introducing-gpt-6-1-sol/>. Official release or documentation; accessed 26 October 2026.
- [30] R. Si, J. Bi, S. Yang, R. Ni, W. Huang, Q. Wang, S. Jiang, Y. Zhang, F. Ruan, K. Zhang, D. Wang, X. Li, H. Feng, Z. Dong, and D. Zhou. Fewer Tokens, Better Action: GPT-6 Astra Robot Agents with 14% Higher Success Rate but 65% Fewer Tokens, 2026. URL <https://arxiv.org/abs/2610.01939>.
- [31] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware, 2023. URL <https://arxiv.org/abs/2304.13705>.
- [32] A. Li. Nori A3: A Bimanual Mobile Manipulator at the Appliance Price Point, 2026. URL <https://arxiv.org/abs/2605.16537>.
- [33] LightwheelAI. LeIsaac, 2025. URL <https://github.com/LightwheelAI/leisaac>. GitHub repository: SO-101 teleoperation, data collection and policy training in Isaac Lab; accessed 26 October 2026.
- [34] A. Kadian, J. Truong, A. Gokaslan, A. Clegg, E. Wijmans, S. Lee, M. Savva, S. Chernova, and D. Batra. Sim2Real Predictivity: Does Evaluation in Simulation Predict Real-World Performance? *IEEE Robotics and Automation Letters*, 2020. URL <https://arxiv.org/abs/1912.06321>.
- [35] J. X. Yang, R. Dagli, A. Zook, H. Hadfield, A. Goyal, S. Birchfield, F. Ramos, and J. Tremblay. RoboLab: A High-Fidelity Simulation Benchmark for Analysis of Task Generalist Policies, 2026. URL <https://arxiv.org/abs/2604.09860>. Robotics: Science and Systems (RSS), 2026.
- [36] L. Ruan, J. Yang, S. Gao, F. Gao, K. Liang, G. Liu, L. Wu, L. Jin, G. Gu, B. Xie, C. Yan, Z. Yuan, K. Luo, E. Chen, S. Chen, Y. Ping, M. Guo, R. Sun, K. Zhang, A. Du, R. Wu, L. Hao, Z. Li, R. Gan, H. Wang, and Q. Wang. X2Real: an eXtensive simulation benchmark for real-world generalist policies, 2026. URL <https://arxiv.org/abs/2609.27449>.
- [37] C. Xiao, H. Zhan, S. Krishna, P. Meng, M. Z. Irshad, S. Zakharov, and D. Held. H2RBench: A Real-to-Sim Benchmark for Evaluating Human-to-Robot Transfer, 2026. URL <https://arxiv.org/abs/2609.24778>. Conference on Robot Learning (CoRL), 2026.
- [38] X. Wang, H. Hao, W. Hu, A. E. Li, D. Ma, K. Ramani, H. Moon, and Y.-D. Kwon. ReVeal: A Reconstruction-Aware Real-to-Sim Framework for VLA Policy Evaluation, 2026. URL <https://arxiv.org/abs/2609.23910>.
- [39] S. Verma, L. Cilio, V. Christodoulou, T. Fermelis, Q. T. Ng, R. Chung, and S. Liang. Measuring Asset and Scene Reconstruction Effects in Real-to-Sim Robot Evaluation, 2026. URL <https://arxiv.org/abs/2610.00731>.
- [40] Gemini Robotics Team, K. Choromanski, C. Devin, Y. Du, D. Dwibedi, R. Gao, A. Jindal, T. Kipf, S. Kirmani, I. Leal, F. Liu, A. Majumdar, A. Marmon, C. Parada, Y. Rubanova, D. Shah, V. Sindhwani, J. Tan, F. Xia, T. Xiao, S. Yang, W. Yu, and A. Zhou. Evaluating Gemini Robotics Policies in a Veo World Simulator, 2025. URL <https://arxiv.org/abs/2512.10675>.
- [41] J. Xu, X. Hu, M. Chen, P. Gao, Y. Ye, and Y. Shi. World Models Dream of Success: Diagnosing and Repairing Failure Insensitivity in Robot World Models, 2026. URL <https://arxiv.org/abs/2610.09134>.
- [42] H. Kim, T. Chen, J. Sun, L. W. Osterberg, Q. Chen, K. Wang, and M. Schwager. LEGS: Fine-Tuning Teleop-Free VLAs for Humanoid Loco-manipulation in an Embodied Gaussian Splatting World, 2026. URL <https://arxiv.org/abs/2606.01458>.
- [43] X. Ye, K. Jaganathan, G. Lim, G. Chhablani, M. Z. Irshad, and Z. Kira. Splat2Act: From Captured Scenes to Personalized VLA Manipulation, 2026. URL <https://arxiv.org/abs/2610.13945>.

- [44] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman, et al. Do As I Can, Not As I Say: Grounding Language in Robotic Affordances, 2022. URL <https://arxiv.org/abs/2204.01691>. Conference on Robot Learning (CoRL), 2022.
- [45] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng. Code as Policies: Language Model Programs for Embodied Control, 2023. URL <https://arxiv.org/abs/2209.07753>.
- [46] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei. VoxPoser: Composable 3D Value Maps for Robotic Manipulation with Language Models, 2023. URL <https://arxiv.org/abs/2307.05973>. Conference on Robot Learning (CoRL), 2023.
- [47] L. X. Shi, B. Ichter, M. Equi, L. Ke, K. Pertsch, Q. Vuong, J. Tanner, A. Walling, H. Wang, N. Fusai, A. Li-Bell, D. Driess, L. Groom, S. Levine, and C. Finn. Hi Robot: Open-Ended Instruction Following with Hierarchical Vision-Language-Action Models, 2025. URL <https://arxiv.org/abs/2502.19417>. International Conference on Machine Learning (ICML), 2025.
- [48] Physical Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, M. Y. Galliker, D. Ghosh, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, D. LeBlanc, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, A. Z. Ren, L. X. Shi, L. Smith, J. T. Springenberg, K. Stachowicz, J. Tanner, Q. Vuong, H. Walke, A. Walling, H. Wang, L. Yu, and U. Zhilinsky. $\pi_{0.5}$: a Vision-Language-Action Model with Open-World Generalization, 2025. URL <https://arxiv.org/abs/2504.16054>.
- [49] L. Fu, J. Yu, K. El-Refai, E. Kou, H. Xue, H. Huang, W. Xiao, G. Wang, D. Niu, F.-F. Li, G. Shi, J. Wu, S. Sastry, Y. Zhu, K. Goldberg, and L. Fan. CaP-X: A Framework for Benchmarking and Improving Coding Agents for Robot Manipulation, 2026. URL <https://arxiv.org/abs/2603.22435>.
- [50] Y. Lyu, X. Jiang, W. Li, S. Lu, and G. Zhang. CodeActionBench: Evaluating Agentic Code-as-Policy for Embodied Manipulation, 2026. URL <https://arxiv.org/abs/2609.33807>.
- [51] Z. Chu, E. T. Barr, C. Le Goues, P. O’Hearn, M. Harman, F. Sarro, and H. Ye. OpenRUA: Robot-Use Agents Are Zero-Shot Visuomotor Policies, 2026. URL <https://arxiv.org/abs/2610.02459>.
- [52] W. Zhang, K. Wang, Y. Ouyang, X. Huang, L. Li, K. Su, W. Jin, W. Chai, H. Liang, Z. Dou, Y. Chen, and T. Chen. An Unexpected Robot Policy: Early Evaluations of GPT-6 Astra on RoboDojo and Beyond, 2026. URL <https://arxiv.org/abs/2609.24170>.
- [53] T. Chen, Y. Chen, Z. Li, J. Tang, K. Su, H. Lu, W. Wan, B. Chen, S. Liu, H. Yan, et al. RoboDojo: A Unified Sim-and-Real Benchmark for Comprehensive Evaluation of Generalist Robot Manipulation Policies, 2026. URL <https://arxiv.org/abs/2607.04434>.
- [54] F. Liu, Y. Shen, A. Cheng, W. Mi, C. Tang, C. Liu, Y. Xiang, T. Li, Y.-L. Li, and Y. Tang. RoboICL: Embodied In-Context Learning with GPT-6 Astra, 2026. URL <https://arxiv.org/abs/2609.34261>.
- [55] Y. Zhang, H. Zhang, F. Gao, X. Li, Z. Liu, Y. Nie, C. Zhu, J. Qiu, Y. Yan, J. Liu, W. Tang, J. Rao, Z. Fang, C. Wei, Y. Wang, W. Ding, and C. Yu. Harness VLA: Steering Frozen VLAs into Reliable Manipulation Primitives via Memory-Guided Agents, 2026. URL <https://arxiv.org/abs/2607.08448>.
- [56] S. Chen, H. Hadfield, A. Zook, M. A. Uy, C. H. Song, E. Coumans, X. Yang, F. Ladhak, Q. Qu, S. Birchfield, J. Tremblay, and V. Blukis. VoLo: A Physical Orchestrator for Open-Vocabulary Long-Horizon Manipulation, 2026. URL <https://arxiv.org/abs/2606.07723>.
- [57] S. Hansen and P. Xu. Introducing Gemini Robotics ER 2, 2026. URL <https://blog.google/innovation-and-ai/models-and-research/google-deepmind/gemini-robotics-er-2/>. Google DeepMind blog post, 30 July 2026; accessed 26 October 2026.
- [58] J. Hu, M. Shridhar, C. Lu, D. Shah, H.-T. L. Chiang, J. Tan, and A. Xie. What Matters in Orchestrating Robot Policies: A Systematic Study of Hierarchical VLA Agents, 2026. URL <https://arxiv.org/abs/2606.10267>.

