OpenHLM: An Empirical Recipe for Whole-Body Humanoid Loco-Manipulation

Yingdong Hu[1,2,3*] **Haodong Zhu**[1*] **Boyuan Zheng**[1,2*] **Yihang Hu**[1,2*] **Tong Zhang**[1,2,3*]
Zunhao Chen[1] **Junming Zhao**[1] **Ruiqian Nai**[1] **Yang Gao**[1,2,3†]

[1] Tsinghua University [2] Shanghai Qi Zhi Institute [3] Spirit AI

https://openhlm-project.github.io/

Abstract: Whole-body humanoid loco-manipulation requires coordinating the robot's entire kinematic chain. However, most existing systems typically decouple the upper and lower bodies into separate controllers, limiting such coordination and yielding behaviors similar to those of a wheeled dual-arm platform. In this paper, we ask what it takes to build a whole-body native vision-language-action (VLA) model that maps language and pixels directly to all of the humanoid's degrees of freedom. We conduct a systematic empirical study organized as a roadmap of one-variable-at-a-time experiments across three phases: whole-body teleoperation, VLA model design, and heterogeneous co-training. Our study yields several intriguing findings: a joint-based whole-body teleoperation interface outperforms alternatives that only partially expose the humanoid's degrees of freedom; a VLA pretrained on static and wheeled dual-arm platforms transfers surprisingly well to a humanoid's full action space; and co-training with HuMI, the humanoid analog of UMI, extends the policy to new objects and instructions without additional whole-body teleoperation on those targets. Following this roadmap yields OpenHLM, an open-source recipe for whole-body humanoid loco-manipulation. In a challenging long-horizon task that spans a wide vertical range of the humanoid, OpenHLM outperforms two state-of-the-art humanoid VLA baselines (GR00T N1.6 and Ψ_0) using less than half the total demonstration time. Our code, training data, and model checkpoints are available at https://openhlm-project.github.io/.

Keywords: Humanoid Loco-Manipulation, Whole-Body Control, VLA

1 Introduction

Humans perform complex loco-manipulation by coordinating their entire body, e.g., pressing a pedal with the foot, or squatting to reach a low shelf. Humanoid robots share a similar kinematic chain and, in principle, the same potential. However, most existing humanoid systems decouple the upper and lower bodies into separate controllers [2, 3, 4, 5, 6, 7, 8]. Inverse kinematics drives the arms, a separate reinforcement-learning-trained controller drives the legs, and the two are stitched together through a navigation command and root-height signal. This formulation limits whole-body coordination in two ways. Visually, the motion is mechanical and unnatural. Functionally, the lower body serves only as a mobile base, rather than an active participant in manipulation, leaving the humanoid closely akin to a wheeled dual-arm platform. In light of this, a coordinated whole-body stack that reasons over the robot's entire kinematic chain has emerged as the path forward [9, 10], yet its design space remains largely unexplored.

A natural starting point is the two-level hierarchy that has recently emerged for such stacks [10]: a high-level vision-language-action (VLA) model [11] that maps language and pixels to whole-body

*Core contributors
†Corresponding author

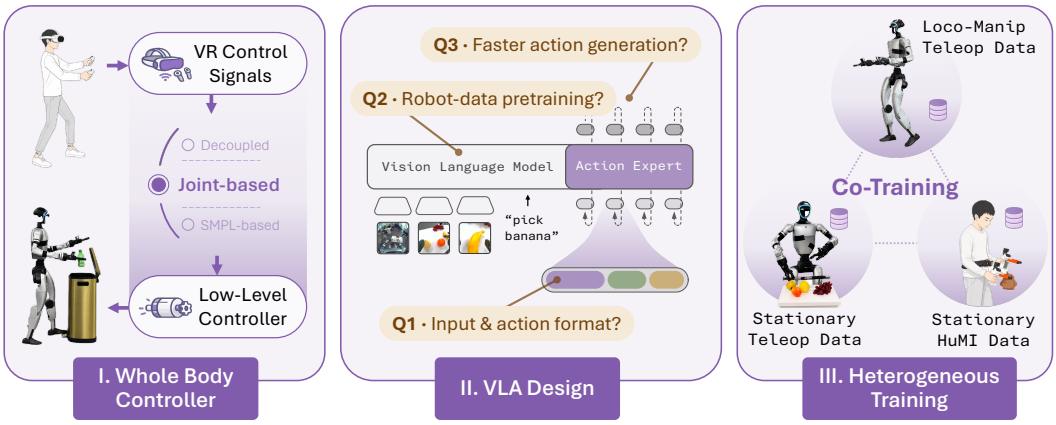


Figure 1: **Overview of `OpenHLM`.** A roadmap of controlled experiments in three phases: **(I)** we compare teleop interfaces for a low-level whole-body controller and adopt a joint-based interface; **(II)** we adapt a manipulation VLA to the humanoid's full action space along several design axes; **(III)** we extend the policy to new objects and instructions by co-training the full loco-manipulation data with stationary teleop or HuMI [1] demonstrations.

commands, and a low-level controller that tracks them [12, 13]. This decomposition raises three questions. The controller and the teleoperation interface built on it determine what demonstrations can be collected; how should they be designed? The VLA must handle a humanoid's full degrees of freedom, yet many widely used VLAs target static and wheeled dual-arm platforms [14, 15]; which adaptations actually matter? And once the pipeline is established, whole-body teleop is too expensive to scale to every new object and instruction; can cheaper data sources fill the gap? This paper answers these questions through an empirical study, organized as a roadmap of controlled, one-variable-at-a-time experiments in three phases (Fig. 1): (1) whole-body controller and teleoperation (§3.1), (2) VLA model design (§3.2), and (3) heterogeneous co-training (§3.3). Within each phase, ablations on subsets of our HLM-12 benchmark quantify how each design choice affects task progress, building up to a concrete recipe for whole-body humanoid loco-manipulation.

Following this roadmap yields `OpenHLM` (Open Humanoid Loco-Manipulation), the open-source realization of this recipe, which we will release to support future research. Along the way, we uncover several surprising findings; we highlight three here. First, the teleoperation interface matters: a joint-level whole-body interface outperforms common alternatives such as VR 3-point control that partially expose the humanoid's degrees of freedom. Second, despite the large embodiment gap, a VLA pretrained on static and wheeled dual-arm platforms transfers surprisingly well to a humanoid's whole-body action space; yet action MSE on held-out demonstrations is a poor proxy for real-world task progress. Third, co-training with cheaper manipulation-only sources, such as stationary teleop (feet planted, no locomotion) and HuMI [1] (the humanoid analog of UMI [16]), extends the policy to new objects and instructions without additional whole-body teleop on those targets.

