ScaleEvals: Scaling Generalist Policy Evaluation through Reset-Free Autonomous Play

Amitoj Sandhu, Ethan Sick, Omkaar Buddhikot, Nadia Laswi, Wesley Durbano, Nadun Ranawaka Arachchige[†], You Liang Tan[†]

[†]Project Leads

amitsandhu1@g.ucla.edu, esick@scu.edu, omkaar.buddikot@sjsu.edu,

nadialaswi@berkeley.edu, wdurbano@scu.edu, nadun.ranawaka@gatech.edu, youliang@berkeley.edu

Work done at NVIDIA

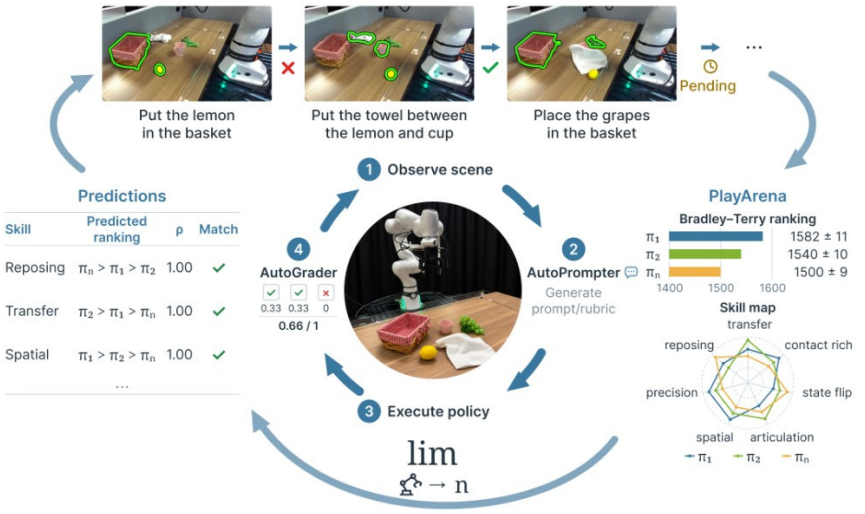


Figure 1: **Reset-free continuous-play evaluation.** AutoPrompter generates a grounded task and rubric from the current scene, a policy executes it, and AutoGrader scores the rollout. The resulting scene seeds the next task; accumulated evidence yields policy rankings and skill-conditioned diagnostics.

Abstract: Real-world evaluation of generalist robot policies is bottlenecked by manual scene resets, human grading, and small fixed task suites. We present SCALEEVALS, a reset-free protocol that evaluates policies through continuous physical play. A VLM AutoPrompter generates scene-grounded tasks and rubrics, policies act in the evolving scene, and a VLM AutoGrader measures task progress; PlayArena matches related rollouts and produces global Bradley–Terry rankings and skill-conditioned diagnostics. Across five policies and 3,486 physical rollouts on DROID and YAM, SCALEEVALS achieves 23.0 net rollouts/h per station and reduces active human labor by 98%. AutoGrader reaches 90.7% criterion agreement with human audit while preserving policy order (r=0.99). On DROID, reset-free rankings correlate with held-out manual evaluation (r=0.94, p=0.018); on YAM, Bradley–Terry rating and mean task progress remain correlated across four anonymized policies (r=0.839, ρ=0.800). Finally, recomposed verb-, skill-, and object-conditioned evidence distinguishes closely ranked $\pi_{0.5}$ and GR00T N1.7 and predicts 9/10 full three-policy orderings (240 rollouts; mean ρ=0.887, 96% pairwise concordance). SCALEEVALS therefore complements controlled reset-required suites with scalable physical evaluation and capability-level diagnostics.
Project website: scaleevals.github.io

Keywords: Generalist Robot Policy Evaluation, Scalability, Benchmark

10th Conference on Robot Learning (CoRL 2026), Austin TX, USA.

1 Introduction

Real-world evaluation is a bottleneck for generalist robot policy development. Current physical evaluation protocols typically rely on fixed task suites, manual scene resets, and human grading, making them costly to scale and limiting feedback to sparse success rates. Simulation-based evaluation provides a scalable alternative, but sim-to-real correlation can depend on embodiment, camera setup, task distribution, and contact regime [1, 2]. Physical evaluation therefore remains essential, especially for manipulation tasks involving clutter, contact-rich interaction, articulated objects, and evolving scene state [3].

Beyond throughput, current evaluation protocols provide limited diagnostic resolution. Aggregate task progress and success rates can obscure whether a policy fails because of spatial reasoning, precision, contact-rich interaction, object state changes, or long-horizon sequencing. Fixed task suites may also overrepresent particular objects, relations, language forms, or interaction modes. Recent diagnostic benchmarks such as INT-ACT [4] highlight this issue by showing that VLA policies can often infer the correct high-level intent while failing to execute it reliably.

We propose SCALEEVALS, a reset-free framework for scalable real-world evaluation of generalist robot policies. Instead of evaluating policies through isolated reset-required trials, SCALEEVALS runs each robot station as a continuous play stream: tasks are generated from the current scene, policies act in the evolving environment, and outcomes are automatically graded and aggregated. This turns real-world evaluation from a sequence of manually reset benchmark trials into an autonomous evaluation loop. Summary of contributions:

1. **Continuous play as evaluation:** SCALEEVALS generates scene-aware tasks online, with each rollout starting from the scene state left by previous rollouts rather than from a manually reset configuration. Unlike reset-required suites that control initial states, ScaleEvals evaluates language-conditioned steering through evolving scenes. It complements rather than replaces controlled spatial- and initial-state-generalization tests.
2. **Automated task generation and grading:** A VLM-based AutoPrompter generates grounded instructions, milestone rubrics, difficulty annotations, and skill tags. A VLM-based AutoGrader scores rollout videos at subtask granularity, producing task-progress estimates without task-specific fine-tuning.
3. **Matched ranking and tag-conditioned diagnostics:** PlayArena matches heterogeneous rollouts by difficulty and task–scene similarity, aggregates progress-derived pairwise comparisons with a soft-win Bradley–Terry (BT) model, and uses tag-stratified analyses to expose relative policy strengths and weaknesses across skill axes.

Across five generalist policies and more than 3,000 real-world trajectories on DROID and YAM, SCALEEVALS achieves 23.0 net rollouts/h per station compared with 14.8 rollouts/h for the manual baseline, while reducing active human involvement from 4 hours to 5 minutes per 100 rollouts and maintaining 90.7% AutoGrader agreement with human rubric judgments. BT rankings correlate strongly with manual-evaluation mean task progress (Pearson $r{=}0.94, p{<}0.05$). Skill-tag-conditioned BT rankings support pre-evaluation prompt generation and rank prediction (Spearman $\rho{=}0.88$ before AutoGrader audit). SCALEEVALS collects comparable signal at higher net throughput while reducing active human time by 98%, and scales across embodiments. Finally, SCALEEVALS reveals how fixed manual suites can confound policy assessment. These results suggest that a reset-free paradigm facilitates scale as an autonomous, continuously running system that reliably assesses generalizability and produces interpretable policy diagnostics.

2 Related Work

Simulation-based evaluation. Simulation-based benchmarks improve evaluation throughput by replacing costly physical rollouts with reproducible simulated trials. Systems such as SIMPLER,

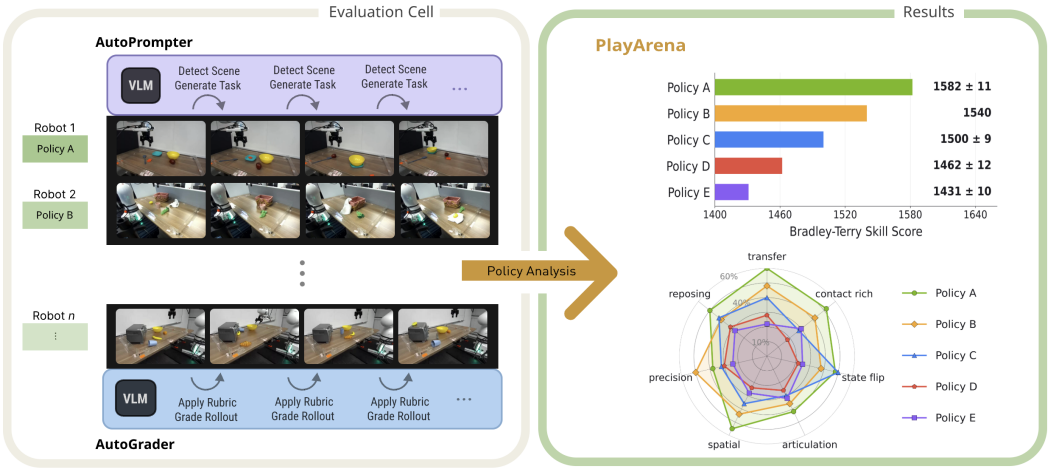


Figure 2: **Deployment and aggregation architecture.** Each evaluation cell runs policy streams through AutoPrompter and AutoGrader. PlayArena accumulates matched rollout evidence across cells to produce a global leaderboard and capability profiles.

RoboLab, and PolaRiS report encouraging sim-real alignment across manipulation settings, IsaacLab task suites, and neural scene reconstructions [1, 2, 5]. Diagnostic probing suites such as INT-ACT further show that high-level intent understanding can outpace execution reliability [4]. World-model approaches such as Ctrl-World and Interactive World Simulator provide controllable environments for policy evaluation and training [6, 7]. Despite these gains, transfer correlations are sensitive to embodiment, sensor setup, and contact dynamics; even small physics mismatches can change behavior, especially for contact-rich, articulated, or deformable interactions. Prior work (e.g., *Genesis*) reports strong but still bounded sim-to-real agreement [8]. World models can also accumulate rollout drift during multi-task, contact-rich, occluded, or long-horizon interactions. We therefore treat simulation and world models as useful proxies, while focusing on scalable real-world evaluation as the direct measure of physical generalization. SCALEEVALS targets this gap by scaling physical evaluation itself: it continuously probes policies in evolving real scenes without manual resets or per-rollout human intervention.