- [59] Z. Li, Y. Zhang, W. Li, H. Huang, C. Wang, S. Deng, Y. Wei, T. Zhang, B. Xie, B. Zhou, Y. Chang, K. Zhou, Y.-C. Chen, J. Cheng, and Y. Li. Robo-Harness K1: Harnessing Robot-Use Agents via Perception Augmentation, 2026. URL <https://arxiv.org/abs/2609.29389>.
- [60] J. Starace. Scaffold Effects on GAIA: A Controlled Comparison, 2026. URL <https://arxiv.org/abs/2606.08529>.
- [61] SenseNova and Joint Research Team, Z. Cai, Z. Cao, T. Chen, H. C. Chua, L. Dai, H. Deng, H. Deng, H. Diao, C. Ding, et al. Has GPT-6-Astra Achieved Embodied Intelligence?, 2026. URL <https://arxiv.org/abs/2610.22471>.
- [62] C. Leet, A. Menon, S. Machcha, S. Koenig, T. K. Chan, and J. Chooi. The Bitter Lesson Comes to Robotics, 2026. URL <https://arxiv.org/abs/2610.25287>.
- [63] B. Zhang, J. Li, J. Shen, Y. Zhang, Y. Cai, L. Liu, H. Ji, Y. Chen, J. Dai, J. Ji, and Y. Yang. VLA-Arena: An Open-Source Framework for Benchmarking Vision-Language-Action Models, 2025. URL <https://arxiv.org/abs/2512.22539>. International Conference on Machine Learning (ICML), 2026.
- [64] S. Fei, S. Wang, J. Shi, Z. Dai, J. Cai, P. Qian, L. Ji, X. He, S. Zhang, Z. Fei, J. Fu, J. Gong, and X. Qiu. LIBERO-Plus: In-depth Robustness Analysis of Vision-Language-Action Models, 2025. URL <https://arxiv.org/abs/2510.13626>.
- [65] X. Zhou, Y. Xu, G. Tie, Y. Chen, G. Zhang, D. Chu, P. Zhou, and L. Sun. LIBERO-PRO: Towards Robust and Fair Evaluation of Vision-Language-Action Models Beyond Memorization, 2025. URL <https://arxiv.org/abs/2510.03827>.
- [66] G. Wang, C. Zhang, Q. Liu, J. Zhang, J. Cai, J. Liu, and X. Liu. LIBERO-X: Robustness Litmus for Vision-Language-Action Models, 2026. URL <https://arxiv.org/abs/2602.06556>.
- [67] Y. Dai, H. Fu, J. Lee, Y. Liu, H. Zhang, J. Yang, C. Finn, N. Fazeli, and J. Chai. RoboMME: Benchmarking and Understanding Memory for Robotic Generalist Policies, 2026. URL <https://arxiv.org/abs/2603.04639>. International Conference on Machine Learning (ICML), 2026.
- [68] P. Atreya, K. Pertsch, T. Lee, M. J. Kim, A. Jain, A. Kuramshin, C. Neary, E. S. Hu, K. Arora, K. Ellis, L. Macesanu, M. Leonard, M. Cho, O. Aslan, S. Dass, T. Wang, X. Yuan, A. Gupta, D. Jayaraman, G. Berseth, K. Daniilidis, R. Martín-Martín, Y. Lee, P. Liang, C. Finn, and S. Levine. RoboArena: Distributed Real-World Evaluation of Generalist Robot Policies. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pages 336–364. PMLR, 2025. URL <https://proceedings.mlr.press/v305/atreya25a.html>.
- [69] Z. Zhou, P. Atreya, Y. L. Tan, K. Pertsch, and S. Levine. AutoEval: Autonomous Evaluation of Generalist Robot Manipulation Policies in the Real World. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pages 1997–2017. PMLR, 2025. URL <https://proceedings.mlr.press/v305/zhou25a.html>.
- [70] A. Badithela, D. Snyder, T. Yin, J. Sceats, G. Pappas, N. Matni, M. Itkina, H. Nishimura, and A. Majumdar. AnyRank: Reliable Anytime Policy Ranking via Sequential Testing, 2026. URL <https://arxiv.org/abs/2610.14603>. Conference on Robot Learning (CoRL), 2026.
- [71] 3D AI Studio. 3D AI Studio: Image-to-3D Tools, 2026. URL <https://www.3daistudio.com/>. Online service; accessed 26 October 2026.
- [72] NVIDIA, J. Bjorck, F. Castañeda, N. Cherniadev, et al. GR00T N1: An Open Foundation Model for Generalist Humanoid Robots, 2025. URL <https://arxiv.org/abs/2503.14734>.
- [73] NVIDIA. Isaac GR00T N1.7: Model and Code Release, 2026. URL <https://github.com/NVIDIA/Isaac-GR00T/releases/tag/n1.7-release>. Official release or documentation; accessed 26 October 2026.

- [74] NVIDIA. GR00T N1.6: An Improved Open Foundation Model for Generalist Humanoid Robots, 2025. URL https://research.nvidia.com/labs/gear/gr00t-n1_6/. Official release or documentation; accessed 26 October 2026.
- [75] Black Forest Labs. FLUX 3 Action, 2026. URL <https://bfl.ai/models/flux-3-action>. Official release or documentation; accessed 26 October 2026.
- [76] Y. Zhang, W. Zhang, Z. Qi, H. Zhang, H. Lin, J. Zhang, Y. Mu, X. Yang, W. Zeng, and X. Jin. ImageWAM: Do World Action Models Really Need Video Generation, or Just Image Editing?, 2026. URL <https://arxiv.org/abs/2606.19531>.
- [77] J. Chen, K. Wang, K. Chen, S. Chen, F. Gao, W. Tang, Z. Li, W. Liu, Z. Yao, B. Li, Y. Xu, and C. Yu. LaWAM: Latent World Action Models for Efficient Dynamics-Aware Robot Policies, 2026. URL <https://arxiv.org/abs/2606.15768>.
- [78] O. Siméoni, H. V. Vo, M. Seitzer, F. Baldassarre, M. Oquab, C. Jose, V. Khalidov, M. Szafraniec, S. Yi, M. Ramamonjisoa, F. Massa, D. Haziza, L. Wehrstedt, J. Wang, T. Darcet, T. Moutakanni, L. Sentana, C. Roberts, A. Vedaldi, J. Tolan, J. Brandt, C. Couprie, J. Mairal, H. Jégou, P. Labatut, and P. Bojanowski. DINOv3, 2025. URL <https://arxiv.org/abs/2508.10104>.
- [79] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. LoRA: Low-Rank Adaptation of Large Language Models, 2021. URL <https://arxiv.org/abs/2106.09685>.
- [80] DeepSeek. DeepSeek-V4.1-Flash-1017, 2026. URL <https://huggingface.co/deepseek-ai/DeepSeek-V4.1-Flash-1017>. Official model card; accessed 26 October 2026.
- [81] Robocurve. Inspect Robots: The Open-Source Evaluation Framework for Physical AI, 2026. URL <https://github.com/robocurve/inspect-robots>.
- [82] N. Carion, L. Gustafson, Y.-T. Hu, S. Debnath, R. Hu, D. Suris, C. Ryali, K. V. Alwala, H. Khedr, A. Huang, J. Lei, T. Ma, B. Guo, A. Kalla, M. Marks, J. Greer, M. Wang, P. Sun, R. Rädle, T. Afouras, E. Mavroudi, K. Xu, T.-H. Wu, Y. Zhou, L. Momeni, R. Hazra, S. Ding, S. Vaze, F. Porcher, F. Li, S. Li, A. Kamath, H. K. Cheng, P. Dollár, N. Ravi, K. Saenko, P. Zhang, and C. Feichtenhofer. SAM 3: Segment Anything with Concepts, 2025. URL <https://arxiv.org/abs/2511.16719>.
- [83] Google DeepMind. Gemini 4 Argon: Model Card, 2026. URL <https://deepmind.google/models/model-cards/gemini-4-argon/>. Official release or documentation; accessed 26 October 2026.
- [84] Xiaomi. Introducing MiMo-V2.6 Series, 2026. URL <https://mimo.xiaomi.com/mimo-v2-6>. Official release or documentation; accessed 26 October 2026.
- [85] R. Lu, Y. Wu, E. Kou, H. Lin, L. Fu, W. Xiao, A. Mandlekar, Y. Xu, G. Shi, K. Goldberg, A. Chen, M. Chowdhury, J. Kautz, Y. Zhu, L. J. Fan, and G. Wang. ASPIRE: Agentic /Skills Discovery for Robotics, 2026. URL <https://arxiv.org/abs/2607.00272v2>.
- [86] Z. Wen, Y. Chen, Y. Cao, W. Chen, Y. Zou, W. Yu, F. Kuang, H. Xue, J. Lv, C. Wen, and C. Lu. RoboRSI: Stable, Efficient, and Reusable Robot Self-Evolution in Complex Real-World Environments, 2026. URL <https://arxiv.org/abs/2610.12424v1>.

Table 2: **GR00T sim–real correspondence.** A real-only GR00T N1.6-3B policy evaluated on the physical SO-101 versus an independently trained sim-only GR00T N1.6-3B policy evaluated in the digital twin, on 906 episodes matched on task, instruction, ordered objects and object split. Δ is sim minus real success in percentage points with a 95% bootstrap CI that resamples matched semantic configurations within task $\times$ split cells; agree and MCC compare paired episode outcomes.

	n	Sim	Real	Δ (pp) [95% CI]	Agree	MCC
1-object bin	368	73% (268/368)	86% (316/368)	−13.0 [−20.1, −6.5]	81%	0.47
4-object bin	56	18% (10/56)	46% (26/56)	−28.6 [−42.9, −14.3]	64%	0.31
Next to	178	25% (45/178)	48% (85/178)	−22.5 [−32.0, −12.4]	49%	−0.04
Between	187	21% (40/187)	37% (69/187)	−15.5 [−24.6, −6.4]	60%	0.06
Move	117	39% (46/117)	48% (56/117)	−8.5 [−20.5, +3.4]	57%	0.14
Seen	397	59% (236/397)	71% (281/397)	−11.3 [−17.4, −5.5]	67%	0.29
Unseen object, seen class	316	41% (131/316)	62% (195/316)	−20.3 [−27.8, −13.2]	62%	0.29
Unseen class	193	22% (42/193)	39% (76/193)	−17.6 [−28.0, −8.3]	72%	0.40
All	906	45% (409/906)	61% (552/906)	−15.8 [−19.9, −11.8]	66%	0.36
Condition level (15 task $\times$ split cells): Pearson $r = 0.86$ [0.71, 0.91], Spearman $\rho = 0.74$ [0.52, 0.91]						

Table 3: **Failure-type correspondence (GR00T, next-to / between / move).** Every failed episode was annotated by hand in each domain and assigned to the first matching type in the order semantic > grasping > placement. Shares are of classified failures; TVD is the total variation distance between the real and sim mixes (0 = identical, 1 = disjoint).