Quantitatively, `OpenHLM` reaches 89% average task progress on the 8 training tasks of our HLM-12 benchmark; on the remaining 4 held-out tasks that whole-body teleop never covers, co-training with cheaper data lifts task progress from 33% to 87%, closing most of the gap to a 12-task oracle (94%). At the system level, on a long-horizon language-conditioned task spanning the humanoid's wide vertical workspace, `OpenHLM` outperforms two state-of-the-art humanoid VLAs (GR00T N1.6 [7] and Ψ_0 [8]) at less than half the demonstration time; both baselines exhibit weak grasping and fail to track language-specified targets, despite including humanoid data in their pretraining. To summarize, our main contributions are:

- **A systematic empirical study.** We explore the design space of whole-body humanoid loco-manipulation through controlled ablations across three phases (controller/teleoperation, VLA design, co-training), yielding a set of key findings about each design choice.
- **A concrete recipe for whole-body humanoid VLAs.** The study yields `OpenHLM`, an open-source recipe for VLAs that jointly control the humanoid's full action space. We will release

the full stack: teleoperation and data-collection code, VLA training and deployment code, model checkpoints, and demonstration data.

- **Heterogeneous co-training for data efficiency.** We demonstrate that our system can be extended to new objects and instructions through cheaper data sources (stationary manipulation, HuMI [1]) without additional loco-manipulation teleop, and characterize what each data stream supplies.

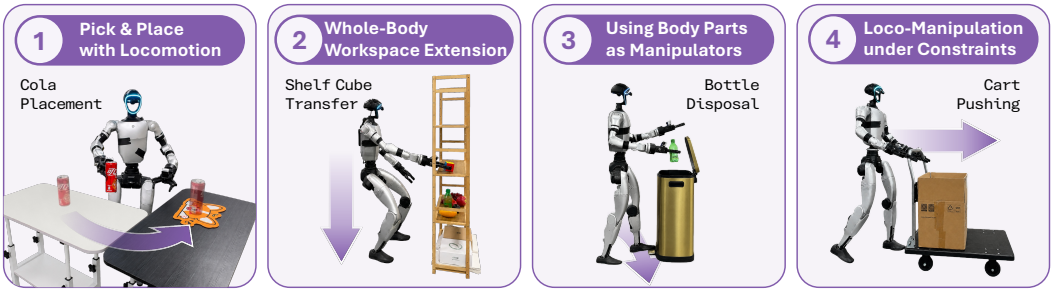


Figure 2: **The HLM-12 Benchmark.** Tasks fall into four capability families targeting different aspects of whole-body behavior, with one representative per family shown. Full task specifications are in Appendix A.

2 Design Goals and Task Suite

Before launching into the roadmap of §3, we first lay out the design goals the system aims to meet, introduce the HLM-12 benchmark, and describe the evaluation protocol.

Design goals. We posit three desiderata for a physical humanoid loco-manipulation system. Every subsection of the roadmap (§3) is framed as a response to one of them.
(G1) *Whole-body native.* One policy commands every joint of the humanoid simultaneously, treating the arms, knees, and feet as potential actuators for manipulation. Under the common decoupled formulation, the manipulation workspace shrinks to that of a wheeled dual-arm platform; behaviors that directly recruit the lower body fall outside the controller's expressible space.
(G2) *Language-steerable, and data-efficient per task.* A single policy should drive the humanoid across many tasks, steered by a language prompt rather than by swapping checkpoints. Each new skill should be learnable from a modest number of demonstrations.
(G3) *Extensible through cheap data.* Whole-body humanoid teleoperation is time-consuming and labor-intensive. The system should leverage cheaper heterogeneous data sources to reduce the demand for whole-body teleop, enabling faster skill extension to new objects and instructions.

The HLM-12 Benchmark. The HLM-12 benchmark contains 12 language-conditioned tasks, organized into four capability families that target different aspects of whole-body loco-manipulation behavior. Fig. 2 illustrates a representative task from each family. (1) *Pick-and-place with locomotion.* The policy composes walking, grasping, and placing into a single rollout (e.g., `Cola Placement`). These tasks are in principle reachable by decoupled control, making them a basic capability check for any approach. (2) *Whole-body workspace extension.* These tasks begin to exploit the humanoid form: some target objects lie outside what upper-body articulation alone can cover, so the policy must coordinate hip flexion, knee bend, and torso pitch with the arm to bring the end-effector into position (e.g., `Shelf Cube Transfer`). (3) *Using body parts as manipulators.* This family pushes further: a non-arm body part is itself the end-effector, performing a manipulation rather than supporting one (e.g., in `Bottle Disposal` the foot presses the pedal of the trash bin open). Such behaviors sit outside what a decoupled controller can express. (4) *Loco-manipulation under environmental constraint.* Here the difficulty comes from environmental or contact constraints that restrict viable motions: object geometry may force a specific manipulation trajectory (e.g., `Sword Extraction` must pull along the sword axis), or contact requirements may constrain locomotion (e.g., `Cart Pushing` must maintain a firm grasping posture while coordinating walking). The full specification of all 12 tasks appears in Appendix A.

Evaluation protocol. We adopt a shared, rigorous evaluation protocol. Every (policy, task) pair is evaluated in the real world over five independent rollouts unless explicitly noted[3]. Across the five rollouts the target object is placed in different positions, and each rollout introduces a different layout of distractor objects. For each task, the same five initial scene configurations are used across all policies to ensure a fair comparison. We score each rollout as a task progress fraction in $[0, 1]$, giving partial credit for each sub-stage. Compared with binary success rate, task progress captures more nuanced failure modes. Per-task scoring rubrics are listed in Appendix A. We report standard errors alongside the mean.

The number of tasks under evaluation grows as the roadmap (§3) advances. Most ablations in the controller/teleoperation phase (§3.1) and VLA-design phase (§3.2) are run on a 4-task subset spanning the four capability families. Once the full VLA is established at the end of §3.2, we evaluate on 8 tasks to confirm that design choices generalize. The co-training phase (§3.3) introduces 4 additional tasks (12 total) to measure whether cheap data sources can extend the policy beyond whole-body teleop coverage. Finally, an extra long-horizon task serves as the testbed for system-level comparison against state-of-the-art baselines.