Autonomous play and Self Improvement. Predictive proxies of post-training performance (e.g., low validation MSE) are often unreliable in real-world deployment [9]. Autonomous play has therefore become a practical route for collecting diverse interaction experience and improving policies. PlayWorld shows that play-derived dynamics improve downstream RL performance [10], and recent work demonstrates autonomous improvement of instruction-following skills through foundation-model-driven loops [11]. Related efforts scale egocentric robot-data collection and online post-training across tasks, environments, and robot fleets [12, 13, 14]. In our setting, we use autonomous play for evaluation rather than policy updates: continuously evolving scenes generate broad behavior coverage, while skill tags provide a structured basis for fine-grained diagnostics.

Automated evaluations. Real-world evaluations require expensive, human annotators to assess trajectory outcomes. Existing automated methodologies include comparing final observation to a goal-state image using visual similarity or fine-tuning a VLM per task for success classification [3, 15]. Reward modeling methods such as Robometer leverage a fine-tuned VLM that runs inference concurrently to output continuous task progress in scalar form [15], which can be opaque in interpretability, computationally intensive, and susceptible to trajectories outside dominant patterns of the 1 million it is trained on. Recent work has highlighted the importance of graduating from binary success rates into partial completions through deploying cost-aware, VLM-based autograders proposed as an open source effort [16]. Recent autonomous evaluation systems, such as AutoEval, reduce human intervention but still rely on per-task models and reset policies for binary success detection [17]. Benchmarks such as RoboChallenge scale via API access, but remain tied to fixed

reset-required suites [18]. Fixed suites are useful for controlled comparison, yet are more susceptible to benchmark-specific fitting; continuous reset-free streams do not remove distributional bias, but can reduce opportunities for memorization and improve coverage of naturally evolving scene states.

Pairwise ranking and preference-based evaluation. Pairwise-comparison models such as Bradley–Terry provide a principled way to aggregate incomplete and imbalanced comparisons into global rankings [19, 20]. They have been widely used in preference-based evaluation and robot learning, including trajectory-preference optimization [15] and large-scale policy leaderboards such as RoboArena [17]. These methods are attractive for robot evaluation because policies need not be evaluated on identical fixed suites, and pairwise outcomes can be pooled across diverse trials. However, existing leaderboard-style rankings primarily summarize aggregate capability: they say which policy is stronger overall, but can take time to converge into stable rankings and provide limited insight into which interaction modes, language factors, or physical skills drive the difference. This is especially important for generalist policies, whose performance often exhibits a "jagged frontier" across capabilities [21].

A/B testing is suited to a fixed policy pair and task suite [22], but K policies require $K(K-1)/2$ separately corrected tests. Because SCALEEVALS yields incomplete, heterogeneous matched comparisons, a joint soft-win Bradley–Terry model aggregates all policies on one latent scale without identical prompts [19]. Similarly, while world model simulators attempt to replace physical execution, world models suffer from compounding rollout drift during contact-rich interactions, occlusion, and long-horizon retries, making physical play-at-scale a valuable diagnostic methodology. SCALEEVALS is complementary to leaderboard efforts such as RoboArena [17]: it serves as a local evaluation engine for testing in-development checkpoints against RoboArena-listed open-source policies while minimizing human effort, helping practitioners iterate toward more competitive submissions before committing to a distributed leaderboard evaluation.

SCALEEVALS builds on this ranking view but adapts it to reset-free, continuously generated real-world evaluations. Instead of requiring exact task matches, PlayArena forms weighted comparisons between semantically related rollouts using task, scene, difficulty, and skill-tag metadata. We convert AutoGrader progress scores into soft pairwise outcomes and fit a weighted Bradley–Terry model to estimate global policy strengths. To move beyond a single leaderboard number, we further stratify rollouts by skill tag and use one-way analysis of variance (ANOVA) to test whether policy identity explains task-progress variation within each capability slice. This combination yields both a global ranking and interpretable diagnostics over skills such as spatial reasoning, precision, articulation, contact-rich manipulation, and reposing.

3 ScaleEvals Overview

3.1 AutoPrompter: Continuous-Play Task Generation

Fixed benchmarks cannot cover the diversity of scenes, object states, and manipulation skills encountered by generalist policies. SCALEEVALS addresses this with AutoPrompter, a few-shot VLM-based task generator that converts the current scene into a grounded instruction, grading rubric, and task metadata. At each rollout, AutoPrompter detects visible objects and proposes a feasible manipulation task for the current scene. It records the motion verb, primary object, estimated difficulty, and primary/secondary skill tags, and generates a milestone rubric of yes/no visual checks for AutoGrader. After the policy acts, AutoPrompter conditions on the updated scene and task history to generate the next instruction, enabling continuous reset-free evaluation. These metadata make generated tasks auditable and comparable across policies. PlayArena uses them to match related rollouts with estimated difficulty and analyze performance by capability. Additional details are provided in Sec. A.

Because tasks are grounded in the objects present, scene composition directly drives behavioral variety: across 2,486 rollouts, AutoPrompter produced 26 distinct action verbs over 32 objects, and the

objects in-scene largely determine which skills are exercised (e.g., a drawer elicits articulation). The resulting prompts span all seven capability tags—state flip (38%), spatial reasoning (31%), transfer (28%), articulation (18%), precision (13%), contact-rich manipulation (12%), and reposing (11%); percentages exceed 100% because a single task may carry multiple tags. Operators can therefore increase variety simply by swapping objects into the scene, while keeping 5–7 props per scene, which we find optimal for throughput and policy performance (excess distractors degrade performance). Candidate tasks are executed only after automated checks for object visibility and safety, and failing tasks are regenerated automatically without human filtering. AutoPrompter is configurable through the scene object inventory, safe-to-lift and heavy-object constraints, recent task history, few-shot examples, and permitted action/task rules. Users can therefore shift the generated task distribution without model fine-tuning by changing scene contents or these prompt-level controls. Appendix Figure 9 visualizes the multi-label skill-tag distribution of accepted prompts.

3.2 AutoGrader: Completion Progress from Multimodal Evidence

SCALEEVALS uses the off-the-shelf VLM Gemini 3 Flash as a cost-effective, fine-tune-free AutoGrader. We also evaluated Nemotron and Gemini Pro backbones and selected Gemini 3 Flash for its balance of criterion-level accuracy, macro F1, and cost. Per trajectory, the frame rate (fps), batch size, and nth frame extraction can be configured prior to submitting an array of frames to the VLM. We group frames into temporally ordered batches, which are passed to the VLM with explicit batch identifiers. This provides a structured representation for temporal reasoning (e.g., referencing when an action occurs) and improves auditability by allowing graders to localize decisions to specific segments of the trajectory. Through a Visual Question Answering (VQA) style of subtask grading provided by Autoprompter, the VLM must answer yes/no per subtask. For example, for a "Place the marker in a cup task" the AutoPrompter will generate subtasks in VQA format such as i) *Did the robot grasp the marker?* ii) *Did the robot pick up the marker?* iii) *Was the marker placed in the cup?* for a normalized score ranging [0,1]. We have found 3 fps, batch size of 10 frames, sending every other frame, preserves 90.7% accuracy at the criterion level.

3.3 PlayArena: Matched Ranking and Capability Diagnostics

PlayArena aggregates reset-free rollout streams into policy rankings and skill-level diagnostics. Since AutoPrompter generates tasks online rather than replaying a fixed suite, policies may encounter different but related prompts. For each pair of policies, we construct a bipartite similarity matrix over their rollouts using instruction difficulty and hybrid task–scene similarity. The score favors shared task factors, including motion verbs and skill tags, and shared scene factors, including the primary manipulated object. We then apply the Hungarian algorithm to obtain the globally optimal one-to-one assignment under this similarity score, rather than greedily pairing each rollout with its nearest match. Higher-quality matches receive larger weights in the downstream Bradley–Terry ranking. Matching is fully automated and requires no manual pairing. Empirically, above 100 rollouts per policy, the induced rank order is insensitive to the matching hyperparameters, and 100% of rollouts obtain at least one match.

Let $m \in \mathcal{M}$ index a matched comparison between policies π_i and π_j, and let $P_i^{(m)}, P_j^{(m)} \in [0, 100]$ be their AutoGrader task-progress percentages. We convert each matched comparison into a fractional preference target,

$$y_{ij}^{(m)} = \frac{P_i^{(m)} + \epsilon}{P_i^{(m)} + P_j^{(m)} + 2\epsilon}, \qquad \epsilon = 0.01.$$

This target assigns more preference mass to the policy with higher task progress, maps equal progress to a tie-like value near 0.5, and handles mutual-zero comparisons. Since P is measured in percentage points, ϵ acts only as a numerical guard.

We fit policy strengths with a weighted soft-win Bradley–Terry model [19]:

$$\mathcal{L}(\theta) = -\sum_{m \in \mathcal{M}} \omega_m \left[y_{ij}^{(m)} \log \sigma(\theta_i - \theta_j) + (1 - y_{ij}^{(m)}) \log(1 - \sigma(\theta_i - \theta_j)) \right] + \lambda \|\theta\|_2^2.$$

Here, $\mathcal{M}$ is the set of matched comparisons, ω_m is the match weight, λ is the L_2 regularization coefficient, and i, j denote the two policies compared in match m. We report the resulting strengths on an Elo-like scale with a fixed anchor policy fixed to reduce ambiguity of Bradley–Terry scores.

For diagnostics, PlayArena stratifies task progress by skill tag and runs one-way ANOVA within each tag to test whether policy identity explains variation in task progress. We report F-statistics, p-values, Benjamini–Hochberg FDR-corrected significance, and effect size η^2. We analyze seven primary skill tags: *transfer, spatial reasoning, state flip, articulation, precision, contact-rich manipulation, and reposing*. Because prompts can contain multiple tags, a single rollout can contribute to multiple capability slices. Thus, PlayArena produces both a global policy ranking and tag-conditioned capability profiles.

4 Experiments

Our experiments serve to answer the following: (1) *How do reset-free play-based evaluations and rankings compare to traditional, human and reset-required approaches?* (2) *How sample-efficient is* SCALEEVALS*?* (3) *Can* SCALEEVALS *enable diagnostic insights and predictive qualities through classical statistics beyond performance ranking?*

4.1 Experimental Setup

Policies. We compare five generalist policies: GR00T N1.6, GR00T N1.7, DreamZero, MolmoAct2, and $\pi_{0.5}$ [23, 24, 25, 26]. This selection emulates academic labs comparing the generalizability of policy checkpoints. Across independent SCALEEVALS sessions, each policy receives identical initial scene setup and time budgets to limit computational advantage. All policies, with the exception of DreamZero, can be launched locally via the DROID robot platform [27].