	Real				Sim				TVD
	n	Sem.	Grasp	Place	n	Sem.	Grasp	Place	
All	256	21%	24%	55%	286	24%	35%	41%	0.13
Seen	97	9%	22%	69%	106	15%	23%	62%	0.07
Unseen obj. / seen class	102	24%	20%	57%	125	21%	41%	38%	0.21
Unseen class	57	37%	37%	26%	55	49%	44%	7%	0.19
Episodes where both domains failed: 144; same type in 47%, Cohen’s $\kappa = 0.17$									

A GR00T Sim–Real Correspondence: Additional Results

Table 4: **GR00T sim–real correspondence, all 15 task×split cells.** Each entry is simulation / real success.

Task	Seen	Unseen object, seen class	Unseen class
1-object bin	90% (144/160) / 97% (155/160)	82% (92/112) / 96% (107/112)	33% (32/96) / 56% (54/96)
4-object bin	39% (9/23) / 65% (15/23)	5% (1/21) / 52% (11/21)	0% (0/12) / 0% (0/12)
Next to	34% (26/77) / 57% (44/77)	21% (14/68) / 46% (31/68)	15% (5/33) / 30% (10/33)
Between	34% (29/85) / 45% (38/85)	11% (8/76) / 38% (29/76)	12% (3/26) / 8% (2/26)
Move	54% (28/52) / 56% (29/52)	41% (16/39) / 44% (17/39)	8% (2/26) / 38% (10/26)

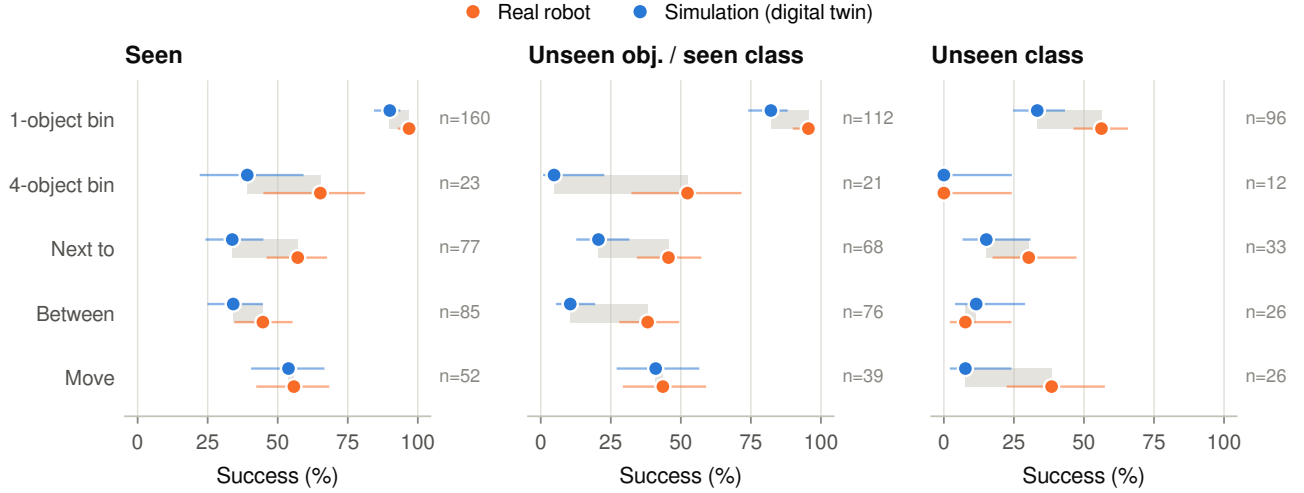


Figure 10: Real vs. simulation success in all 15 task × object-split cells (95% Wilson intervals).

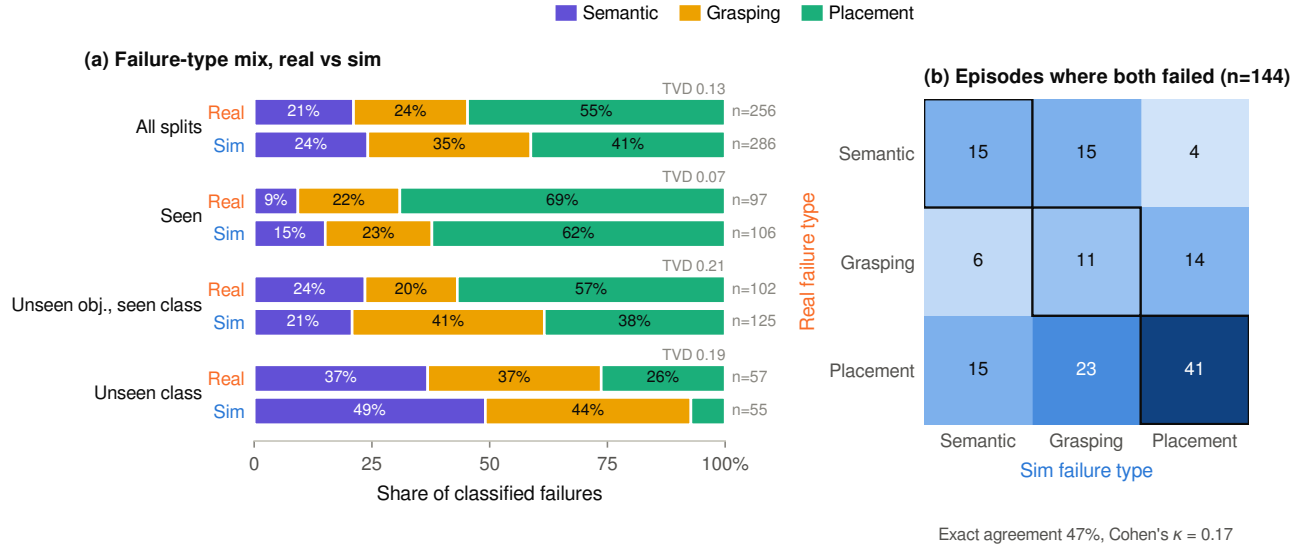


Figure 11: Failure-type mix per split and real × sim failure-type confusion on episodes where both domains failed.

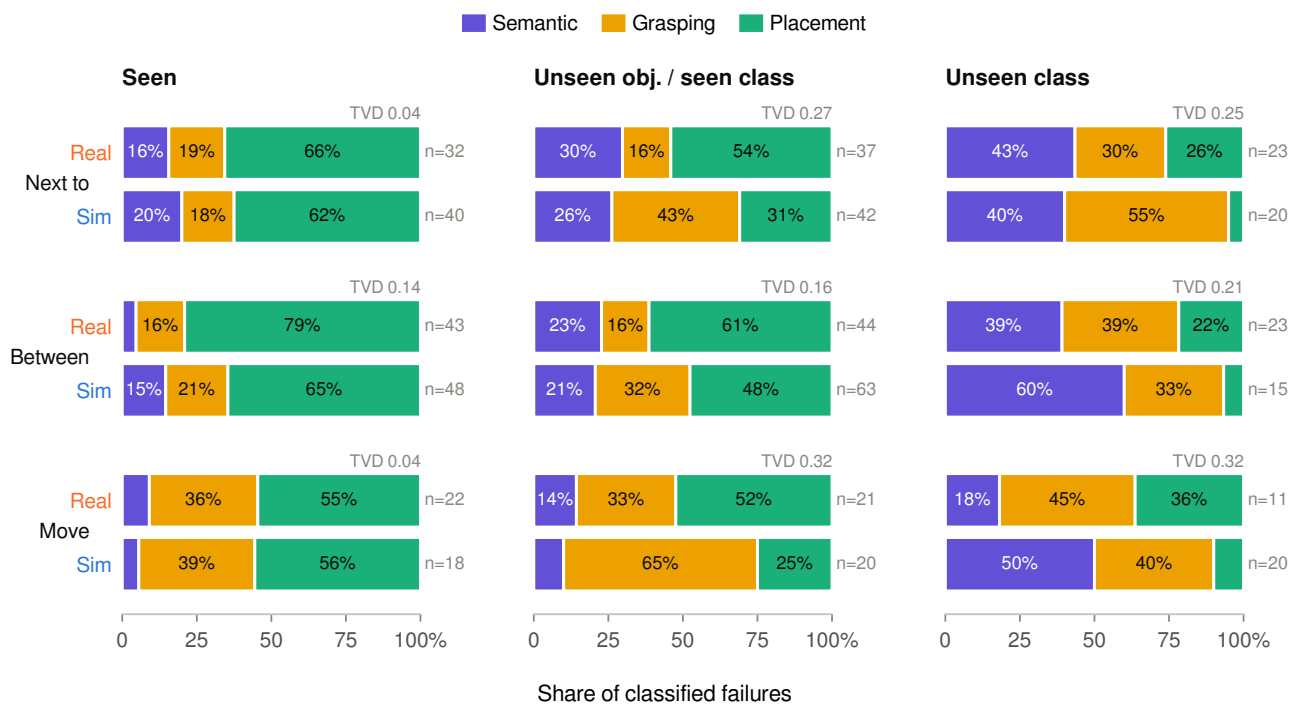


Figure 12: Failure-type mix per task and object split.

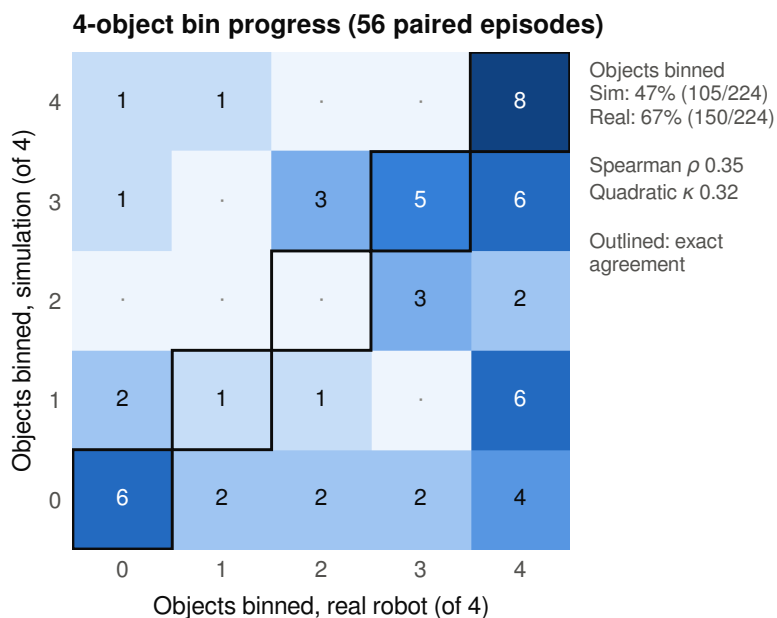


Figure 13: Objects binned in 4-object bin episodes, real vs. simulation (56 paired episodes).