3 Building a Whole-Body Loco-Manipulation System: A Roadmap

We construct the system through controlled experiments, one design decision at a time, in three phases that answer the questions raised in §1. *Controller and teleoperation* (§3.1): how to design the controller and its teleop interface for high-quality whole-body demonstrations. *VLA design* (§3.2): which adaptations turn a VLA built for static and wheeled robots into a whole-body humanoid policy. *Heterogeneous co-training* (§3.3): whether cheaper data sources can extend the policy beyond what whole-body teleop alone covers.

3.1 Low-Level Controller & Teleoperation

We follow a *two-level hierarchical control framework*, in line with recent humanoid loco-manipulation stacks [7, 17, 8, 10]. A high-level policy (a human operator during data collection, the learned VLA at deployment) takes vision and language as input and emits reference whole-body commands at low frequency (typically 10 Hz). A lightweight low-level controller consumes these commands and outputs target joint positions at higher frequency (typically 50 Hz), which are then tracked by a PD controller. With this framework fixed, two design questions immediately follow.

Opening question. *What should the interface between the high-level policy and the low-level controller look like, and what properties should the low-level controller itself satisfy?*

We study the teleop interface along two axes. First, *expressivity*: interfaces exposing only a subset of the humanoid's degrees of freedom make certain tasks unreachable by construction. Second, *demonstration quality*: interfaces with similar expressivity can still differ in the quality of demonstrations they elicit, directly affecting the learned policy. On the low-level controller side, we study one parameter that strongly affects teleop experience and data quality: future-frame preview latency, i.e., how far into the future the controller sees the reference motion before tracking it.

Joint-based whole-body teleoperation beats decoupled control and VR 3-point. We compare three teleoperation methods representative of recent humanoid loco-manipulation systems:
(1) *Decoupled control teleoperation.* The upper and lower bodies are two decoupled systems. Operator-provided targets (head and the two wrists) are mapped to upper-body joints through inverse kinematics, while an RL-trained lower-body controller, conditioned on these upper-body commands, tracks a base-velocity and root-height command. This formulation is widely adopted by recent systems including AMO [6], Ψ_0 [8], and GR00T N1.5/N1.6 [7]; we use the GR00T variant [18] here. Excluding the two gripper dimensions, the action space on the Unitree G1 is 21 dimensions: dual-arm joint positions (14) + waist joint positions (3) + root height (1) + navigation command (3).

[3] Five rollouts rather than more: each loco-manipulation trial requires resetting both the scene and the robot to its home pose, making it substantially slower than a stationary-manipulation trial.

Teleop method	Cola Placement			Shelf Cup Transfer			Bottle Disposal		
	Prog. (%)	Time (s)	Steps	Prog. (%)	Time (s)	Steps	Prog. (%)	Time (s)	Steps
Decoupled control	66.7 ± 14.9	36.7	42.3	93.3 ± 6.7	33.6	38.4	✗		
VR 3-point teleoperation	40.0 ± 6.7	67.8	12.3	✗			✗		
Joint-based whole-body teleop.	86.7 ± 8.2	40.2	12.0	80.0 ± 13.3	41.8	10.5	85.0 ± 6.1	29.7	10.3

Table 1: **Teleop method comparison. Prog. (%)**, mean task progress over 5 rollouts; **Time (s)**, mean rollout duration; **Steps**, mean footsteps per rollout. A purple ✗ marks a task the method cannot perform by construction: `Shelf Cup Transfer` requires squatting; `Bottle Disposal` requires the foot to depress a pedal.

(2) *VR 3-point teleoperation.* A widely used scheme in humanoid teleoperation [12, 13]; we adopt the SONIC variant [13] here. The operator supplies head and wrist poses via a VR headset, with a navigation command from its joystick. A learned kinematic motion planner produces the lower-body motion, yielding a hybrid command of three upper-body keypoints and lower-body joint positions that is tracked by the SONIC controller. The action space is 24 dimensions: left wrist pose (7) + right wrist pose (7) + head pose (7) + navigation command (3).
(3) *Joint-based whole-body teleoperation.* A portable motion-capture rig (here, a PICO VR headset with body trackers [19]) captures the operator's whole-body motion and retargets it in real time to *every humanoid joint* via GMR [20]. The resulting joint trajectory is tracked by a general motion tracker (we use SONIC). The action space is 32 dimensions: dual-arm joint positions (14) + dual-leg joint positions (12) + waist joint positions (3) + root roll/pitch angles and yaw angular velocity (3).

We select three tasks stressing different capabilities, collect matched data per task (40 demonstrations) under each teleoperation method, and train one VLA per teleop method (VLA details in §3.2). Results are reported in Table 1. *Joint-based whole-body teleoperation* is the only interface that completes all three tasks, reaching 80%–87% task progress at 10–12 footsteps per rollout. The two alternatives degrade in distinct ways. *Decoupled control* walks in small, visibly unnatural steps, averaging 42.3 footsteps on `Cola Placement` (a 3.5× inflation over joint-based); `Bottle Disposal` is unreachable, since depressing a pedal requires a foot motion this controller cannot express. *VR 3-point teleoperation* produces a policy that stalls indecisively in front of the cola can on `Cola Placement`, inflating rollout duration to 67.8 s and dropping task progress to 40%. Both `Shelf Cup Transfer` and `Bottle Disposal` are out of reach by construction. *Based on these results, we will adopt joint-based whole-body teleoperation as the data-collection interface.*

Joint-space retargeting beats native SMPL recording. SMPL [21] is a natural representation of human whole-body motion. Using SMPL as the action representation skips the online retargeting step joint-based collection requires, in principle eliminating errors from an imperfect retargeter. We test this alternative, called *SMPL-based whole-body teleoperation*; the SONIC controller accepts SMPL inputs natively, making it a drop-in substitute. The action space is 81-D: SMPL joint positions (72, from 24 joints × xyz) + wrist joint positions (6, for fine wrist control) + root roll/pitch angles and yaw angular velocity (3). On the 4-task subset, we collect the same number of demonstrations for each teleop method; the two are comparable in operator experience, motion quality, and throughput. We then compare the VLAs trained on each set; results are shown in Fig. 3.