Scenes and Setups. We evaluate in tabletop and room-scale scenes with varying object sets (5–7 per scene). To minimize human intervention, acrylic barriers confine the objects to the area navigable by the robot. Remote intervention, autonomous monitoring, workspace exclusion zones, and velocity/force thresholds constrain execution. Should the Franka arms encounter hardware issues, we implemented a slack-bot to notify the operator for intervention. Backgrounds can vary above cubicle dividers, the introduced noise serves as a proxy for real-world scene variance, counterbalanced by significant samples within-policy.



Figure 3: **Evaluation scenes.** Four tabletop and room-scale setups with different object sets. Panels show the original evaluation setups and object configurations used for physical rollouts.

4.2 Results

AutoGrader. We audit AutoGrader on 525 randomly sampled trajectories, corresponding to 1,454 rubric-level VQA judgments sampled across difficulty buckets and skill tags. AutoGrader achieves 90.7% criterion-level agreement with human graders, supporting its use as a scalable measurement primitive with subtask resolution. Residual errors skew toward false positives under depth ambiguity, occlusion, and uncertain contact state; Figure 13 and Section 6 provide further analysis.

Skill-tag profile agreement. Figure 6 compares per-tag capability profiles between reset-free ScaleEvals and held-out manual evaluation. The radar view highlights where the two protocols

	ScaleEvals (n=350 per policy)				Manual Eval Suite (up to n=40/policy)	
Policy	n	**BT** ↑	**TP %** ↑	**SR %** ↑	**TP %** ↑	**SR %** ↑
DreamZero	350	**1580** ± 16	**38.5**	**13.1**	**54.4**	**31.6**
$\pi_{0.5}$ (anchor)	350	1539	30.7	7.7	45.0	12.5
GR00T N1.7	350	1500 ± 17	23.2	3.4	43.9	10.0
MolmoAct2	350	1456 ± 16	17.3	2.6	27.0	5.0
GR00T N1.6	350	1453 ± 16	15.5	0.6	34.3	12.5

Table 1: **Policy performance under equal-N balancing (n=350 per policy). ScaleEvals** reports bootstrap-median (K=200) Bradley–Terry (BT), mean task progress (TP), and success rate (SR) on a per-skill-balanced experiment. $\pi_{0.5}$ is anchored at BT=1539 on the Elo scale ($400/\ln 10$; Hessian SE shown). **Manual Eval Suite** reports mean TP and SR on held-out human-graded reset-required tasks (up to 40 rollouts per policy; 198 observed rollouts total because DreamZero lacks two trials for one task).

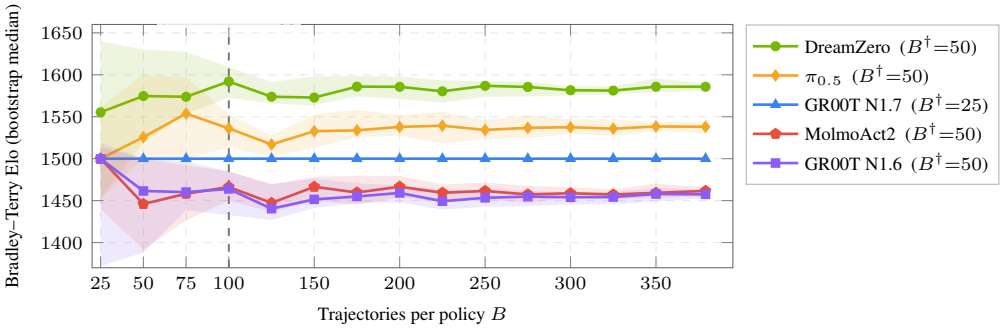


Figure 4: **ScaleEvals rank convergence (reset-free only).** Bradley–Terry Elo per policy versus rollout budget B under per-tag-balanced sub-sampling, computed entirely from reset-free ScaleEvals play (no manual data). For this convergence visualization, every bootstrap fit is recentered to GR00T N1.7=1500; this additive shift does not change ordering and differs from the $\pi_{0.5}$=1539 anchor used in the leaderboard. The policy ordering becomes stable by B=100 trajectories per policy, and subsequent estimates remain within approximately ±25 Elo.

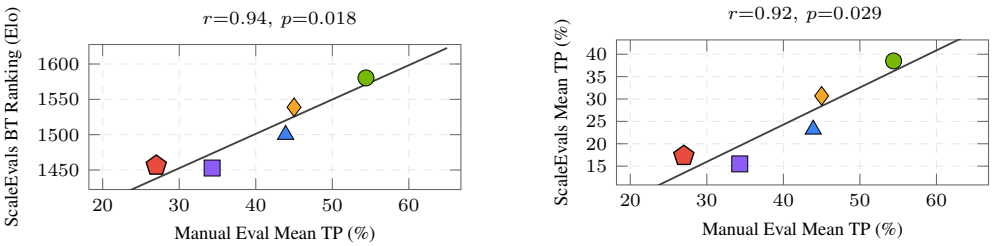


Figure 5: **ScaleEvals task-progress signal agrees with the held-out manual suite.** Reset-free ScaleEvals plotted against the held-out manual evaluation suite (n=40 per policy). *Left*: manual mean TP versus ScaleEvals BT rank (r=0.94, p=0.018). *Right*: manual mean TP versus raw ScaleEvals mean TP (r=0.92, p=0.029). Reset-free measurements preserve the relative policy signal of reset-required evaluation while using far broader task coverage. Task-level manual progress and initial frames are reported in Table 7.

preserve the same relative ordering signal across skill tags and divergence due to coverage differences.

Cross-embodiment transfer (YAM). To test whether SCALEEVALS generalizes beyond the DROID Franka setup, we run the same reset-free protocol on the YAM bimanual robot, collecting 250 rollouts per policy across four anonymized policies (Policy A–D) under continuous 4-policy alternation. Figure 7 shows that on YAM the ScaleEvals Bradley–Terry rating and mean task progress

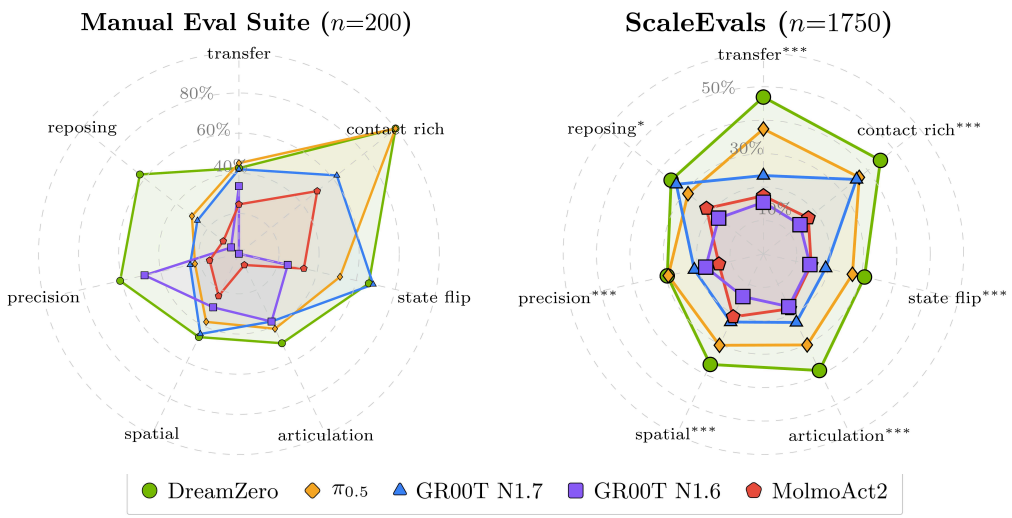


Figure 6: **Skill-tag-conditioned capability diagnostics.** Per-tag mean task-progress profiles for each policy, from reset-free ScaleEvals and the held-out manual suite (up to 40 rollouts per policy). The panels use different radial ranges, shown on their grids; compare policy ordering within each panel rather than polygon area across panels. The tag decomposition localizes *where* policies differ, while coverage gaps in the small manual suite appear as axes the reset-free stream fills in.

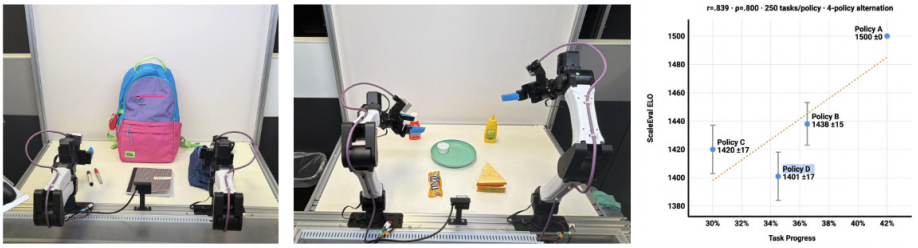


Figure 7: **Cross-embodiment evaluation on the YAM bimanual platform.** *Left and center*: two views of the YAM dual-arm tabletop setup. *Right*: ScaleEvals Bradley–Terry rating versus mean task progress for four anonymized policies (Policy A–D), with 250 rollouts per policy under four-policy alternation. Their correlation (Pearson r=0.839, Spearman ρ=0.800) supports internal consistency of the reset-free protocol on a second embodiment; YAM does not include an independent manual-evaluation baseline.

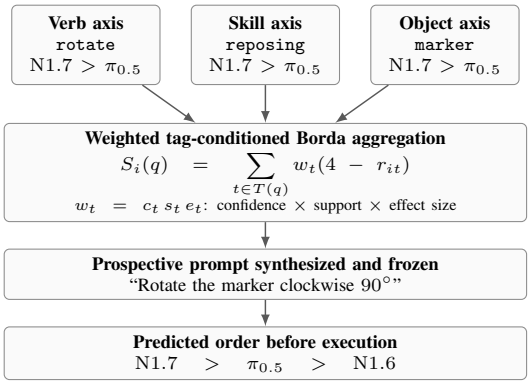
remain strongly correlated (Pearson r=0.839, Spearman ρ=0.800), supporting feasibility of the same protocol on a second embodiment.