B Sim2Real: Additional Results

The agent on the real robot. GPT-6.1 Sol (PyRUA) + Galaxea 0.5 runs on the real robot with the same harness (PyRUA-Lean v9.6), reasoning effort (high) and Galaxea 0.5 checkpoint as in simulation, with the real overhead and wrist cameras in place of the simulated ones. Table 5 compares elapsed real-robot rollout time for all five systems on the same 45 layouts. We use the last monotonic control timestamp relative to episode start for every policy, checked against its original recording. Three older recordings omit elapsed gaps present in the control traces, so compressed video duration is not substituted for elapsed time. These are deployed-system measurements of time to rollout stop, not time to success; early failures can therefore shorten a rollout.

Table 5: **Real-robot elapsed time in Sim2Real.** Forty-five matched layouts per policy, including both successes and failures. Elapsed wall time runs from episode start to the last logged control command and includes policy/API waits and any within-rollout stalls; server/model startup, scene resets and subsequent operator scoring are excluded. Times measure rollout stopping, not time to successful completion.

Policy	n	Median (s)	P95 (s)
GPT-6.1 Sol + PyRUA-Lean + Galaxea 0.5	45	196.6	391.8
Galaxea 0.5	45	37.5	116.5
GR00T N1.7	45	37.9	95.8
$\pi_{0.5}$	45	59.1	153.3
MolmoAct2-SO101	45	56.0	138.0

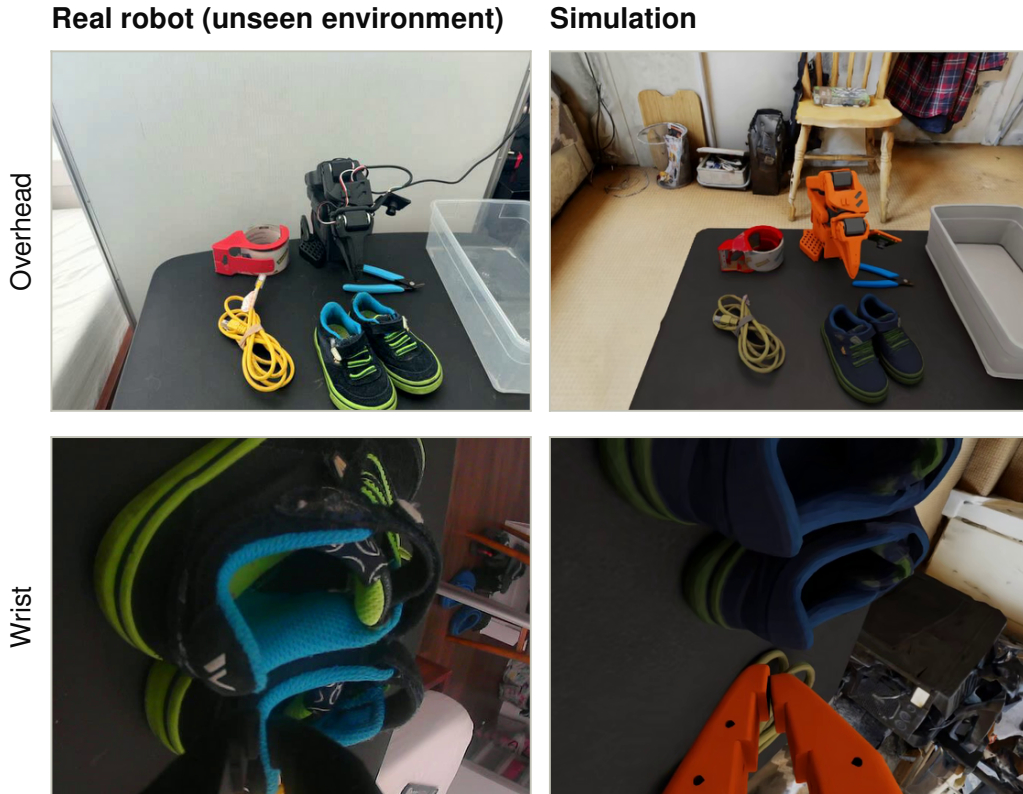


Figure 14: **One sim2real episode, real vs. simulation.** Initial overhead (top) and wrist (bottom) views of the same 4-object layout on the real robot and in the digital twin.

Table 6: **Sim2real evaluation with identical weights.** Each policy was evaluated on the same 45 layouts (5 task families $\times$ 3 object splits $\times$ 3) in simulation and on the physical SO-101. Agreement is the fraction of paired episodes with the same binary outcome. Ranking metrics are computed across the five policies; CIs resample episodes within each policy (10,000 replicates). MMRV [18] is the mean maximum rank violation (0 = sim preserves the real ranking).

Policy	Simulation	Real robot	Δ (pp)	Agreement
GPT-6.1 Sol (PyRUA) + Galaxea 0.5	84% (38/45)	64% (29/45)	+20.0	76% (34/45)
Galaxea 0.5	67% (30/45)	53% (24/45)	+13.3	64% (29/45)
GR00T N1.7	56% (25/45)	24% (11/45)	+31.1	56% (25/45)
$\pi_{0.5}$	49% (22/45)	18% (8/45)	+31.1	64% (29/45)
MolmoAct2-SO101	24% (11/45)	16% (7/45)	+8.9	91% (41/45)
	Pearson r [95% CI]	Spearman ρ	MMRV [95% CI]	
Binary success	0.90 [0.66, 0.97]	1.00	0.000 [0.000, 0.169]	
4-bin as progress	0.87 [0.62, 0.96]	0.90	0.002 [0.000, 0.187]	

Table 7: **Sim2real by task family and object split.** Successes in simulation / on the real robot; 9 episodes per family and 15 per split for each policy. GPT-6.1 Sol[†]: GPT-6.1 Sol (PyRUA) + Galaxea 0.5.

	GPT-6.1 Sol [†]	Galaxea 0.5	GR00T N1.7	$\pi_{0.5}$	MolmoAct2-SO101
1-object bin	9 / 7	8 / 7	9 / 3	7 / 3	5 / 5
4-object bin	7 / 4	7 / 5	3 / 0	5 / 2	0 / 0
Next to	8 / 8	4 / 2	6 / 2	3 / 1	4 / 1
Between	7 / 5	5 / 6	3 / 2	1 / 1	0 / 0
Move	7 / 5	6 / 4	4 / 4	6 / 1	2 / 1
Seen	15 / 12	14 / 11	10 / 5	12 / 5	5 / 3
Unseen object, seen class	13 / 10	10 / 9	10 / 4	7 / 2	5 / 3
Unseen class	10 / 7	6 / 4	5 / 2	3 / 1	1 / 1

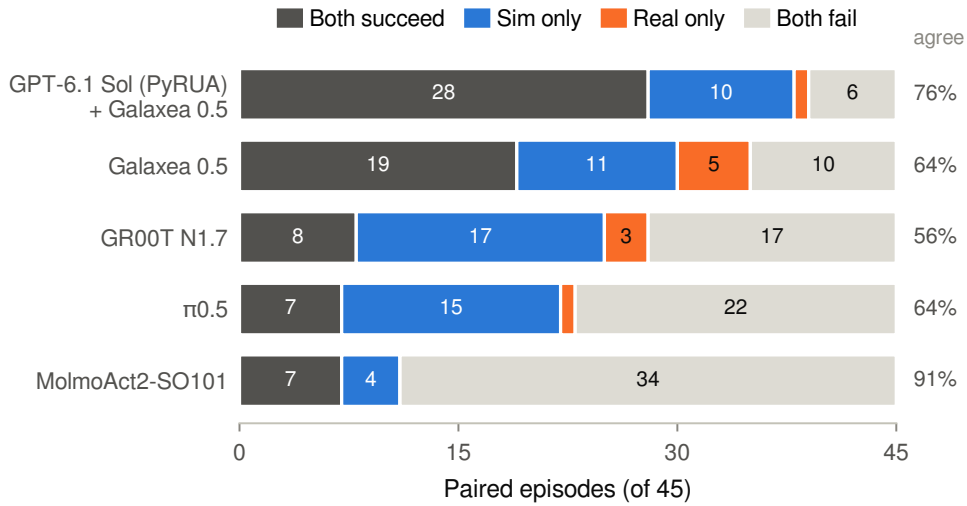


Figure 15: Paired episode outcomes per policy.

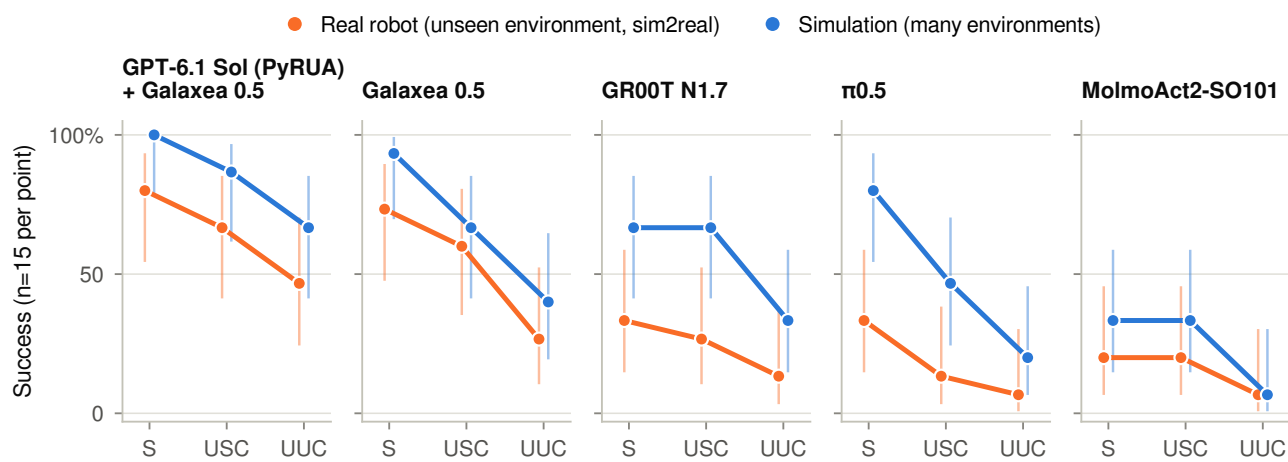


Figure 16: Seen $\rightarrow$ unseen gradient in each domain, per policy.