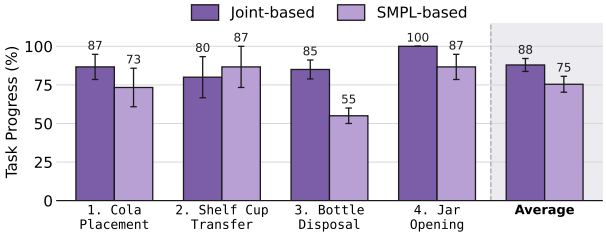


Figure 3: **Joint-based vs. SMPL-based whole-body teleop.**

Joint-space training data reaches 88% average task progress against 75% for SMPL. Two failure modes account for most of the gap. On `Bottle Disposal`, the SMPL-trained policy lifts the heel without sufficiently lifting the toes, leaving inadequate clearance to depress the pedal. On `Cola Placement`, it occasionally walks too close to the table and knocks the can over. Neither failure appears in the collected demonstrations. We attribute the gap to the much higher action-space dimensionality (81 vs. 32): SMPL's extra dimensions are largely redundant given the body's kinematic chain, yet the VLA

must still learn to coordinate them all, and this harder learning problem surfaces as the foot-lift and walking-distance errors above. Based on this finding, *we will retarget whole-body demonstrations to robot joint space online during data collection.*

Future-frame preview latency: 0.2 s balances locomotion and manipulation. A whole-body controller trained via motion tracking exposes a tunable preview latency Δt, controlling how far into the future it sees the reference motion. Longer preview yields smoother motion but adds delay between the operator's command and its enactment. We sweep $\Delta t \in \{0, 0.2, 0.4, 0.6\}$ s on `Cola Placement`, collecting the same number of demonstrations (40) per setting, training one VLA on each, and tracking average demonstration duration as a proxy for teleoperation difficulty. Results are shown in Fig. 4.

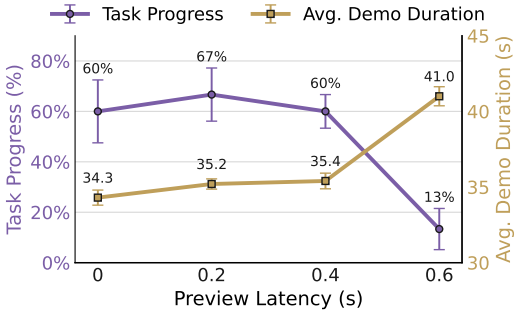


Figure 4: **Future-frame preview latency sweep.**

At $\Delta t = 0$ s, the robot is most responsive and stationary manipulation feels best, but locomotion exhibits stuttering and ground-stomping in data collection and testing. At $\Delta t = 0.6$ s, accumulated delay overwhelms the operator: demonstration duration jumps from ∼35 s to 41 s and task progress collapses to 13%. $\Delta t = 0.2$ s strikes the best balance, reaching 67% task progress[4] at a demonstration duration (35.2 s) essentially unchanged from the zero-preview case. This finding holds across multiple teleoperators in our pipeline, and $\Delta t = 0.2$ s yields high-quality demonstrations on all subsequent tasks. The data-collection pipeline that all later phases build on is now settled: *joint-based whole-body teleoperation, retargeted to robot joint space online, with 0.2 s preview latency.*

3.2 Whole-Body VLA Policy Design

We now turn to the high-level policy. An appealing starting point is a pretrained VLA that already brings vision-language reasoning and manipulation priors. However, existing VLAs nearly all target static or wheeled dual-arm platforms; none were designed for humanoid loco-manipulation.

Opening question. *How do we adapt a VLA pretrained on static and wheeled dual-arm platforms into a whole-body humanoid policy, and which design choices actually matter?*

We organize the exploration into three families: (1) *Action and proprioception interface* — adapting the VLA's original lower-DoF action space to the humanoid's high-DoF commands. (2) *The role of pretraining* — whether robot pretraining ($\pi_{0.5}$ [22]) is needed, or vision-language pretraining alone (PaliGemma [23]) or even training from scratch can already work. (3) *Faster action generation* — whether the multi-step flow matching [24] can be replaced by single-step inference. We fix a default configuration and ablate one component at a time on the 4-task subset; results are shown in Fig. 5.

Action and proprioception interface: adaptations barely affect performance. We use $\pi_{0.5}$ [22] as our default backbone and leave its internal architecture untouched, focusing instead on the interface between the VLA and the humanoid. Two things must change by construction: the output action vector and the input proprioceptive state. We ablate four design choices around these two axes. (1) *Action projection initialization.* $\pi_{0.5}$'s action projection supports up to 32 action dimensions, but our 34-dim action vector (32 dims from §3.1 plus two parallel-jaw gripper dimensions; a dexterous hand would push this higher) requires resizing it. We compare random re-initialization against weight surgery (default), which preserves the pretrained weights for the first 32 dimensions in the input and output linear projection layers and randomly initializes only the new entries. (2) *Action ordering.* $\pi_{0.5}$'s pretrained action vector is laid out as [left arm, left gripper, right arm, right gripper]. We can preserve this layout and append the

[4] Lower than the 87% on the same task in Table 1 and Fig. 3, where models are trained on multiple tasks; here we train single-task models to isolate latency.

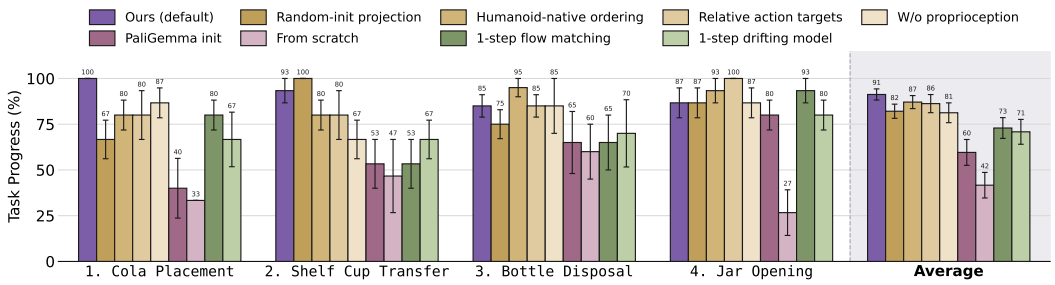


Figure 5: **VLA design ablations on the 4-task subset.** Amber: interface ablations (one choice flipped per bar); drops are minor and no single choice is the bottleneck. Rose: pretraining ablations; robot pretraining ($\pi_{0.5}$) dominates, with PaliGemma and from-scratch collapsing sharply. Sage: one-step action generation; both underperform the 10-step baseline by $\sim$20 points despite lower validation action MSE.

humanoid-specific waist and leg joints at the end (default), or pick a fresh humanoid-native ordering (e.g., legs first). (3) *Absolute vs. relative action targets.* Predict absolute joint positions (default), or relative deltas with each action chunk re-expressed relative to the first action in that chunk. (4) *Proprioceptive input.* The head- and wrist-mounted cameras do not cleanly observe the lower body; feeding the full joint-position vector as input (default) gives the policy direct access to its body pose, but risks underusing vision in favor of this proprioceptive shortcut.