5 Discussion

Agreement with manual evaluation. Figures 4 and 5 show that the Bradley–Terry (BT) ordering stabilizes by approximately 100 rollouts per policy and correlates strongly with manual mean task progress (r=0.94, p=0.018); raw ScaleEvals mean task progress shows similar agreement (r=0.92, p=0.029). The sole rank disagreement is MolmoAct2 versus GR00T N1.6. We hypothesize that the fixed manual suite over-weights skill axes on which N1.6 is stronger, whereas ScaleEvals samples broader skill-tag coverage (Figure 6).

Capability-conditioned prediction. The skill-tag profiles in Figure 6 localize capability differences hidden by aggregate scores. We therefore compose prompts from BT-ranked motion verbs, skill tags, and primary objects and predict their policy ordering before execution. This prediction is nontrivial because $\pi_{0.5}$ and GR00T N1.7 are adjacent in aggregate BT rank, yet reverse order

Recomposing behavioral evidence prospectively



Prospective physical validation

Family	Predicted order	N	Exact	$\bar{\rho}$
Place	$\pi_{0.5} >$ N1.7 $>$ N1.6	5	5/5	0.974
Rotate	N1.7 $> \pi_{0.5} >$ N1.6	5	4/5	0.800
Overall	capability-conditioned	**10**	**9/10**	**0.887**

Figure 8: **Tag recomposition enables prospective policy-order prediction.** *Left*: in a successful rotation example, the predictor recomposes verb-, skill-, and object-conditioned BT orders using confidence-, support-, and ANOVA-weighted Borda aggregation. The resulting prompt and predicted order are frozen before execution and then validated physically. *Right*: 9/10 exact orders over 240 rollouts (96% pairwise concordance). Appendix Table 6 reports task-level results and Appendix B.3 analyzes the Red 5 discrepancy.

between the placement and rotation families. Across 10 prompts and 240 physical rollouts, Table 6 reports mean Spearman ρ=0.887, 90% exact-order accuracy, and 96% pairwise concordance. These results show that the capability axes can be reused rather than merely described: we refer to this operational reuse as *behavioral compositionality*. The evidence is produced mostly autonomously through task generation, rollout grading, matching, and aggregation; humans configure the rig, oversee safety, and audit outputs.

6 Limitations

VLM perception and rubric quality. AutoGrader depends on VLM perception and rubric specificity; occlusion, ambiguous states, and underspecified rubrics can confound progress, with errors tending to over-credit. Failures concentrate in articulation and rotation tasks, where depth ordering, contact, or final object state can be ambiguous. AutoGrader achieves 90.7% criterion accuracy (86.2% macro F1, 66.7% precision, 94.5% recall), so periodic auditing remains important. Human correction preserves ordering (Pearson r=0.99, Spearman ρ=1.00; Figure 16), while proprietary VLMs limit exact reproducibility.

Scene drift, task-stream noise, and matching. Continuous play makes future tasks endogenous and difficulty sensitive to pose, occlusion, and reachability; string uniqueness also does not ensure semantic diversity. Matching and 530 interleaved rollouts reduce but do not remove this dependence. Results remain deployment-specific; YAM lacks a manual baseline and diversity statistics are descriptive. Moderate drift is useful as a robustness stress test, but can confound fine-grained comparisons under sparse tag coverage. Accordingly, SCALEEVALS complements controlled suites: diagnostics can screen checkpoints and prioritize manual trials, but controlled initial-state tests remain necessary.

Prediction scope. Behavioral compositionality is validated prospectively on ten prompts and three target policies. Establishing its breadth requires more prompt families, embodiments, and laboratories; composed predictions should prioritize rather than replace controlled physical validation.

Acknowledgments

We thank Mengda Xu, Ryan Punamiya, Kaiyuan Zheng, Jonathan Tremblay, Yu Qi, and Tavish Chen for feedback and valuable discussions, as well as Danyi Chen, Jeremy Chimienti, Mimoza Huynh, Nick Aguilar, Zizheng Wang, Amanpreet Singh, Cristaldo Campos, Gloria Whang, Leilee Naderi, Alec Nagal, Anshuman Sahu, Amy Nguyen, Caleb Geballe, and Matin Nikoui.

References

[1] 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. 5 2024. URL http://arxiv.org/abs/2405.05941.

[2] 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.

[3] Z. Zhou, P. Atreya, Y. L. Tan, K. Pertsch, and S. Levine. Autoeval: Autonomous evaluation of generalist robot manipulation policies in the real world. 4 2025. URL http://arxiv.org/abs/2503.24278.

[4] I. Fang, J. Zhang, S. Tong, and C. Feng. From intention to execution: Probing the generalization boundaries of vision-language-action models. 6 2025. URL http://arxiv.org/abs/2506.09930.

[5] 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. 12 2025. URL http://arxiv.org/abs/2512.16881.

[6] Y. Guo, L. X. Shi, J. Chen, and C. Finn. Ctrl-world: A controllable generative world model for robot manipulation. 3 2026. URL http://arxiv.org/abs/2510.10125.

[7] Y. Wang, R. Syed, F. Wu, M. Zhang, A. Onol, J. Barreiros, H. Nayyeri, T. Dear, H. Zhang, and Y. Li. Interactive world simulator for robot policy training and evaluation. 3 2026. URL http://arxiv.org/abs/2603.08546.

[8] Genesis Team. Genesis: A generative and universal physics engine for robotics and beyond, December 2024. URL https://github.com/Genesis-Embodied-AI/Genesis.

[9] Y. Hu, F. Lin, P. Sheng, C. Wen, J. You, and Y. Gao. Data scaling laws in imitation learning for robotic manipulation. 10 2025. URL http://arxiv.org/abs/2410.18647.

[10] T. Yin, Z. Mei, Z. Zheng, M. Yamane, D. Wang, J. Sceats, S. M. Bateman, L. Zha, A. Badithela, O. Shorinwa, and A. Majumdar. Playworld: Learning robot world models from autonomous play. 3 2026. URL http://arxiv.org/abs/2603.09030.

[11] Z. Zhou, P. Atreya, A. Lee, H. Walke, O. Mees, and S. Levine. Autonomous improvement of instruction following skills via foundation models. 10 2024. URL http://arxiv.org/abs/2407.20635.

[12] R. Punamiya, S. Kareer, Z. Liu, J. Citron, R.-Z. Qiu, X. Cai, A. Gavryushin, J. Chen, D. Liconti, L. Y. Zhu, P. Aphiwetsa, B. Li, A. Cheluva, P. Kuppili, Y. Liu, D. Patel, A. Gao, H.-Y. Chung, R. Co, R. Zbizika, J. Liu, X. Xu, H. Xiong, G. Chen, S. Oliani, C. Yang, X. Wang, J. Fort, R. Newcombe, J. Gao, J. Chong, G. Matsuda, A. Doriwala, M. Pollefeys, R. Katzschmann, X. Wang, S. Song, J. Hoffman, and D. Xu. Egoverse: An egocentric human dataset for robot learning from around the world. 4 2026.

[13] R. Zheng, D. Niu, Y. Xie, J. Wang, M. Xu, Y. Jiang, F. Castañeda, F. Hu, Y. L. Tan, L. Fu, T. Darrell, F. Huang, Y. Zhu, D. Xu, and L. Fan. Egoscale: Scaling dexterous manipulation with diverse egocentric human data. 2 2026.

[14] M. Pan, S. Feng, Q. Zhang, X. Li, J. Song, C. Qu, Y. Wang, C. Li, Z. Xiong, Z. Chen, Y. Liu, and J. Luo. Sop: A scalable online post-training system for vision-language-action models. 1 2026. URL http://arxiv.org/abs/2601.03044.

[15] W. Liang, S. Wang, H.-J. Wang, O. Bastani, Y. J. Ma, and D. Jayaraman. Tether: Autonomous functional play with correspondence-driven trajectory warping. 3 2026. URL http://arxiv.org/abs/2603.03278.

[16] R. ElMallah, K. Chhajer, and C.-G. Lee. Score the steps, not just the goal: Vlm-based subgoal evaluation for robotic manipulation. 9 2025. URL http://arxiv.org/abs/2509.19524.

[17] P. Atreya, K. Pertsch, T. Lee, M. J. Kim, A. Jain, A. Kuramshin, C. Eppner, C. Neary, E. Hu, F. Ramos, J. Tremblay, K. Arora, K. Ellis, L. Macesanu, M. T. Villasevil, M. Leonard, M. Cho, O. Aslan, S. Dass, J. Wang, W. Reger, X. Yuan, X. Yang, A. Gupta, D. Jayaraman, G. Berseth, K. Daniilidis, R. Martin-Martin, Y. Lee, P. Liang, C. Finn, and S. Levine. Roboarena: Distributed real-world evaluation of generalist robot policies. 11 2025. URL http://arxiv.org/abs/2506.18123.

[18] A. Yakefu, B. Xie, C. Xu, E. Zhang, E. Zhou, F. Jia, H. Yang, H. Fan, H. Zhang, H. Peng, J. Tan, J. Huang, K. Liu, K. Liu, K. Gu, Q. Zhang, R. Zhang, S. Huang, S. Cheng, S. Liu, T. Wang, T. Wang, W. Sun, W. Tang, Y. Wei, Y. Chen, Y. Gui, Y. Zhao, Y. Ma, Y. Wei, Y. Yang, Y. Guo, Z. Chen, Z. Du, Z. Zhang, Z. Liu, and Z. Yan. Robochallenge: Large-scale real-robot evaluation of embodied policies. 10 2025. URL http://arxiv.org/abs/2510.17950.

[19] R. A. Bradley and M. E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

[20] A. Liusie, V. Raina, Y. Fathullah, and M. Gales. Efficient llm comparative assessment: a product of experts framework for pairwise comparisons. 11 2024. URL http://arxiv.org/abs/2405.05894.

[21] K. Sun and R. Wang. Breaking myths in llm scaling and emergent abilities with a comprehensive statistical analysis. *Neurocomputing*, 670:132542, 3 2026. ISSN 09252312. doi: 10.1016/j.neucom.2025.132542.