C Leaderboard: Additional Results

Measured inference latency. Table 8 summarizes server-reported `dt_ms` values from the completed primary-evaluation runs, after removing the same eight excluded trials as the leaderboard. Each sample times one action-chunk query, including preprocessing and action conversion within the server’s prediction call; it excludes HTTP transport, simulation and model loading. PNG decoding is inside some server timers and outside others. All recorded first calls are retained. ImageWAM uses precomputed instruction embeddings. The shared HTTP client labels these messages “`MolmoAct2 inference`” for all eight HTTP-served policies; policy identity comes from the completed run and checkpoint, not that generic log label. GR00T N1.7 did not record this metric and is marked NR. The returned chunk length can exceed the prefix actually executed before replanning, so chunk latency is not per-control-step latency or an end-to-end control rate.

Table 8: **Measured learned-policy inference latency.** Server-reported wall time per action-chunk query over the 916 scored primary-evaluation episodes, pooled across completed runs (one evaluation worker). First calls are retained. Steps are returned / executed, not necessarily the underlying model’s full prediction horizon. Median and 95th percentile (P95) are in milliseconds. Hardware differs across policies; these are deployment measurements, not a controlled hardware comparison. NR: not recorded.

Policy	GPU	Steps	Queries	Median (ms)	P95 (ms)
GR00T N1.7	RTX 3090	– / 16	NR	NR	NR
$\pi_{0.5}$	RTX A5000	30 / 15	44,334	337.0	376.9
Galaxea 0.5	RTX A5000	27 / 15	33,422	791.8	933.8
MolmoAct2-SO101	RTX A5000	30 / 30	26,046	696.8	741.3
MolmoAct2 base	RTX A5000	30 / 30	25,661	775.3	818.6
FLUX 3 Action SO-101 LoRA	A40	42 / 32	25,080	2894.0	3129.4
ImageWAM	RTX A5000	16 / 16	41,721	322.5	410.0
LaWAM	RTX A5000	36 / 36	18,651	274.1	333.4
InternW0- Δ	A40	16 / 16	34,378	986.5	1089.8

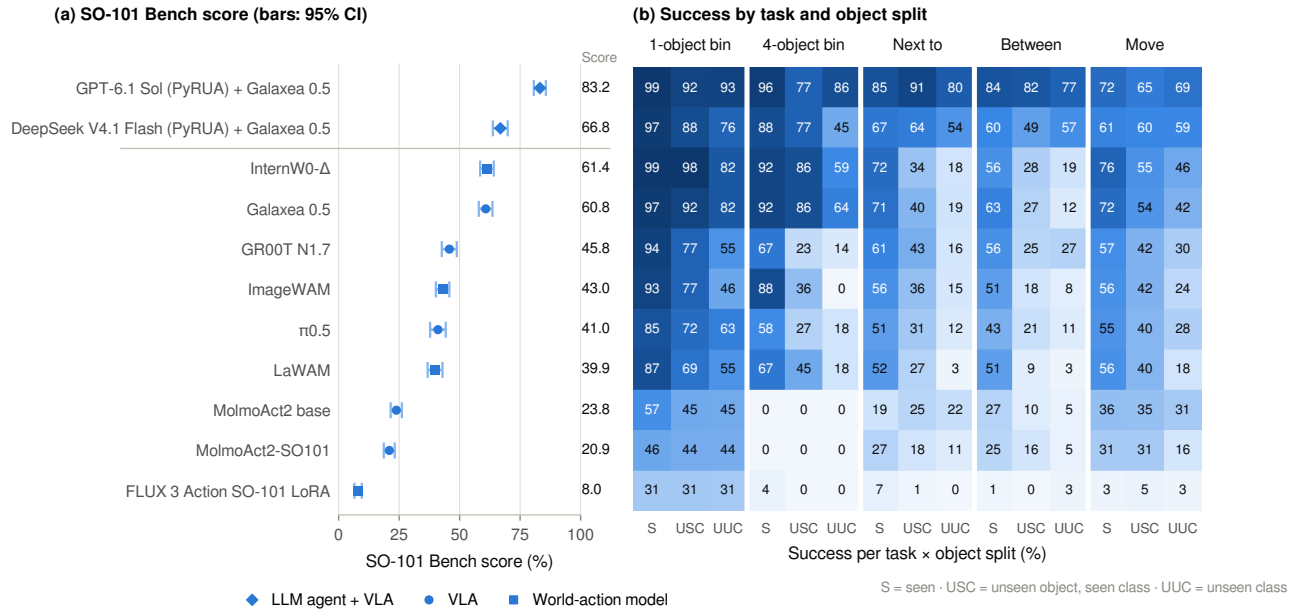


Figure 17: Score and success in all 15 task × object-split cells.

Table 9: **SO-101 Bench leaderboard** (primary evaluation, 916 episodes). The score is the mean success over the 15 task $\times$ object-split cells, with a 95% bootstrap CI that resamples episodes within cells. Remaining columns are raw success rates (%) over the n episodes in the header. All learned policies were fine-tuned on so101_bench_sim; LLM agents are frozen PyRUA-Lean v9.6 harnesses that hand grasping to Galaxea 0.5 and perform placement themselves.

Policy	Type	Score [95% CI]	1-bin <i>n</i> =202	4-bin <i>n</i> =68	Next to <i>n</i> =216	Between <i>n</i> =216	Move <i>n</i> =214	S 316	USC 285	UUC 315	4-bin prog.
1 GPT-6.1 Sol (PyRUA) + Galaxea 0.5	LLM agent + VLA	83.2 [80.7, 85.7]	95	87	85	81	69	85	82	80	94
2 DeepSeek V4.1 Flash-1017 (PyRUA) + Galaxea 0.5	LLM agent + VLA	66.8 [63.7, 69.9]	87	71	62	56	60	72	66	60	81
3 InternW0- Δ	WAM	61.4 [58.5, 64.1]	93	79	42	35	59	76	56	42	91
4 Galaxea 0.5	VLA	60.8 [57.9, 63.6]	90	81	44	34	56	76	55	40	90
5 GR00T N1.7	VLA	45.8 [42.6, 48.8]	75	35	40	37	43	66	45	30	52
6 ImageWAM	WAM	43.0 [40.2, 45.7]	71	43	36	26	41	65	42	22	58
7 $\pi_{0.5}$	VLA	41.0 [37.8, 44.3]	73	35	31	25	41	58	40	28	65
8 LaWAM	WAM	39.9 [36.8, 42.9]	70	44	27	21	38	61	36	19	60
9 MolmoAct2 base	VLA	23.8 [21.5, 26.1]	49	0	22	14	34	31	27	24	18
10 MolmoAct2-SO101	VLA	20.9 [18.6, 23.2]	45	0	19	16	26	29	25	17	22
11 FLUX 3 Action SO-101 LoRA	WAM	8.0 [6.5, 9.5]	31	1	3	1	3	9	8	8	15

Table 10: **Leaderboard, all 15 task $\times$ object-split cells.** Success as % (successes/episodes).

Policy	1-object bin			4-object bin			Next to			Between			Move		
	S	USC	UUC	S	USC	UUC	S	USC	UUC	S	USC	UUC	S	USC	UUC
GPT-6.1 Sol (PyRUA) + Galaxea 0.5	99% (66/67)	92% (59/64)	93% (66/71)	96% (23/24)	77% (17/22)	86% (19/22)	85% (64/75)	91% (61/67)	80% (59/74)	84% (63/75)	82% (55/67)	77% (57/74)	72% (54/75)	65% (42/65)	69% (51/74)
DeepSeek V4.1 Flash-1017 (PyRUA) + Galaxea 0.5	97% (65/67)	88% (56/64)	76% (54/71)	88% (21/24)	77% (17/22)	45% (10/22)	67% (50/75)	64% (43/67)	54% (40/74)	60% (45/75)	49% (33/67)	57% (42/74)	61% (46/75)	60% (39/65)	59% (44/74)
InternW0- Δ	99% (66/67)	98% (63/64)	82% (58/71)	92% (22/24)	86% (19/22)	59% (13/22)	72% (54/75)	34% (23/67)	18% (13/74)	56% (42/75)	28% (19/67)	19% (14/74)	76% (57/75)	55% (36/65)	46% (34/74)
Galaxea 0.5	97% (65/67)	92% (59/64)	82% (58/71)	92% (22/24)	86% (19/22)	64% (14/22)	71% (53/75)	40% (27/67)	19% (14/74)	63% (47/75)	27% (18/67)	12% (9/74)	72% (54/75)	54% (35/65)	42% (31/74)
GR00T N1.7	94% (63/67)	77% (49/64)	55% (39/71)	67% (16/24)	23% (5/22)	14% (3/22)	61% (46/75)	43% (29/67)	16% (12/74)	56% (42/75)	25% (17/67)	27% (20/74)	57% (43/75)	42% (27/65)	30% (22/74)
ImageWAM	93% (62/67)	77% (49/64)	46% (33/71)	88% (21/24)	36% (8/22)	0% (0/22)	56% (42/75)	36% (24/67)	15% (11/74)	51% (38/75)	18% (12/67)	8% (6/74)	56% (42/75)	42% (27/65)	24% (18/74)
$\pi_{0.5}$	85% (57/67)	72% (46/64)	63% (45/71)	58% (14/24)	27% (6/22)	18% (4/22)	51% (38/75)	31% (21/67)	12% (9/74)	43% (32/75)	21% (14/67)	11% (8/74)	55% (41/75)	40% (26/65)	28% (21/74)
LaWAM	87% (58/67)	69% (44/64)	55% (39/71)	67% (16/24)	45% (10/22)	18% (4/22)	52% (39/75)	27% (18/67)	3% (2/74)	9% (38/75)	3% (6/67)	5% (2/74)	3% (42/75)	56% (26/65)	18% (13/74)
MolmoAct2 base	57% (38/67)	45% (29/64)	45% (32/71)	0% (0/24)	0% (0/22)	0% (0/22)	19% (14/75)	25% (17/67)	22% (16/74)	27% (20/75)	10% (7/67)	5% (4/74)	5% (27/75)	36% (23/65)	31% (23/74)
MolmoAct2-SO101	46% (31/67)	44% (28/64)	44% (31/71)	0% (0/24)	0% (0/22)	0% (0/22)	27% (20/75)	18% (12/67)	11% (8/74)	25% (19/75)	16% (11/67)	5% (4/74)	31% (23/75)	31% (20/65)	16% (12/74)
FLUX 3 Action SO-101 LoRA	31% (21/67)	31% (20/64)	31% (22/71)	4% (1/24)	0% (0/22)	0% (0/22)	7% (5/75)	1% (1/67)	0% (0/74)	1% (1/75)	0% (0/67)	3% (2/74)	3% (2/75)	5% (3/65)	3% (2/74)

Table 11: **Evaluation subsets and visual conditions.** ROBUST (held-out backgrounds and robot colours), PLAN and MEM are tagged subsets of the same 916 episodes; appearance profiles partition them. Profiles use different layouts, so the columns are not a paired visual-robustness test.