We ablate each choice independently, holding the other three at default; results appear as the amber group in Fig. 5. In each case, the "wrong" choice (random-init projection, humanoid-native ordering, relative action targets, no proprioception) produces a slight drop in 4-task average task progress, with no qualitative shift in rollout behavior or failure modes. The small numerical drops are most plausibly attributed to lower robustness or to the noise floor of a 5-rollout evaluation. None of these four choices is, in itself, the bottleneck of humanoid VLA adaptation. However, this does not mean the choices can be arbitrarily combined: our extra experiments showed that removing proprioception and switching to relative actions simultaneously causes catastrophic failure, as the policy easily drifts into out-of-distribution states from which it cannot recover. Since no single alternative beats the default, we keep the default for the rest of the paper: *we will adopt weight surgery on the action projection, the pretrained bimanual ordering, absolute joint targets, and proprioception as input.*

Pretraining on non-humanoid robots transfers well to humanoid VLA adaptation. With the interface fixed, we ask whether pretraining on non-humanoid robot data (static and wheeled dual-arm) still transfers to a humanoid despite the substantial embodiment gap, or whether a vision-language backbone alone would suffice. We compare three backbone initializations: (i) $\pi_{0.5}$ (non-humanoid robot-pretrained); (ii) PaliGemma (same architecture, no robot data); (iii) random initialization. Results appear as the rose group in Fig. 5. The gap is stark: $\pi_{0.5}$ reaches 91% average task progress, PaliGemma drops to 60%, and random initialization collapses to 42%.

Behind this gap lies a surprise: action mean squared error (MSE) on held-out validation is virtually indistinguishable between the $\pi_{0.5}$- and PaliGemma-initialized models throughout fine-tuning. However, on the robot they diverge sharply: the PaliGemma-initialized policy is consistently weaker at grasping and rarely recovers from a failed grasp, while the $\pi_{0.5}$-initialized policy retries fluently. Two takeaways follow. First, $\pi_{0.5}$'s manipulation prior, especially the closed-loop "see error, correct, retry" behavior implicit in its pretraining data, transfers despite the embodiment gap. Second, action MSE is a poor proxy for the value of robot pretraining: two models with matching action MSE can behave very differently on-robot. Random initialization fails differently: the policy learns a rough stepping gait but its manipulation ability collapses almost entirely. The cross-embodiment gap from dual-arm pretraining to our humanoid is real, but dwarfed by the gap between any robot pretraining and none at all. *We will initialize from $\pi_{0.5}$ for all subsequent experiments.*

Faster action generation: one-step alternatives underperform. At inference, $\pi_{0.5}$'s flow-matching [24] action head integrates the learned vector field over multiple denoising steps (typically 10). A single-step alternative would cut per-call inference latency from $\sim$90 ms to $\sim$60 ms on our

5080 GPU, potentially improving closed-loop responsiveness. To this end, we test two alternatives: (i) one-step flow matching (set the integration steps to one at inference, no retraining); (ii) drifting model [25], a recent generative model that naturally admits one-step inference by learning a drifting field whose pushforward distribution converges during training.

Notably, validation action MSE is lower for both one-step alternatives ($\sim$0.007) than for the 10-step baseline ($\sim$0.009). However, on the robot (sage group in Fig. 5) both underperform the baseline by roughly 20 task-progress points. We hypothesize that single-step inference produces actions that are close in ℓ_2 to the target but jitterier and less temporally smooth on the robot; we leave the precise mechanism to future work. *We will keep multi-step flow matching for action generation.*

Whole-body teleop scaling: 40 demonstrations per task is a data-efficient default. With the VLA architecture fixed, we ask how the system responds to more whole-body teleop. We sweep the per-task demonstration budget on the 4-task subset (Fig. 6). The largest jump comes between 10 and 20 demos per task; returns flatten thereafter, reaching roughly 90% at 40 demos. 40 demonstrations is a modest budget, costing a skilled operator about 1.5 hour per medium-difficulty task; every ablation up to this point already uses it as the default whole-body teleop budget.

At this point we have a humanoid-adapted VLA: a $\pi_{0.5}$-initialized backbone with weight-surgery action projection, the pretrained bimanual ordering, absolute joint targets, proprioception as input, and multi-step flow matching inference. Carrying it to the 8 training tasks (40 demonstrations each), the system reaches *89%* average task progress (Fig. 7), confirming that the choices made on the 4-task subset generalize to the broader training distribution.

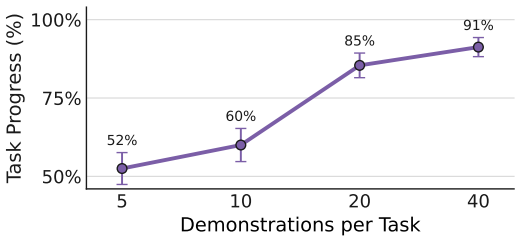


Figure 6: **Whole-body teleop data scaling.**

3.3 Heterogeneous Co-Training

With the whole-body VLA established, scaling to every task through loco-manipulation teleop is expensive. We turn to cheaper data sources and ask whether co-training can incorporate them effectively. We focus on two: (i) stationary same-embodiment teleoperation (feet-planted manipulation, no locomotion); (ii) HuMI-collected robot-free demonstrations [1] (captured with low-cost wearable devices). We measure co-training on the 4 held-out tasks that whole-body teleop never covers.

Opening question. *Can cheaper data extend the policy to tasks whole-body teleop never covers — and what does each stream supply: new motions, new semantic understanding, or both?*

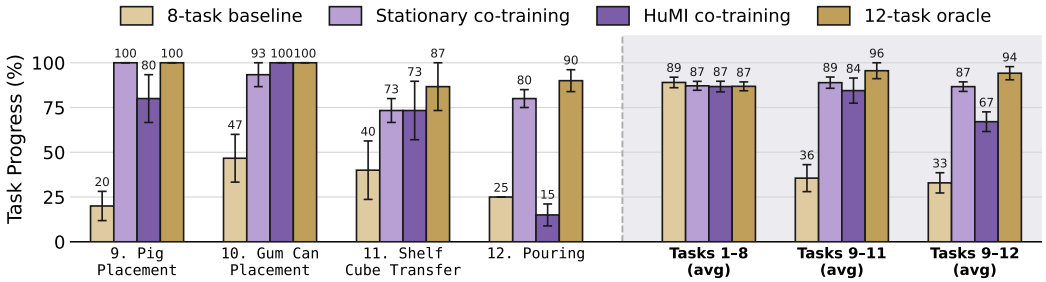


Figure 7: **Heterogeneous co-training results.** Per-task progress on the 4 held-out tasks and aggregate averages over the 8 training tasks (Tasks 1–8 (avg)), the 3 motion-reuse held-out tasks (Tasks 9–11 (avg)), and all 4 held-out tasks (Tasks 9–12 (avg)). Per-task breakdown for all 12 tasks is in Appendix E.1.