[22] D. Snyder, A. J. Hancock, A. Badithela, E. Dixon, P. Miller, R. A. Ambrus, A. Majumdar, M. Itkina, and H. Nishimura. Is your imitation learning policy better than mine? policy comparison with near-optimal stopping. 6 2025. URL http://arxiv.org/abs/2503.10966.

[23] P. 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.

[24] NVIDIA, J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. J. Fan, Y. Fang, D. Fox, F. Hu, S. Huang, J. Jang, Z. Jiang, J. Kautz, K. Kundalia, L. Lao, Z. Li, Z. Lin, K. Lin, G. Liu, E. Llontop, L. Magne, A. Mandlekar, A. Narayan, S. Nasiriany, S. Reed, Y. L. Tan, G. Wang, Z. Wang, J. Wang, Q. Wang, J. Xiang, Y. Xie, Y. Xu, Z. Xu, S. Ye, Z. Yu, A. Zhang, H. Zhang, Y. Zhao, R. Zheng, and Y. Zhu. Gr00t n1: An open foundation model for generalist humanoid robots, 2025. URL https://arxiv.org/abs/2503.14734.

[25] S. Ye, Y. Ge, K. Zheng, S. Gao, S. Yu, G. Kurian, S. Indupuru, Y. L. Tan, C. Zhu, J. Xiang, A. Malik, K. Lee, W. Liang, N. Ranawaka, J. Gu, Y. Xu, G. Wang, F. Hu, A. Narayan, J. Bjorck, J. Wang, G. Kim, D. Niu, R. Zheng, Y. Xie, J. Wu, Q. Wang, R. Julian, D. Xu, Y. Du, Y. Chebotar, S. Reed, J. Kautz, Y. Zhu, L. J. Fan, and J. Jang. World action models are zero-shot policies. 2 2026. URL http://arxiv.org/abs/2602.15922.

[26] J. Duan, D. Clay, S. Wang, S. Liu, W. Huang, X. Fan, W.-C. Tsai, S. Chen, Y. R. Wang, S. Xing, 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. Technical report, Allen Institute for AI, 2026.

[27] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, P. D. Fagan, J. Hejna, M. Itkina, M. Lepert, Y. J. Ma, P. T. Miller, J. Wu, S. Belkhale, S. Dass, H. Ha, A. Jain, A. Lee, Y. Lee, M. Memmel, S. Park, I. Radosavovic, K. Wang, A. Zhan, K. Black, C. Chi, K. B. Hatch, S. Lin, J. Lu, J. Mercat, A. Rehman, P. R. Sanketi, A. Sharma, C. Simpson, Q. Vuong, H. R. Walke, B. Wulfe, T. Xiao, J. H. Yang, A. Yavary, T. Z. Zhao, C. Agia, R. Baijal, M. G. Castro, D. Chen, Q. Chen, T. Chung, J. Drake, E. P. Foster, J. Gao, V. Guizilini, D. A. Herrera, M. Heo, K. Hsu, J. Hu, M. Z. Irshad, D. Jackson, C. Le, Y. Li, K. Lin, R. Lin, Z. Ma, A. Maddukuri, S. Mirchandani, D. Morton, T. Nguyen, A. O'Neill, R. Scalise, D. Seale, V. Son, S. Tian, E. Tran, A. E. Wang, Y. Wu, A. Xie, J. Yang, P. Yin, Y. Zhang, O. Bastani, G. Berseth, J. Bohg, K. Goldberg, A. Gupta, A. Gupta, D. Jayaraman, J. J. Lim, J. Malik, R. Martín-Martín, S. Ramamoorthy, D. Sadigh, S. Song, J. Wu, M. C. Yip, Y. Zhu, T. Kollar, S. Levine, and C. Finn. Droid: A large-scale in-the-wild robot manipulation dataset. 4 2025. URL http://arxiv.org/abs/2403.12945.

Appendix Overview

Appendix A documents AutoPrompter examples, difficulty calibration, accepted task coverage, scene-aware tagging, and the full task-generation prompt. Appendix B reports capability diagnostics, sample-efficiency and throughput accounting, prospective prediction details, and the held-out manual suite. Appendix C contains the AutoGrader audit, balanced-leaderboard calibration, and extended capability profiles.

A AutoPrompter Few-Shot Task Generation

AutoPrompter generates instructions from the current scene while preserving enough structure for grading and comparison. Rather than relying on unconstrained prompting, we ground generation with a curated few-shot library of tasks. Each few-shot example specifies the task language, required scene objects, difficulty annotation, and a stepwise grading rubric expressed as yes/no visual checks. Table 2 shows representative examples from this library, and Table 3 summarizes how the examples calibrate the difficulty labels used downstream for matching.

A.1 Few-Shot Examples, Difficulty, and Rubrics

The few-shot library spans the canonical capability tags used in our analysis: transfer, spatial reasoning, precision, reposing, state flip, contact-rich manipulation, and articulation. These examples discourage low-quality prompt distributions dominated by simple pick-and-place, calibrate task difficulty, and demonstrate how generated instructions should be paired with decomposed grading rubrics. Skill tags are assigned after a task is accepted, using the generated instruction and scene context; the tags are used for downstream capability analysis rather than as hard labels during task generation.

Example instruction	Skill / role	Difficulty role
Put the banana on the green plate	transfer	easy single-step placement
Center the cup on the plate	spatial	medium placement with alignment
Stack the red cube on the yellow cube	precision	medium tight-tolerance placement
Turn the apple so its stem points up	reposing	hard orientation control
Push the yellow cube into the black square	contact-rich	easy non-prehensile motion
Open the drawer fully	articulation	easy articulated-object interaction
Open drawer, place marker inside, close it, then stack cubes	composite	expert multi-skill sequence

Table 2: Representative AutoPrompter few-shot examples. The library covers the skill tags used for downstream capability analysis and gives the model concrete examples of how task difficulty and rubric decomposition should be expressed.

AutoPrompter outputs a 1–10 difficulty score with a corresponding label; Table 3 summarizes the calibration used for matching and aggregation.

Difficulty	Score range	Few-shot calibration
Easy	1.0–4.0	One direct manipulation goal, e.g., transfer or simple push
Medium	4.1–6.0	Spatial, precision, or sustained-contact requirement
Hard	6.1–8.0	Three-step composition, articulation, or fragile orientation control
Expert	8.1–10.0	Four or more ordered subgoals or mixed-skill sequence

Table 3: AutoPrompter difficulty calibration. Difficulty labels are grounded by few-shot examples and used for matching and aggregation rather than as a standalone measure of task quality.

For every generated instruction, AutoPrompter also produces a milestone rubric. Rubric milestones are written as yes/no visual questions and weighted to sum to one. For example, a transfer task may ask whether the object was lifted and whether it reached the target receptacle, while a long-horizon task allocates credit across each subgoal. This makes generated tasks compatible with AutoGrader and avoids reducing evaluation to binary success.

A.2 Observed Output Distribution

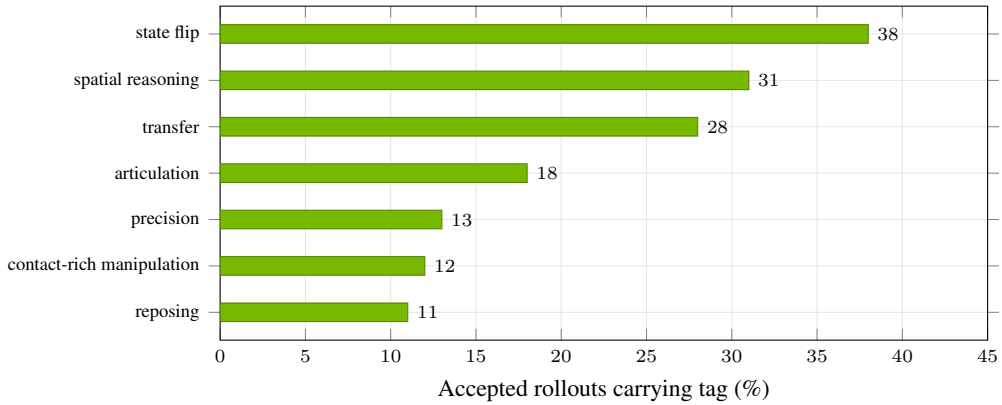


Figure 9: **Observed AutoPrompter skill-tag distribution.** Across 2,486 accepted DROID rollouts, generated prompts contain 26 distinct action verbs and reference 32 objects. Tags are multi-label, so percentages sum to more than 100%. The distribution documents task coverage but is not a standardized semantic-diversity metric.

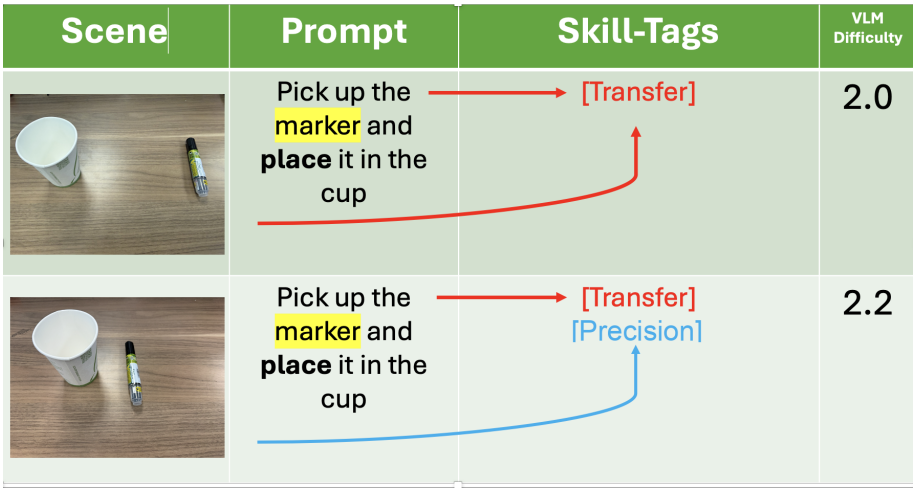


Figure 10: **Scene-aware skill tagging and difficulty.** Both prompts require object transfer. In the second scene, the marker-to-cup alignment adds a precision demand, illustrating that skill tags depend on both instruction semantics and scene context. VLM-estimated difficulty, action verbs, and primary objects provide additional metadata for matched Bradley–Terry rankings and prospective prompt composition (Table 6).