Policy	Subsets			Appearance profile		
	ROBUST	PLAN	MEM	Canonical	Varied	Held-out
GPT-6.1 Sol (PyRUA) + Galaxea 0.5	82% (146/178)	75% (36/48)	80% (41/51)	82% (307/374)	83% (303/364)	82% (146/178)
DeepSeek V4.1 Flash-1017 (PyRUA) + Galaxea 0.5	62% (111/178)	35% (17/48)	55% (28/51)	66% (248/374)	68% (246/364)	62% (111/178)
InternW0- Δ	53% (94/178)	12% (6/48)	47% (24/51)	59% (219/374)	60% (220/364)	53% (94/178)
Galaxea 0.5	57% (102/178)	15% (7/48)	31% (16/51)	57% (214/374)	57% (209/364)	57% (102/178)
GR00T N1.7	48% (85/178)	17% (8/48)	43% (22/51)	49% (184/374)	45% (164/364)	48% (85/178)
ImageWAM	34% (60/178)	8% (4/48)	27% (14/51)	47% (174/374)	44% (159/364)	34% (60/178)
$\pi_{0.5}$	36% (64/178)	8% (4/48)	22% (11/51)	45% (167/374)	41% (151/364)	36% (64/178)
LaWAM	18% (32/178)	6% (3/48)	29% (15/51)	46% (173/374)	42% (152/364)	18% (32/178)
MolmoAct2 base	28% (50/178)	4% (2/48)	24% (12/51)	30% (112/374)	24% (88/364)	28% (50/178)
MolmoAct2-SO101	26% (47/178)	6% (3/48)	20% (10/51)	23% (86/374)	24% (86/364)	26% (47/178)
FLUX 3 Action SO-101 LoRA	4% (7/178)	0% (0/48)	6% (3/51)	10% (39/374)	9% (34/364)	4% (7/178)

Table 12: **Evaluated checkpoints.** Hugging Face repository and revision (learned policies) or frozen harness (LLM agents), and the evaluation hardware.

Policy	Checkpoint	Hardware
GPT-6.1 Sol (PyRUA) + Galaxea 0.5	PyRUA-Lean v9.6 + G0.5 step25600 (@957d33b)	RunPod A40, 2 isolated agent workers per pod
DeepSeek V4.1 Flash (PyRUA) + Galaxea 0.5	PyRUA-Lean v9.6 + G0.5 step25600 (@957d33b)	RunPod A40, 2 isolated agent workers per pod
InternW0-Δ	Shadytru/so101_bench_InternW0-Delta@8316ccc	RunPod A40, 1 worker
Galaxea 0.5	Shadytru/so101_bench_G0.5_full_fine_tune@957d33b (step25600)	RunPod RTX A5000 secure, 1 worker
GR00T N1.7	Shadytru/so101_bench_GR00T_N1.7-3B@572a932 (step16000)	workstation RTX 3090, 1 worker
ImageWAM	Shadytru/so101_bench_ImageWAM-FLUX.2-4B_full_fine_tune@0ba49ae	RunPod RTX A5000, 1 worker
$\pi_{0.5}$	Shadytru/so101_bench_pi05@de712e6 (step18000)	RunPod RTX A5000 secure, 1 worker
LaWAM	Shadytru/so101_bench_LaWAM_full_fine_tune@a7ddd36	RunPod RTX A5000, 1 worker
MolmoAct2 base	Shadytru/so101_bench_MolmoAct2-SO101_WM_base_full_fine_tune@723a9a8	RunPod RTX A5000, 1 worker
MolmoAct2-SO101	Shadytru/so101_bench_MolmoAct2-SO101_WM_full_fine_tune@8d04677	RunPod RTX A5000, 1 worker
FLUX 3 Action SO-101 LoRA	Shadytru/so101_bench_flux-3-action-so101@ac49157	RunPod A40, 1 worker

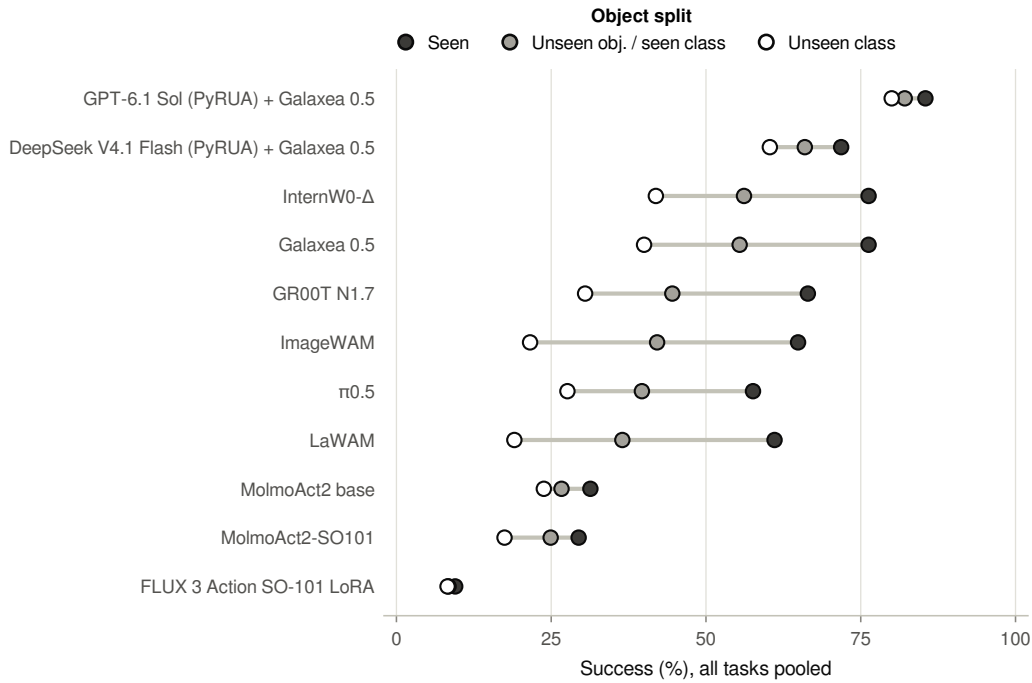


Figure 18: Success by object split, all tasks pooled.

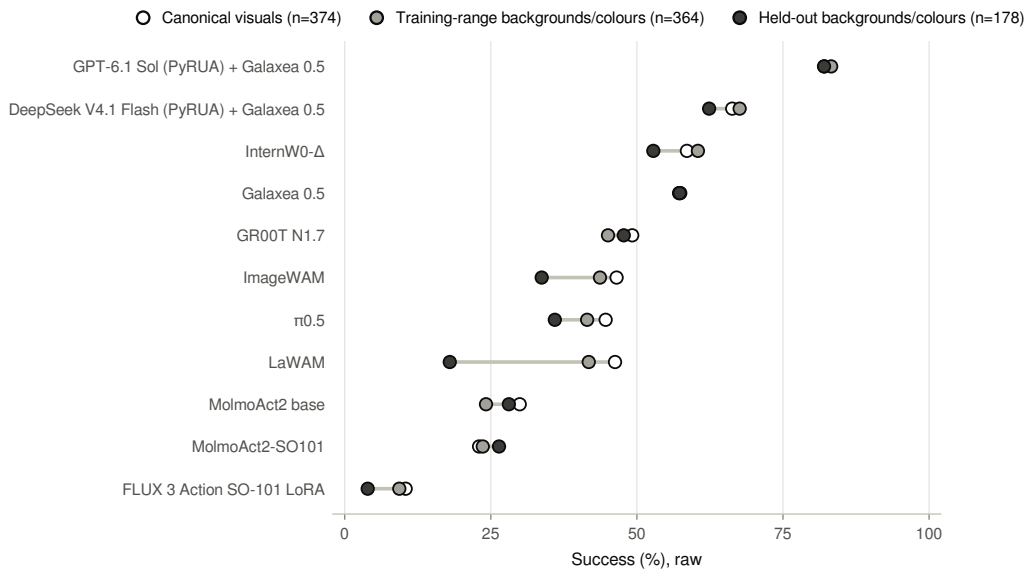


Figure 19: Success by appearance profile: canonical, training-range and held-out visuals.

checkpoint (estimated from measured s/update). so101_bench_sim_WM adds episode-start (init) frames to so101_bench_sim's overhead and wrist cameras.
“?” = not yet recorded.