Stationary co-training delivers both new motions and new semantic understanding. Stationary teleoperation operates the same humanoid with feet planted in place: manipulation only, no locomotion. The motivation is straightforward: manipulation is contact-rich and usually demands higher positional precision than locomotion, and thus requires relatively more demonstrations. Besides,

manipulation-only demos are also much cheaper to collect (shorter episodes, smaller scenes, faster resets). The average duration of 40 demos for each held-out task is 13 min under stationary teleop vs. 21 min under full teleop, excluding overheating shutdowns and scene resets, which take much longer under full teleop.

We co-train the VLA on whole-body teleop for the 8 training tasks plus stationary teleop for the 4 held-out tasks (40 demonstrations each), and compare against two references: the 8-task baseline (whole-body teleop on 8 tasks only) and the 12-task oracle (whole-body teleop on all 12 tasks). Fig. 7 shows: stationary co-training does not regress the 8 training tasks; on the 4 held-out tasks, it lifts average progress from 33% (8-task baseline) to 87%, closing much of the gap to the oracle. A per-task breakdown reveals what stationary co-training supplies. Tasks 9–11 (`Pig Placement`, `Gum Can Placement`, `Shelf Cube Transfer`) reuse motions already present in the 8 training tasks but introduce new objects and language prompts; here stationary data provides new semantic understanding (new instruction grounding, new object recognition). `Pouring` (task 12) raises the bar: no vessel-tilt motion exists in the 8 training tasks, so co-training must deliver a new motion. Stationary co-training reaches roughly the oracle level on both kinds, indicating that *stationary co-training delivers both new semantic understanding (new objects, new instructions) and new motions.*

HuMI co-training delivers new semantic understanding but not new motions. HuMI [1] is the humanoid analog of UMI [16]: a robot-free rig of two hand-held UMI grippers plus three body-pose trackers [26] (pelvis and the two feet) that captures task-space SE(3) trajectories of the grippers and base, which an IK pipeline lifts to full-body joint positions. By construction the resulting action vector matches the dimensionality and semantics of the joint-based whole-body teleop action of §3.1, so co-training reduces to mixing the two streams with no architectural change. As with stationary teleop, we collect manipulation only. Because the humanoid stays out of the loop, HuMI is faster to collect: 40 HuMI demonstrations total 7 min per held-out task versus 13 min for stationary teleop.

We co-train the 8-task whole-body teleop set with HuMI on the 4 held-out tasks (40 demonstrations each). Fig. 7 shows: HuMI co-training does not regress the 8 training tasks. On tasks 9–11 (motion-reuse, new objects and prompts), HuMI matches stationary co-training, grounding new semantics. On `Pouring` (task 12), which requires a new motion, HuMI fails to acquire the vessel-tilt motion absent from the 8-task training set. We attribute the limitation to a two-axis domain gap between HuMI and teleop. Visually, the two differ in cameras (RealSense vs. rectified GoPro fisheye) and grippers (humanoid's adaptive parallel vs. HuMI's rigid hand-held). In action, HuMI data is fundamentally human motion: even after IK retargeting, the resulting trajectories differ from those produced by teleoperating the humanoid directly, since no robot is in the loop. Together the gaps are large enough that, at the current data scale, the policy appears to treat HuMI primarily as semantic supervision rather than motion supervision. We conjecture that scaling HuMI data could enable motion transfer as well; investigating this is a natural direction for future work. *With the current data budget, HuMI co-training delivers new semantic understanding but not new motions, at roughly half the operator-time cost of stationary same-embodiment teleop.*

The roadmap is complete and yields `OpenHLM`. Joint-based whole-body teleoperation (§3.1) and a $\pi_{0.5}$-initialized humanoid-adapted VLA (§3.2) together cover the humanoid's full degrees of freedom and reach 89% average task progress on the 8 training tasks, meeting G1 and G2. Heterogeneous co-training with cheaper data sources (§3.3) extends the policy to new objects, instructions, and motions without additional whole-body teleop, meeting G3. Comparisons so far have been internal; §4 benchmarks `OpenHLM` against state-of-the-art humanoid VLAs on a long-horizon task.

4 System-Level Comparison on Long-Horizon Tasks

Baseline systems. We compare `OpenHLM` (HuMI co-training) against two representative humanoid VLAs. (1) GR00T N1.6 [7] pairs a Cosmos-2B vision-language model [27] with a DiT [28] action head, using decoupled control as the low-level controller; its pretraining data includes Unitree G1 loco-manipulation demonstrations. (2) Ψ_0 [8] pretrains a vision-language backbone on large-scale egocentric human videos, then mid-trains an action expert on humanoid demonstrations. Both baselines

use decoupled control and share the same demonstration set; OpenHLM uses training demonstrations of the same count, but only contains a small fraction (3/10) of whole-body teleoperation data. For both baselines we use the official checkpoint and follow the recommended fine-tuning protocol.

Task and data. The humanoid performs a long-horizon language-conditioned task spanning its large vertical workspace (Fig. 8). From a home pose, it walks to a low coffee table and picks up {fruit 1} with the right hand, walks to a medium-height table and picks up {fruit 2} with the left hand, then walks to a tall shelf and places them in separate containers on the top shelf. The instruction specifies {fruit 1} and {fruit 2} from five fruits (banana, peach, mangosteen, lemon, tomato), with distractors at each site, giving $P(5, 2) = 20$ ordered pairs. We collect 6 demonstrations per pair under each condition. GR00T N1.6 and Ψ_0 receive full loco-manipulation teleop on all 20 pairs (120 demos). OpenHLM (HuMI co-training) receives full loco-manipulation teleop only on the 6 pairs drawn from {banana, peach, mangosteen} (36 demos), plus HuMI demonstrations of the manipulation phases for the remaining 14 pairs that include lemon or tomato (84 demos); this tests whether HuMI co-training extends the policy to new objects without further whole-body teleop. OpenHLM (teleop oracle) replaces the HuMI portion on the 14 lemon/tomato pairs with joint-based whole-body teleop (§3.1), yielding full loco-manipulation teleop on all 20 pairs, and serves as the oracle.