A.3 Prompt Template

The AutoPrompter prompt is instantiated with scene-specific objects, recent task history, and few-shot rubric examples. We reproduce the compact template below.

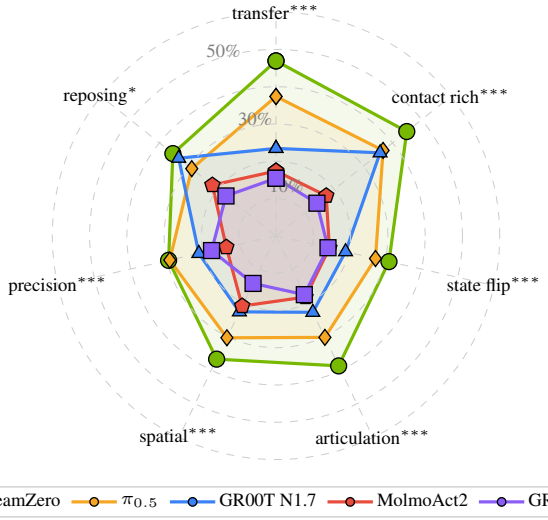
AutoPrompter task-generation prompt

```
You are a robot task generator for a single-arm tabletop manipulation robot (Franka FR3).
AVAILABLE OBJECTS IN THE SCENE (with notes):
{object_details_text}
OBJECTS SAFE TO LIFT:
{safe_to_lift_json}
HEAVY OBJECTS (DO NOT ATTEMPT TO LIFT - robot payload limit is 0.5kg):
{heavy_objects_json}
For heavy objects, you can ONLY: push, slide, or interact with items inside/on top of them.
EXAMPLE TASKS (for reference on style and complexity):
{example_task_text}
PREVIOUSLY ATTEMPTED TASKS (DO NOT REPEAT THESE - choose something DIFFERENT):
```

```
{previously_attempted_tasks}
IMPORTANT: The tasks above have already been attempted in this session. You MUST generate a
     DIFFERENT task.
Even if the scene looks similar, pick a different object or a different action. Be creative and
     vary your choices.
ACTION VARIETY - Use diverse verbs and motions. Consider these actions:
  - MANIPULATION: pick, place, grasp, release, lift, lower, hold
  - MOVEMENT: push, slide, drag, move, shift, reposition, transfer
  - ROTATION: rotate, turn, flip, spin, orient, tilt, tip
  - ARRANGEMENT: stack, align, center, match, group, separate, spread
  - CONTAINER: put into, take out of, deposit, extract, scoop
  - ORGANIZATION: clear, gather, arrange, sort, line up
FORBIDDEN ACTIONS (the robot physically CANNOT do these - NEVER generate tasks involving them):
  - cutting, slicing, chopping (no blade/knife capability)
  - pouring, filling, emptying liquids (no liquid handling)
  - throwing, tossing, catching (no ballistic control)
  - folding, unfolding (no deformable-object manipulation)
  - tying, untying, knotting (no rope/string manipulation)
  - pressing buttons (too precise for the gripper)
  - using two hands (single-arm robot)
Note: object NAMES like "watermelon slice" are fine. The word "slice" in a name is not a slicing
     action.
RULES FOR TASK GENERATION:
1. ONLY use objects that you can ACTUALLY SEE in the camera image.
2. Use the EXACT object names from the AVAILABLE OBJECTS list. Do not add adjectives, modifiers,
     or synonyms.
3. NEVER attempt to lift heavy objects marked as unsafe. Only push, slide, or interact with items
     inside/on top of them.
4. Tasks must be achievable by a single-arm robot on a tabletop.
5. Tasks should be one clear sentence in natural language.
6. Task timeout MUST match the difficulty SCORE:
   score <= 4  -> 30s
   score <= 6  -> 60s
   score <= 8  -> 90s
   score <= 10 -> 120s
7. Avoid multi-step tasks that require more than 2-3 actions unless the requested difficulty is
     hard or expert.
8. Avoid tasks that would require two hands or complex tool use.
9. Be creative with actions. Do not generate only pick-and-place tasks.
10. Do not repeat any previously attempted task.
EXAMPLE TASK RUBRICS:
{few_shot_rubric_examples}
DIFFICULTY CLASSIFICATION (you MUST classify the task on a 1-10 scale, NOT 0-1):
- easy (score 1.0-4.0): Single-step pick-and-place, simple slides/pushes, basic object
     identification.
- medium (score 4.1-6.0): Spatial reasoning, alignment ("center", "between"), precision
     insertion/stacking, sustained contact-rich manipulation.
- hard (score 6.1-8.0): Multi-step tasks with 3 or more actions, mixed-skill chains,
     articulation, dependent sequencing.
- expert (score 8.1-10.0): Long ordered sequences, complex multi-step tasks with precision
     subtasks, tool-like interaction.
The score MUST be between 1 and 10. Do NOT use a 0-1 scale. The label MUST match the score band.
GRADING RUBRIC GENERATION:
You must also generate a grading rubric with 2-4 milestones that an autograder can use to score
     the task.
Each milestone should award partial credit for progress toward the goal.
Every sub-task or milestone description must be phrased as a yes/no question, for example:
"Did the robot pick up the banana?"
"Did the robot place the fruit on the plate?"
"Did the drawer end up mostly open?"
Milestones should be ordered from easiest to hardest, with weights summing to 1.0.
RESPOND IN JSON FORMAT:
{
    "language": "<task description in natural language>",
    "timeout": <MUST match difficulty SCORE: score<=4 -> 30, score<=6 -> 60, score<=8 -> 90,
         score<=10 -> 120>,
    "objects_used": ["<exact object names from the AVAILABLE OBJECTS list that this task
         involves>"],
    "reasoning": "<brief explanation of why this task is appropriate given the current scene>",
    "difficulty": {
        "label": "<easy|medium|hard|expert>",
        "score": <1.0-10.0 within the difficulty range>,
        "reasoning": "<why this difficulty level is appropriate>"
    },
    "grading_rubric": {
        "type": "milestone_stepwise",
        "max_score": 1.0,
        "milestones": [
            {
                "weight": <0.0-1.0>,
                "description": "<yes/no question a grader or autograder answers to award this
                     credit>"
            },
            {
                "weight": <0.0-1.0>,
                "description": "<yes/no question a grader or autograder answers to award this
                     credit>"
            }
        ]
    }
}
```

B Capability Diagnostics and Robustness

B.1 Skill-Conditioned Capability Analysis



(a) Skill radar per policy. Spoke labels carry FDR-significance stars only; effect sizes η^2 are in the table below.

Skill tag	n	F	p	η^2	DZ	$\pi_{0.5}$	N1.7	MA2	N1.6
transfer†	707	22.0	<0.001	0.11	**47**	37	23	17	15
contact_rich†	298	9.0	<0.001	0.11	**45**	37	36	17	14
state_flip†	952	17.2	<0.001	0.07	**31**	27	19	15	14
long_horizon†	996	19.5	<0.001	0.07	**32**	25	19	14	15
articulation†	448	7.9	<0.001	0.07	**39**	30	23	18	18
spatial†	761	13.3	<0.001	0.07	**37**	30	23	21	14
precision†	333	5.4	<0.001	0.06	**30**	29	21	14	18
reposing†	272	3.3	0.011	0.05	**35**	29	33	22	17

(b) Tag-stratified one-way ANOVA, skill dimension + long_horizon (matched-only, 5 policies). † Significant under Benjamini–Hochberg FDR at α=0.05. Bold marks the per-row leader.

Figure 11: **Skill profile and per-tag ANOVA. (a)** Per-skill mean task progress on matched-pair tasks across all five policies. Spokes are sorted by one-way-ANOVA effect size (η^2, policy across TP) descending, starting from the top and going clockwise. Radii are raw mean TP in percent (radial gridlines at 10, 20, 30, 40, 50, 60%). Significance stars use Benjamini–Hochberg-adjusted q-values ($^*q<0.05$, $^{**}q<0.01$, $^{***}q<0.001$). Color palette matches the PlayArena dashboard (positional by BT rank). **(b)** Same data in tabular form, augmented with the binary `longHorizon` dimension; cells are per-policy mean TP in %. DreamZero now leads on all eight tags (narrowest margins: precision 30 vs $\pi_{0.5}$ 29, reposing 35 vs GR00T N1.7 33); the previously reported precision/reposing inversions disappeared as more matched comparisons accrued. *Caveat (radar).* Radar charts encode value as radius but readers perceive area ($\propto r^2$), which inflates apparent differences; the table is the quantitative source of record.

B.2 Sample Efficiency and Operational Throughput

Policy	Full n	BT at B=100 ↑	TP %	SR %
DreamZero	385	1594 ± 15	39.1	14.0
$\pi_{0.5}$ (anchor)	681	1524 ± 30	30.0	8.0
GR00T N1.7	601	1500 ± 5	23.3	4.0
MolmoAct2	440	1470 ± 28	17.5	2.5
GR00T N1.6	379	1453 ± 25	14.7	0.0

Table 4: **Minimum-sample stress test: balanced subset at B=100 trajectories per policy.** The same two-pass balancing protocol as the equal-N, B=350 analysis trims each policy to B=100, the rank lock-in identified in Figure 4. Values are K=40 bootstrap medians and the BT $\pm$ column reports median Hessian SE on the Elo scale. The five-way ordering is unchanged; relative to B=350, BT shifts are at most 15 Elo and TP/SR shifts are below one percentage point.

Evaluation method	Net rollouts/h/station	Active human time/100	Operator scaling
Manual reset-required baseline	14.8	4 h	1:1
ScaleEvals	**23.0**	**5 min**	**1:N**

Table 5: **End-to-end operational throughput and human involvement.** Net throughput includes AutoPrompter VLM calls, failed task-generation retries, safety and visibility checks, policy execution, and rollout time. The manual baseline uses one operator per station for scene resets and grading. ScaleEvals reduces active human involvement from 4 hours to 5 minutes per 100 rollouts (98%) and recorded five human interventions over the additional 530-rollout temporal-drift evaluation.