(a) Performance (success %, n episodes in each header)

Policy	Type	Score [95% CI]	All	S	USC	UUC	1-bin	4-bin	Next to	Between	Move	4-bin prog.	MEM	PLAN	ROBUST
			916	316	285	315	202	68	216	216	214	68	51	48	178
GPT-6.1 Sol (PyRUA) + Galaxea 0.5	LLM agent + VLA	83.2 [80.7, 85.7]	83	85	82	80	95	87	85	81	69	94	80	75	82
DeepSeek V4.1 Flash-1017 (PyRUA) + Galaxea 0.5	LLM agent + VLA	66.8 [63.7, 69.9]	66	72	66	60	87	71	62	56	60	81	55	35	62
InternW0-Delta	WAM (video generation)	61.4 [58.5, 64.1]	58	76	56	42	93	79	42	35	59	91	47	12	53
Galaxea 0.5	VLA	60.8 [57.9, 63.6]	57	76	55	40	90	81	44	34	56	90	31	15	57
GR00T N1.7	VLA	45.8 [42.6, 48.8]	47	66	45	30	75	35	44	37	43	52	43	17	48
ImageWAM	WAM (image editing)	43.0 [40.2, 45.7]	43	65	42	22	71	43	36	26	41	58	27	8	34
$\pi 0.5$	VLA	41.0 [37.8, 44.3]	42	58	40	28	73	35	31	25	41	65	22	8	36
LaWAM	WAM (latent action)	39.9 [36.8, 42.9]	39	61	36	19	70	44	27	21	38	60	29	6	18
MolmoAct2 base	VLA	23.8 [21.5, 26.1]	27	31	27	24	49	0	22	14	34	18	24	4	28
MolmoAct2-SO101	VLA	20.9 [18.6, 23.2]	24	29	25	17	45	0	19	16	26	22	20	6	26
FLUX 3 Action SO-101 LoRA	WAM (video generation)	8.0 [6.5, 9.5]	9	9	8	8	31	1	3	1	3	15	6	0	4

(b) Evaluation cost (916 episodes)

Policy	Hardware	Sim time (h)	GPU-h	API \$	API \$/episode
GPT-6.1 Sol (PyRUA) + Galaxea 0.5	A40, 2 agent workers per GPU	5.7	—	216	0.236
DeepSeek V4.1 Flash-1017 (PyRUA) + Galaxea 0.5	A40, 2 agent workers per GPU	5.8	—	22	0.024
InternW0-Delta	A40	5.4	50	—	—
Galaxea 0.5	RTX A5000	5.0	44	—	—
GR00T N1.7	RTX 3090	6.6	39	—	—
ImageWAM	RTX A5000	6.5	53	—	—
$\pi 0.5$	RTX A5000	6.5	54	—	—
LaWAM	RTX A5000	6.5	59	—	—
MolmoAct2 base	RTX A5000	7.4	59	—	—
MolmoAct2-SO101	RTX A5000	7.5	56	—	—
FLUX 3 Action SO-101 LoRA	A40	7.7	105	—	—

(c) Fine-tuning (to the evaluated checkpoint; equal 36 h budget on $4 \times A100-80GB$)

Policy	Trainable parameters	Memory	Updates $\times$ batch	Samples	Epochs	% of LIBERO recipe	A100-h	W&B
GPT-6.1 Sol (PyRUA) + Galaxea 0.5	0 (LLM); G0.5 as below	Agent context (full episode) + G0.5's frame history	LLM frozen (API); uses the G0.5 checkpoint below					
DeepSeek V4.1 Flash-1017 (PyRUA) + Galaxea 0.5	0 (LLM); G0.5 as below	Agent context (full episode) + G0.5's frame history	LLM frozen (API); uses the G0.5 checkpoint below					
Galaxea 0.5	Full (Wan2.2-5B video expert + 1.2B action expert; RymnBrain-2B VLM frozen)	Episode-start anchor frame + frame 32 steps back	18,200 $\times$ 128	2.33M	1.17	?	142	run
GR00T N1.7	Full (8-bit AdamW); count TBD	6 observation frames at 1 s stride	25,600 $\times$ 128	3.28M	1.64	?	134	run
ImageWAM	Projector + diffusion action head + VLLN (LLM and vision frozen)	Episode-start frames (overhead + wrist init)	16,000 $\times$ 640	10.24M	5.13	80%	142	project
$\pi 0.5$	FLUX.2-klein-4B editing branch + $\sim 642M$ action DiT (text encoder frozen)	None (current frame)	30,000 $\times$ 128	3.84M	1.92	?	130	run
LaWAM	Full, 4.14B	Episode-start frames (overhead + wrist init)	9,000 $\times$ 256	2.30M	1.15	30%	136	run
MolmoAct2 base	Full (Qwen3-VL-2B + DINOv3 + LAM)	None (current frame)	45,000 $\times$ 256	11.52M	5.77	?	132	run
MolmoAct2-SO101	Full, $\sim 5.4B$	Episode-start frames (overhead + wrist init)	8,000 $\times$ 128	1.02M	0.51	16%	110	run
FLUX 3 Action SO-101 LoRA	Full, $\sim 5.4B$	Episode-start frames (overhead + wrist init)	10,000 $\times$ 128	1.28M	0.64	20%	129	run
	LoRA r32 on attention/MLP + action heads (full)	8 ticks (0.27 s) of state history + 2 frames	3,000 $\times$ 128	0.38M	0.19	?	69	run

(d) Data

Policy	Initial checkpoint	SO-101 in pre-training	SO-101 in post-training	Fine-tuning dataset
GPT-6.1 Sol (PyRUA) + Galaxea 0.5	gpt-6.1-sol (API, frozen) + G0.5 step 25600	via G0.5	via G0.5	none (LLM frozen); G0.5 as below
DeepSeek V4.1 Flash-1017 (PyRUA) + Galaxea 0.5	deepseek-flash-1017 (API, frozen) + G0.5 step 25600	via G0.5	via G0.5	none (LLM frozen); G0.5 as below
InternW0-Delta	InternRobotics/InternW0-Delta-Base ca53444	@ ?	No	so101_bench_sim
Galaxea 0.5	Galaxea g05-base	?	?	so101_bench_sim
GR00T N1.7	nvidia/GR00T-N1.7-3B	?	No	so101_bench_sim_WM (+ init frames)
ImageWAM	yuyangalin/ImageWAM-FLUX.2-4B-InternData-A1-EE @ 51a71ed	No	No	so101_bench_sim
$\pi 0.5$	lerobot/pi05_base @ b211f3d	?	No	so101_bench_sim_WM (+ init frames)

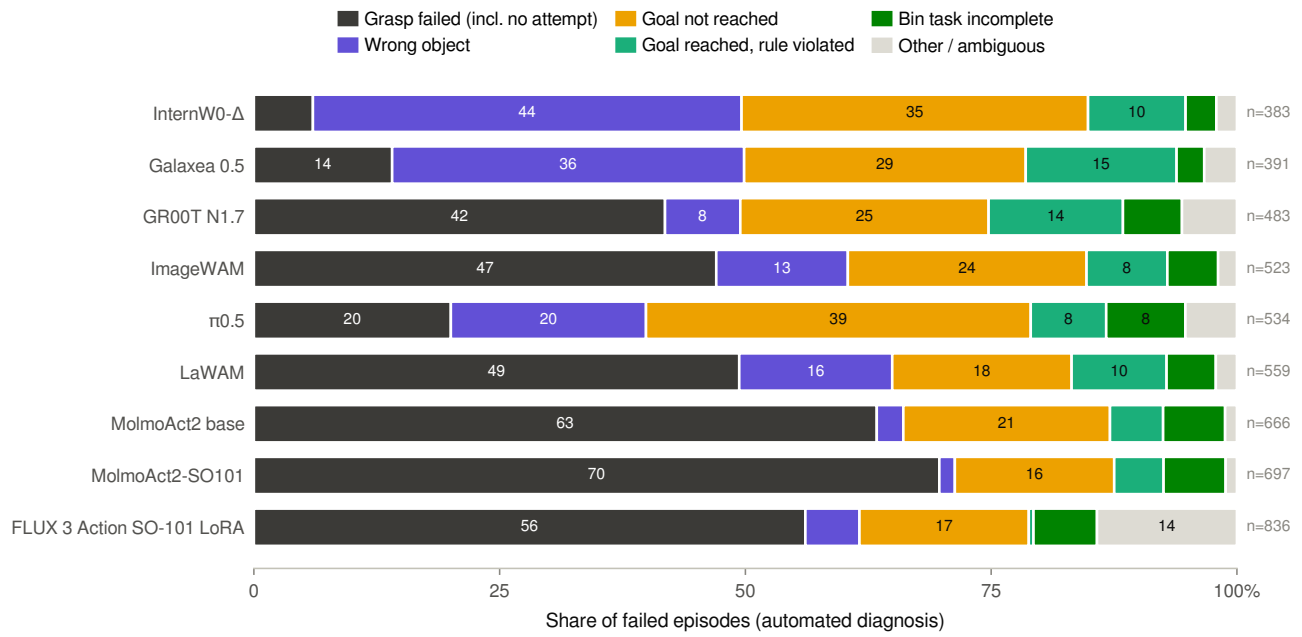


Figure 20: Automated failure diagnosis for the learned policies. [TODO: Only include after validating the classifier against human labels.]

Table 14: **Harness ablation** on 60 paired primary-evaluation episodes (30 easy, 30 hard; six per task in each stratum, selected from the nine learned policies’ outcomes before any LLM run). Direct: the LLM issues end-effector tool calls. PyRUA-Lean: code-as-policy with perception and IK tools. + Galaxea 0.5: the same harness hands grasping to Galaxea 0.5; the LLM performs placement. API cost is an estimate at list prices. [†]Stopped by the operator after the listed episodes (easy stratum only).

LLM	Harness	All 60	Easy 30	Hard 30	1-bin	4-bin	Next to	Between	Move
GPT-6.1 Sol	Direct tool calls	35% (21/60)	47% (14/30)	23% (7/30)	1/12	0/12	6/12	6/12	8/12
	PyRUA-Lean	67% (40/60)	73% (22/30)	60% (18/30)	8/12	5/12	9/12	10/12	8/12
	PyRUA-Lean + Galaxea 0.5	85% (51/60)	93% (28/30)	77% (23/30)	11/12	10/12	11/12	12/12	7/12
Gemini 4 Argon	Direct tool calls [†]	19% (4/21)	19% (4/21)	–	0/5	0/5	0/4	0/3	4/4
	PyRUA-Lean	65% (39/60)	70% (21/30)	60% (18/30)	8/12	2/12	9/12	12/12	8/12
	PyRUA-Lean + Galaxea 0.5	77% (46/60)	87% (26/30)	67% (20/30)	9/12	9/12	9/12	10/12	9/12
DeepSeek V4.1 Flash-1017	Direct tool calls [†]	27% (3/11)	27% (3/11)	–	0/2	0/3	0/2	1/2	2/2
	PyRUA-Lean	30% (18/60)	43% (13/30)	17% (5/30)	4/12	0/12	5/12	6/12	3/12
	PyRUA-Lean + Galaxea 0.5	65% (39/60)	83% (25/30)	47% (14/30)	12/12	9/12	6/12	4/12	8/12
Galaxea 0.5 alone (reference)		53% (32/60)	93% (28/30)	13% (4/30)	9/12	7/12	5/12	5/12	6/12

Table 15: **LLM ablation** with the PyRUA-Lean v9.6 + Galaxea 0.5 harness held fixed, on the same 60 paired episodes. Effort is the provider reasoning setting. API cost is an estimate at list prices (Gemini: gross Vertex list rate). The last column repeats the full-benchmark score where the hybrid was run on all 916 episodes.