Figure 8: **Long-horizon language-conditioned task.**

Results. Table 2 reports task progress and total demo time. OpenHLM (HuMI co-training) reaches 87.5% task progress, far above Ψ_0 (48.8%) and GR00T N1.6 (57.5%) and within 10 points of the teleop oracle (97.5%), at less than half the operator time (1.14 vs. 2.70–2.73 hours). Both baselines exhibit weak grasping in our evaluated rollouts: they execute a stereotyped arm trajectory rather than tracking the language-specified fruit. Ψ_0 additionally stops short of the tall shelf, indicating weaker locomotion. Notably, GR00T N1.6 and Ψ_0 both include Unitree G1 demonstrations in their pretraining data, while OpenHLM's $\pi_{0.5}$ initialization does not. This suggests that mixing humanoid data into pretraining is not enough on its own; building a strong humanoid VLA is a question of design details, and the roadmap of §3 is precisely about getting those details right.

Method	Progress (%)	Demo Duration
Ψ_0	48.8 ± 4.4	2.70 h
GR00T N1.6	57.5 ± 4.6	2.70 h
OpenHLM (HuMI co-training)	87.5 ± 3.7	1.14 h
OpenHLM (teleop oracle)	97.5 ± 1.7	2.73 h

Table 2: **Long-horizon task progress.** 10 of 20 pairs sampled uniformly, one rollout each. Held-out lemon/tomato subset (7/10 pairs collected by HuMI) introduction and scoring rubric are listed in Appendix D.1.

5 Related Work

Whole-body humanoid teleoperation. Teleoperation is a foundational data source for humanoid loco-manipulation, and the interface itself shapes what behaviors the trained policy can express. *Decoupled control* drives the lower body with a separate locomotion controller that takes velocity or coarse motion commands [2, 3, 4, 5, 6, 7, 8]; this rules out coordinated whole-body behaviors and

any use of the lower body as a manipulator. *Whole-body teleoperation* instead commands all degrees of freedom jointly [29, 30, 12, 31], typically through a learned motion-tracking controller [32, 33, 34]. Within this paradigm, TWIST2 [9] establishes the data-collection side, but pairs it with single-task visuomotor imitation [35] rather than a multi-task VLA; SONIC [13] scales motion-tracking controllers to unprecedented size, but reports only a handful of experiments in which the controller is plugged into a VLA. How the teleoperation interface interacts with a VLA-driven policy is therefore left open. To this end, we use whole-body teleop as the data source for a VLA and systematically compare alternative teleop formats, offering practical insights for future use (§3.1).

Humanoid VLAs. Vision-language-action models fine-tune a vision-language backbone to emit robot actions. RT-2 [11], OpenVLA [14], and the π series [15, 22, 36, 37] establish strong generalization across manipulation scenes, but on fixed-base or wheeled platforms whose action distribution is dominated by arm and wrist motion. Two recent efforts extend this recipe to humanoids. GR00T N1.6 [7] pretrains a vision-language model with a DiT [28] action head on a mixture that includes humanoid demonstrations; Ψ_0 [8] pretrains on large-scale egocentric human video [38] and then mid-trains an action expert on humanoid demonstrations. Both follow the same recipe: add humanoid data to pretraining, then finetune. Yet in our system-level comparison (§4), both underperform a $\pi_{0.5}$-initialized humanoid-adapted VLA (§3.2) that pretrains on no humanoid data at all. The bottleneck instead lies in design details: how to humanoid-adapt a VLA backbone, what teleop format to fit, and how to mix in cheap data. These are the questions §3 works through.

Heterogeneous data for humanoids. Cheap, robot-free data collection has a rich toolkit: handheld sensorized grippers [16, 39, 40], wearable smart glasses [41], and AR/VR headsets [42, 43, 19]. Together, these tools have powered scene-understanding pretraining [44, 45], visual representation learning [46, 47], motion priors [48], latent action codebooks [49, 50], and direct co-training of robot policies [51, 52]. However, most of this work targets fixed-base or wheeled manipulation; humanoid loco-manipulation is still emerging. As a first step in this direction, HuMI [1] extends UMI with wearable body trackers and IK retargeting to full-body humanoid joints; we adopt it as a starting point, while the broader space (smart glasses, AR/VR headsets) remains open. The closest concurrent effort, EgoHumanoid [17], also studies human-to-humanoid co-training, but along an orthogonal axis: it pairs human and teleop demonstrations on the same full tasks to drive environment generalization, whereas §3.3 uses cheap data streams to extend the policy to new objects and language prompts that whole-body teleop never covers.

6 Discussion, Conclusion and Limitations

Whole-body humanoid loco-manipulation is an exciting and rapidly advancing area in robotics. Rather than scaling humanoid data or model size indiscriminately, we believe progress depends on getting the design details right: how the humanoid is teleoperated, how a manipulation VLA is adapted to the humanoid's full action space, and how cheap data extends the policy beyond what teleoperation can cover. These are the fundamental questions our roadmap (§3) works through, with each design choice backed by a controlled experiment. This roadmap yields `OpenHLM`, a recipe with three ingredients: joint-based whole-body teleoperation as the data source, a $\pi_{0.5}$-initialized humanoid-adapted VLA as the policy, and heterogeneous co-training to broaden object and language coverage at a fraction of the operator-time cost. On a challenging long-horizon task spanning the humanoid's large vertical workspace, `OpenHLM` outperforms two strong humanoid VLAs (GR00T N1.6 and Ψ_0) at less than half the demonstration time, and approaches a teleop oracle that uses more than twice the demo duration. We will release our code, data, and models to support future research toward reliable everyday humanoids.

Our work has several limitations that future research can address. (1) Though our work explores a broad design space of humanoid VLAs, it is still confined to our limited experiment resources (e.g., we only use Unitree G1 as the humanoid platform). Whether our conclusions hold in more general settings (e.g., other humanoid robots, more diverse scenes) remains uncertain. (2) Heterogeneous co-training does not yet close the gap to whole-body teleoperation demonstrations. HuMI data im-

proves grounding for new objects and language prompts but is less effective at teaching novel motion patterns, likely due to visual and action gaps between robot-free and on-robot demonstrations, or our limited data budget. How to better utilize cheap robot-free data remains an open question. (3) We do not explore VLA architecture design specifically for loco-manipulation tasks, and during training on the current architecture, several findings remain mechanistically underexplained: validation action MSE does not reliably predict on-robot performance, and one-step action generation underperforms despite lower MSE. We hope that future researchers could extend our roadmap following these directions.