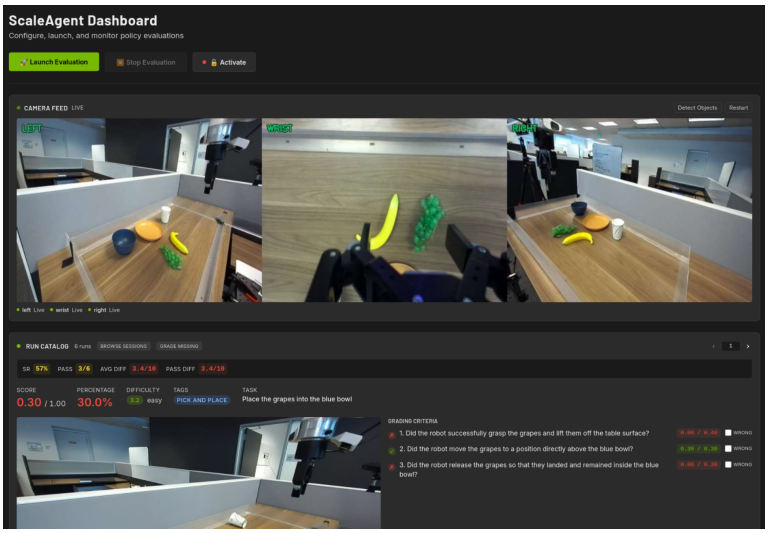


Figure 12: **ScaleEvals operator dashboard.** The WebUI configures and launches evaluations, displays live camera observations, tracks generated tasks and rollout history, and exposes AutoGrader rubric-level scores for auditing and correction. The displayed camera panes show the original monitored rollout views.

B.3 Prospective Prediction Details

For each action-verb, skill-tag, and primary-object value, we fit a tag-conditioned BT ranking from matched rollouts and retain values for which all three target policies have sufficient evidence. The tag-conditioned predictor assigns Borda points $3, 2, 1$ to each ordering and computes

$$S_i(q) = \sum_{t \in T(q)} c_t \, s_t \, e_t \, (4 - r_{it}), \qquad s_t = \min(n_t/10, 1), \quad e_t = \begin{cases} \max(\eta_t^2, 0.01), & \eta_t^2 \text{ available,} \\ 1, & \text{otherwise,} \end{cases}$$

where $c_t \in \{1.0, 0.7, 0.3, 0\}$ is the confidence weight, n_t is the minimum target-policy pair count, η_t^2 is the tag's ANOVA effect size, and r_{it} is policy i's rank for tag value t. The resulting score orders candidate tag compositions; a prompt template then realizes the selected composition, and its predicted order is frozen before execution. Pairwise tag tests are Benjamini–Hochberg corrected globally and reported as reliability diagnostics, not used as a hard inclusion gate.

The single miss, Red 5, contains conflicting evidence: its rotate and reposing components favor GR00T N1.7, while the object-conditioned ordering favors $\pi_{0.5}$. The object result has only 10–15 comparisons and does not pass BH-FDR, so it is treated as a low-confidence conflict rather than a statistically established explanation. All ten prompts and orderings are fixed before physical execution; the subsequent 240 rollouts are used only for prospective validation.

Prompt	**Tags**	ρ	**Exact Order**
Place suite — predicted $\pi_{0.5} >$ N1.7 $>$ N1.6			
Place banana on orange plate	`(place; transfer; banana)`	1.00	✓
Slide blue bowl to orange plate	`(slide; contact_rich; blue bowl)`	1.00	✓
Place grapes on blue bowl	`(place; transfer; grapes)`	1.00	✓
Place paper cup on orange plate	`(place; transfer; paper_cup)`	1.00	✓
Place blue bowl on orange plate	`(place; transfer; blue bowl)`	0.87	✓
Rotate suite — predicted N1.7 $> \pi_{0.5} >$ N1.6			
Rotate marker 90° CW	`(rotate; reposing; marker)`	1.00	✓
Rotate pink mug 90° CW	`(rotate; reposing; pink mug)`	1.00	✓
Rotate green plate 90° CW	`(rotate; reposing; green plate)`	1.00	✓
Rotate cup 90° CW	`(rotate; reposing; cup)`	1.00	✓
Rotate red 5 90° CW	`(rotate; reposing; Red 5)`	0.00	×

Summary: N=240 trajs (10 prompts × 8 rollouts × 3 policies); $\bar{\rho}$=0.887; exact-order hit = 90%; concordance = 96%.

Table 6: **Predictive validation of tag-composed prompts.** Each prompt combines action-verb, skill-tag, and primary-object evidence, shown in the **Tags** column in `teletype`. The predicted ordering (stated in each section header) is produced *before* the rig is run; we then execute N=8 episodes per cell and grade with the AutoGrader. Spearman ρ is the per-prompt rank correlation between predicted ranks and observed mean task-progress scores; **Exact Order** indicates whether the complete predicted three-policy order is observed. N1.7 and N1.6 denote GR00T N1.7 and GR00T N1.6, respectively.

B.4 Tag-Conditioned Statistics and Manual Evaluation

The held-out suite contains 20 diverse reset-required tasks with two initial frames per policy/task cell where available. Table 7 reports the task-level progress values and corresponding initial observations used by the main calibration and radar analyses.

Table 7: Manual task-progress results for the 20 diverse manipulation tasks underlying the calibration in Figure 5 and radar diagnostics in Figure 6. Each entry is the mean task progress over two trials with different initial frames. The missing DreamZero entry indicates that its two trials were unavailable, yielding 38 DreamZero and 198 total observed rollouts. Best per row in **bold**.

Task (Language Prompt)	Initial frame	GR00T N1.6	GR00T N1.7	$\pi_{0.5}$	DreamZero	MolmoAct 2
Put the banana on the blue plate		**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
Put the ‘red’ cube on ‘top’ of the yellow cube		0.75	0.50	0.25	**1.00**	0.25
Open the oven door, pickup the croissant, and place it in the oven		0.25	0.00	0.33	**0.49**	0.00
Pick up bread and place into toaster		**0.75**	0.25	0.25	0.62	0.00
Put the marker in the drawer and close the drawer		0.50	**0.75**	0.50	–	0.12
Flip cup		0.25	0.38	0.38	**0.50**	**0.50**
Pick up the pink cup and pour it into the blue plate		0.10	0.20	**0.60**	**0.60**	0.20
slide plate to left side of table		0.00	0.62	**1.00**	**1.00**	0.50
Center cup on plate		0.33	**0.66**	0.33	**0.66**	**0.66**
Cover the blue letter with the plate		0.00	0.00	**0.33**	0.00	0.00
Remove all objects from basket		0.00	0.50	**0.75**	0.38	0.62
Separate bowls and place grapes in red bowl		**0.25**	**0.25**	**0.25**	0.00	0.12
Put all objects into the box		0.60	0.60	**0.70**	0.50	0.00
Place cube on sandwich		**0.67**	0.49	0.00	**0.67**	0.00
Assemble pyramid		**0.25**	**0.25**	0.12	**0.25**	**0.25**
Disassemble pyramid		0.25	**0.75**	0.25	0.00	0.62
Fold the towel		0.25	**1.00**	0.66	0.83	0.17
Mix the vegetables in the bowl with the spoon		0.00	0.33	0.00	**0.67**	0.00
Assemble burger		0.17	0.00	0.00	**0.67**	0.00
Put the blocks in the matching hole		**0.50**	0.25	**0.50**	**0.50**	0.25
Mean Task Progress		0.34	0.44	0.41	**0.54**	0.26

C AutoGrader Reliability and Extended Evaluation

C.1 AutoGrader Audit

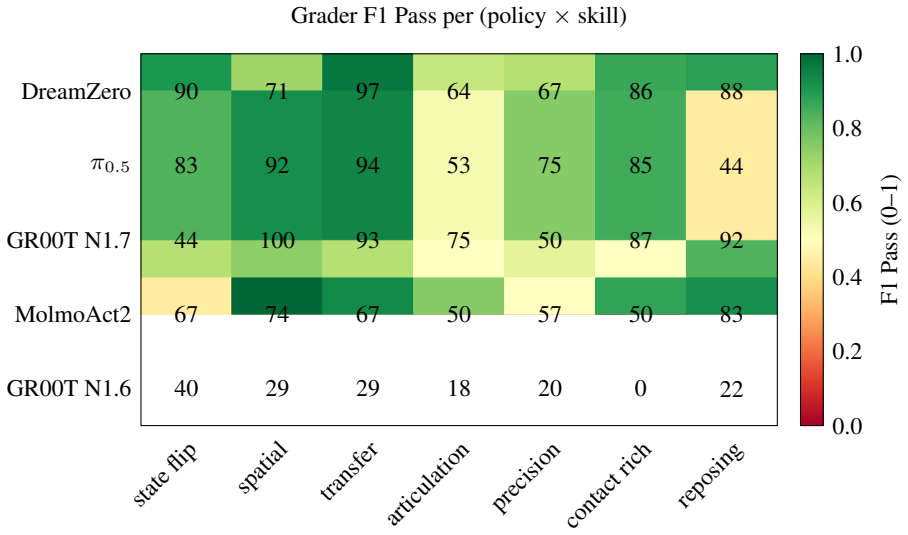


Figure 13: **Per-policy, per-skill grader F1 (pass class).** Values are %; color encodes F1 on a $[0, 1]$ scale (red–yellow–green).



Figure 14: Grader **Accuracy** of the VLM grader versus human review, per policy × skill tag ($N = 525$ audited rollouts, 15 per cell). Each cell shows the accuracy (bold), with the count of criteria the grader labelled incorrectly below. Cells are coloured by accuracy (red–yellow–green, 0–1).



Figure 15: Grader **Macro F1** of the VLM grader versus human review, per policy $\times$ skill tag ($N = 525$ audited rollouts, 15 per cell). Each cell shows the macro f1 (bold), with grader–human agreement below. Cells are coloured by macro f1 (red–yellow–green, 0–1).