LLM	Effort	Runtime	All 60	Easy	Hard	1-bin	4-bin	Next to	Between	Move	\$/episode
GPT-6.1 Sol	high	Codex	85% (51/60)	93% (28/30)	77% (23/30)	11/12	10/12	11/12	12/12	7/12	0.245
Gemini 4 Argon	medium	OpenAI-compatible chat (Vertex)	77% (46/60)	87% (26/30)	67% (20/30)	9/12	9/12	9/12	10/12	9/12	1.237
DeepSeek V4.1 Flash-1017	high	Codex	65% (39/60)	83% (25/30)	47% (14/30)	12/12	9/12	6/12	4/12	8/12	0.023
MiMo v2.6 Pro	high	Codex via Responses→Chat bridge	55% (33/60)	73% (22/30)	37% (11/30)	7/12	8/12	7/12	7/12	4/12	0.044
Galaxea 0.5 alone			53% (32/60)	93% (28/30)	13% (4/30)	9/12	7/12	5/12	5/12	6/12	0

D LLM Agents: Additional Results

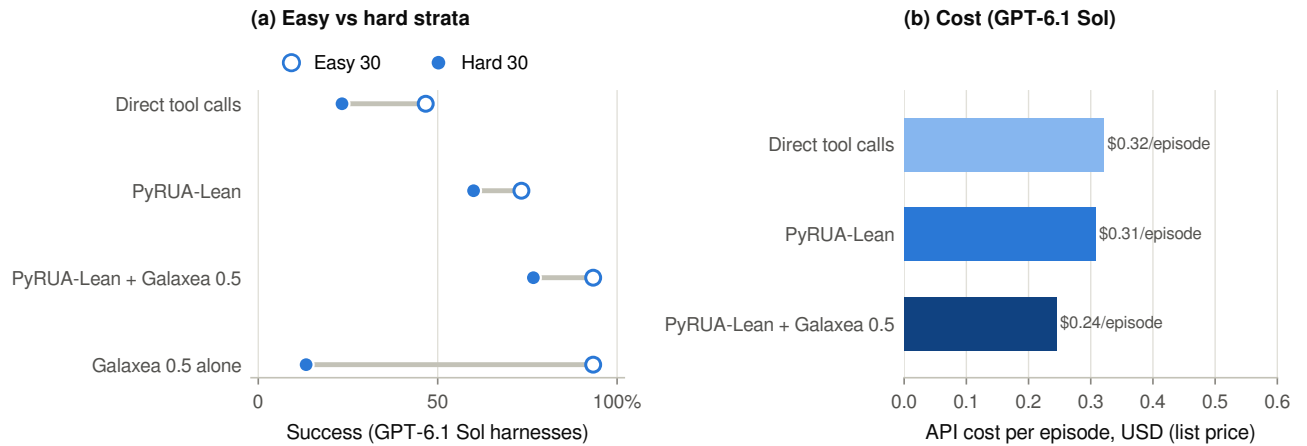


Figure 21: Harness ablation by stratum (easy30 / hard30) and cost.

PyRUA-Lean + Galaxea 0.5, successes per task and stratum

GPT-6.1 Sol	11/12	10/12	11/12	12/12	7/12	28/30	23/30
Gemini 4 Argon	9/12	9/12	9/12	10/12	9/12	26/30	20/30
DeepSeek V4.1 Flash	12/12	9/12	6/12	4/12	8/12	25/30	14/30
MiMo v2.6 Pro	7/12	8/12	7/12	7/12	4/12	22/30	11/30
Galaxea 0.5 alone	9/12	7/12	5/12	5/12	6/12	28/30	4/30
	1-bin	4-bin	Next to	Between	Move	Easy 30	Hard 30

Figure 22: Successes per task and stratum for each LLM, and for Galaxea 0.5 alone.

E Object Real2Sim Pipeline

Every object passes through the same pipeline, which produces a rigid body with separate visual and collision geometry.

1. Asset generation. We start from an image of the object: a photograph of the real object, an image from the internet, or an image from a text-to-image model. An image-to-3D model accessed through 3D AI Studio [71] converts it into a textured mesh (GLB) with physically based rendering (PBR) materials.

2. Mesh processing (Blender). The mesh is scaled to the object’s real dimensions, oriented, and given an origin at its base, which serves as the object’s root pose for the success conditions. Texture colours are corrected towards the object’s real palette and brightness. The mesh is then duplicated into a *visual* and a *physics* mesh.

- **Visual mesh:** remeshed and decimated to a fixed visual budget, repaired (loose geometry removed, duplicate vertices merged, no non-manifold edges), and the original colour and roughness are baked onto it.
- **Physics mesh:** cleaned and remeshed to a coarser physical budget, avoiding small crevices in which the gripper could become caught. Over-faceted results are smoothed, remaining non-manifold regions are filled or remeshed at a larger voxel size, and face normals are made consistent. The physics mesh is checked to align with the visual mesh.

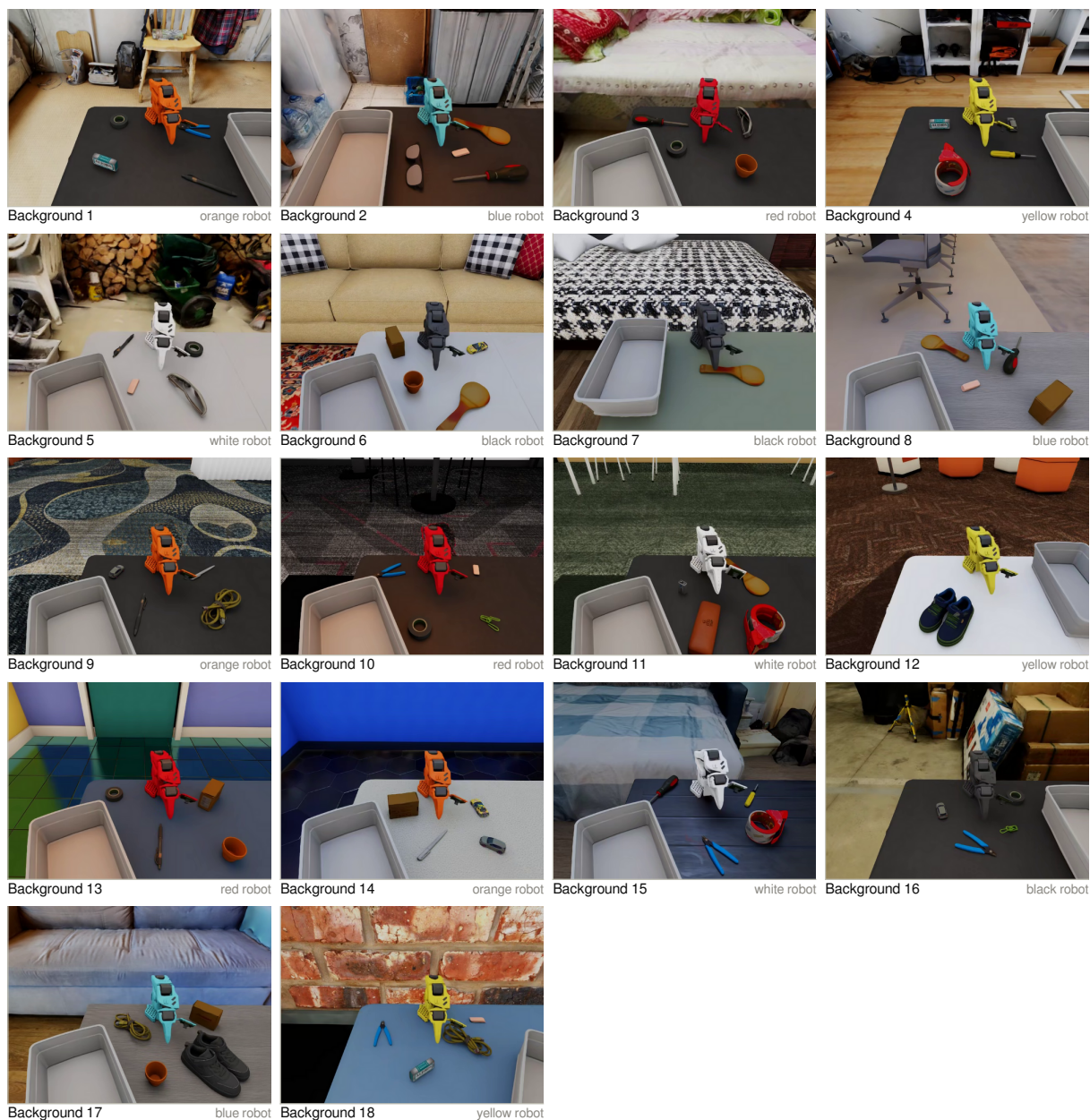
Both meshes are exported together as a USD file with packed textures.

3. Physics setup (Isaac Sim). The root prim is made a rigid body with mass. The physics mesh is given a collider, using a signed-distance-field (SDF) collider for complex non-convex shapes without thin walls and a convex hull or convex decomposition otherwise. A physics material sets static and dynamic friction and restitution, and the physics mesh is hidden from rendering.

4. Appearance tuning (Isaac Lab). The visual mesh’s colour texture is adjusted, usually by tuning its brightness and occasionally by repainting regions, until the object’s appearance under the benchmark cameras matches the real object.

F Simulation Backgrounds

Training backgrounds (18): used in so101_bench_sim and in the “varied” evaluation episodes



Held-out backgrounds (4): ROBUST evaluation subset only, with held-out robot colours

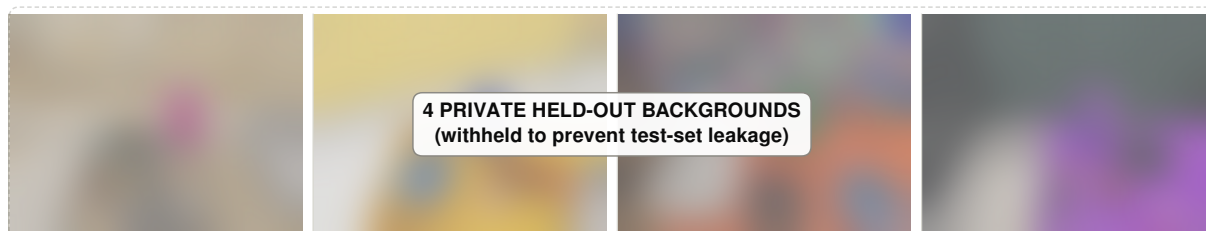


Figure 23: **Simulation backgrounds.** The 18 training backgrounds (shown with varied robot colours) and the 4 held-out ROBUST backgrounds, which are withheld to prevent test-set leakage.