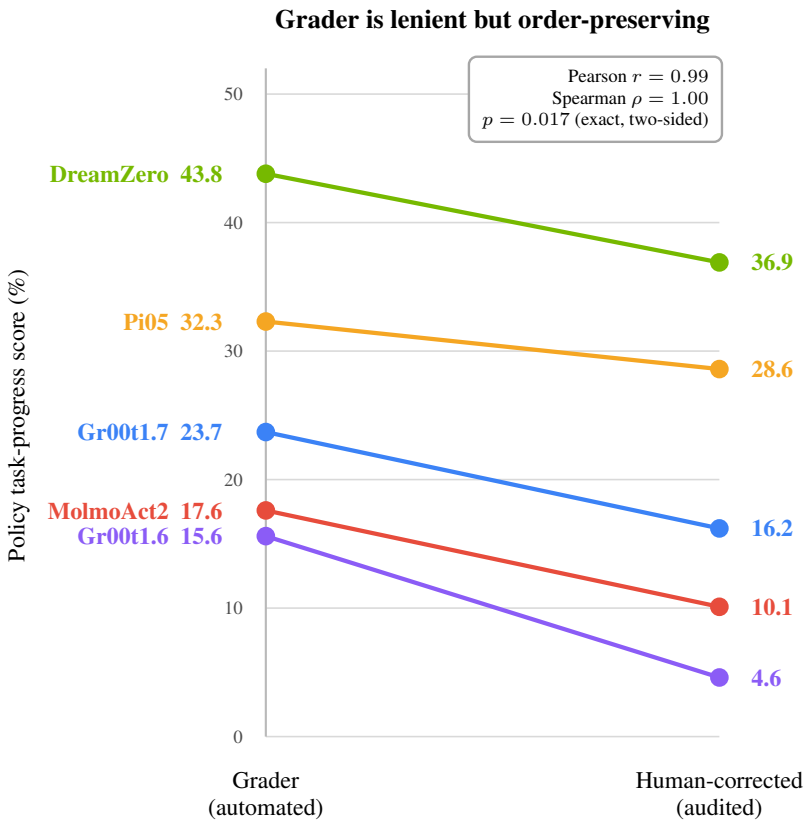


Figure 16: The VLM grader is lenient (every policy's task-progress score drops after human correction) yet order-preserving as no lines cross. Grader and human-corrected policy scores are near-perfectly correlated (Pearson $r = 0.99$, Spearman $\rho = 1.00$; exact two-sided permutation $p = 0.017$, $n = 5$ policies over 525 audited rollouts), so the automated grader recovers the same policy ranking as audited human review.

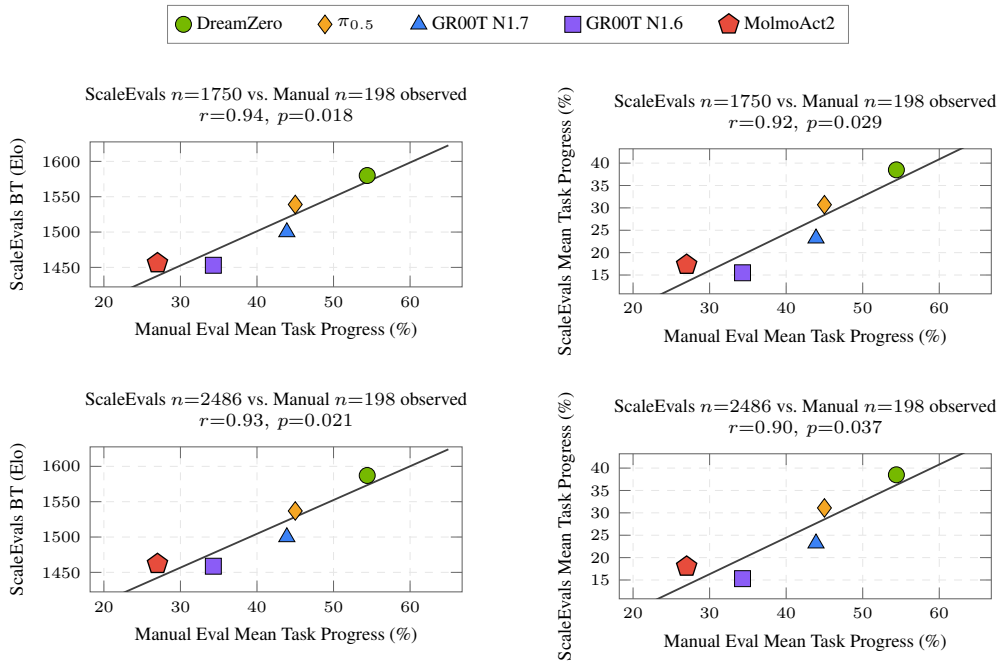


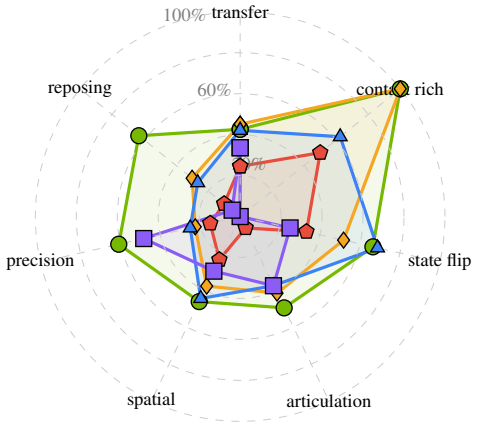
Figure 17: **ScaleEvals aligns with manual evaluation, at both data scales.** Per-policy ScaleEvals Bradley–Terry (*left column*) and raw mean task-progress (*right column*) plotted against the held-out manual eval suite's mean TP. The manual suite contains n=198 observed rollouts (40 reset-required rollouts for four policies and 38 for DreamZero), versus n=1750 ScaleEvals rollouts in the balanced subset (*top row*, 350 per policy) and n=2486 in the full corpus (*bottom row*); ScaleEvals therefore uses roughly $8.75\times$ and $12.4\times$ more rollouts than the manual suite, respectively. Pearson r with exact two-tailed p from the closed-form $t(\text{df}=3)$ CDF. All four cells clear α=0.05; the BT correlation is uniformly tighter than the raw-TP correlation on both data slices, indicating that the soft-win Bradley–Terry aggregation removes some of the per-policy idiosyncratic noise present in the raw mean-TP. Spearman ρ=0.90 in every panel; the only rank disagreement with the manual suite is MolmoAct2 $\leftrightarrow$ GR00T N1.6 (manual ranks N1.6 above MolmoAct2; ScaleEvals reverses the pair on both TP and BT).

C.2 Balanced Leaderboard and Calibration

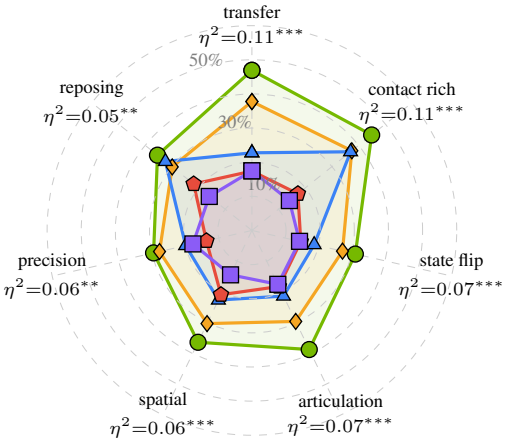
Policy	**Rank**	$\boldsymbol{\theta}$ (BT)	$\boldsymbol{\sigma_\theta}$	$\overline{\textbf{TP}}$ **(%)**	**SR (%)**	**Pairs**
DreamZero	1	1580	16	38.5	13.1	989
Pi05†	2	1539	0	30.7	7.7	748
Gr00tN1.7	3	1500	17	23.2	3.4	809
MolmoAct2	4	1456	16	17.3	2.6	950
Gr00tN1.6	5	1453	16	15.5	0.6	970

Table 8: **Bradley–Terry leaderboard on the balanced 1750 subset (n=350 rollouts per policy).** Soft wins use raw task-progress score, pair weights track the matched-tag count. † Pi05 is the BT anchor (σ_θ=0 by construction). $\overline{\text{TP}}$ is mean task progress across the 350 rollouts; SR is the perfect-rollout rate.

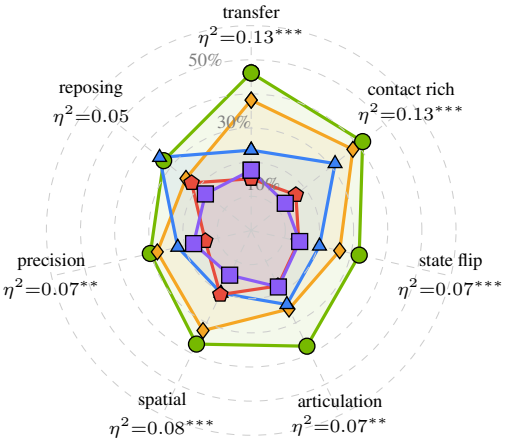
C.3 Extended Capability Profiles



(a) Manual eval — held-out reset-required suite (n=198 observed; 40 rollouts for four policies and 38 for DreamZero).



(b) ScaleEvals — full corpus (n=2486 rollouts).



(c) ScaleEvals — balanced subset (n=1750; 350/policy).



Figure 18: **Skill radar across three evaluation slices.** Per-skill mean task progress (%) for each policy, rendered as polar profiles. The manual panel (a) spans a $[0, 100]$% radial axis (its per-axis-normalized scores reach 100%; gridlines every 20%); the two ScaleEvals panels are normalized to $[0, 60]$% to resolve the AutoGrader task-progress spread (gridlines every 10%). Spoke positions are anchored to the (b) ScaleEvals full-corpus one-way-ANOVA η^2 ordering and held fixed across all three panels, so polygon shapes can be compared at the same angular position. a: held-out human-graded reset-required suite from the Manual Evaluation workbook (5–11 tasks per skill across all policies; *contact rich* has only 1 task per cell "slide plate to left side of table," so the $\{0, 50, 62.5, 100\}$ values shown are single-task outcomes). η^2 is omitted from a because at this n the per-skill ANOVA is underpowered (no skill clears BH-FDR at α=0.05). ScaleEvals AutoGrader scores on the unbalanced full corpus and on the per-tag-balanced subset; spoke labels carry each panel's own η^2 with Benjamini–Hochberg-adjusted significance stars ($^*q$<0.05, $^{**}q$<0.01, $^{***}q$<0.001). The radars demonstrate that trajectories per skill-tag should be balanced for improved coverage of generalizability.