Core Contributors

The five core contributors jointly shaped this work, each leading complementary and often overlapping parts of the project. The descriptions below highlight each author's primary areas of leadership, and should not be interpreted as disjoint partitions or as a ranking of individual effort.

Yingdong Hu. Led the manuscript outline and high-level policy (VLA) design, and contributed to the low-level controller design and the teleoperation and evaluation software.

Haodong Zhu. Led the teleoperation and evaluation software design and the organization of the full codebase, and contributed to teleoperation data collection and baseline implementation.

Boyuan Zheng. Led co-training data collection and baseline implementation, including data collection and evaluation, and contributed to teleoperation data collection.

Yihang Hu. Led hardware design and assembly, task scene construction and process design, and teleoperation data collection, and contributed to the teleoperation and evaluation software.

Tong Zhang. Led the low-level controller design and testing, and contributed to high-level policy (VLA) design and hardware design and assembly.

Acknowledgements

This research was conducted with the support of the Shanghai Qi Zhi Institute & Spirit AI Innovation Program and the Tsinghua University Dushi Program. Funding and support for this work were also provided by the Tsinghua University - Keystone Electrical (Zhejiang) Co.,Ltd Joint Research Center for Embodied Multimodal Artificial Intelligence (JCEMAI). Additionally, we would like to extend our thanks to the Xiongan AI Institute.

References

[1] R. Nai, B. Zheng, J. Zhao, H. Zhu, S. Dai, Z. Chen, Y. Hu, Y. Hu, T. Zhang, C. Wen, et al. Humanoid manipulation interface: Humanoid whole-body manipulation from robot-free demonstrations. *arXiv preprint arXiv:2602.06643*, 2026.

[2] X. Cheng, Y. Ji, J. Chen, R. Yang, G. Yang, and X. Wang. Expressive whole-body control for humanoid robots. *arXiv preprint arXiv:2402.16796*, 2024.

[3] M. Liu, Z. Chen, X. Cheng, Y. Ji, R.-Z. Qiu, R. Yang, and X. Wang. Visual whole-body control for legged loco-manipulation. *arXiv preprint arXiv:2403.16967*, 2024.

[4] C. Lu, X. Cheng, J. Li, S. Yang, M. Ji, C. Yuan, G. Yang, S. Yi, and X. Wang. Mobile-television: Predictive motion priors for humanoid whole-body control. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5364–5371. IEEE, 2025.

[5] Q. Ben, F. Jia, J. Zeng, J. Dong, D. Lin, and J. Pang. Homie: Humanoid loco-manipulation with isomorphic exoskeleton cockpit. *arXiv preprint arXiv:2502.13013*, 2025.

[6] J. Li, X. Cheng, T. Huang, S. Yang, R.-Z. Qiu, and X. Wang. Amo: Adaptive motion optimization for hyper-dexterous humanoid whole-body control. *arXiv preprint arXiv:2505.03738*, 2025.

[7] NVIDIA GEAR Team. Gr00t n1.6: An improved open foundation model for generalist humanoid robots. https://research.nvidia.com/labs/gear/gr00t-n1_6/, Dec. 2025. NVIDIA Research Blog, Accessed: 2026-05-06.

[8] S. Wei, H. Jing, B. Li, Z. Zhao, J. Mao, Z. Ni, S. He, J. Liu, X. Liu, K. Kang, et al. Ψ_0: An open foundation model towards universal humanoid loco-manipulation. *arXiv preprint arXiv:2603.12263*, 2026.

[9] Y. Ze, S. Zhao, W. Wang, A. Kanazawa, R. Duan, P. Abbeel, G. Shi, J. Wu, and C. K. Liu. Twist2: Scalable, portable, and holistic humanoid data collection system. *arXiv preprint arXiv:2511.02832*, 2025.

[10] Figure AI. Introducing helix 02: Full-body autonomy. https://www.figure.ai/news/helix-02, Jan. 2026. Figure AI Blog, Accessed: 2026-05-06.

[11] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR, 2023.

[12] T. He, Z. Luo, X. He, W. Xiao, C. Zhang, W. Zhang, K. Kitani, C. Liu, and G. Shi. Omnih2o: Universal and dexterous human-to-humanoid whole-body teleoperation and learning. *arXiv preprint arXiv:2406.08858*, 2024.

[13] Z. Luo, Y. Yuan, T. Wang, C. Li, S. Chen, F. Castaneda, Z.-A. Cao, J. Li, D. Minor, Q. Ben, et al. Sonic: Supersizing motion tracking for natural humanoid whole-body control. *arXiv preprint arXiv:2511.07820*, 2025.

[14] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[15] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[16] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. *arXiv preprint arXiv:2402.10329*, 2024.

[17] M. Shi, S. Peng, J. Chen, H. Jiang, Y. Li, D. Huang, P. Luo, H. Li, and L. Chen. Egohumanoid: Unlocking in-the-wild loco-manipulation with robot-free egocentric demonstration. *arXiv preprint arXiv:2602.10106*, 2026.

[18] NVIDIA GEAR Team. Decoupled wbc. https://nvlabs.github.io/GR00T-WholeBodyControl/references/decoupled_wbc.html, 2026. GR00T-WholeBodyControl Documentation. Last updated: 2026-05-07. Accessed: 2026-05-14.

[19] PICO Immersive Pte. Ltd. PICO 4 Ultra. https://www.picoxr.com/global/products/pico4-ultra, 2024. Product webpage. Accessed: 2026-05-14.

[20] J. P. Araujo, Y. Ze, P. Xu, J. Wu, and C. K. Liu. Retargeting matters: General motion retargeting for humanoid motion tracking. *arXiv preprint arXiv:2510.02252*, 2025.

[21] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black. Smpl: A skinned multi-person linear model. *ACM Transactions on Graphics (TOG)*, 34(6), Oct. 2015. doi:10.1145/2816795.2818013. URL https://doi.org/10.1145/2816795.2818013.

[22] P. Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

[23] L. Beyer, A. Steiner, A. S. Pinto, A. Kolesnikov, X. Wang, D. Salz, M. Neumann, I. Alabdulmohsin, M. Tschannen, E. Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

[24] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[25] M. Deng, H. Li, T. Li, Y. Du, and K. He. Generative modeling via drifting. *arXiv preprint arXiv:2602.04770*, 2026.

[26] HTC VIVE. VIVE Ultimate Tracker. https://www.vive.com/eu/accessory/vive-ultimate-tracker/. Accessed: 2026-05-16.

[27] NVIDIA. Cosmos-reason2: Physical ai common sense and embodied reasoning models. https://huggingface.co/nvidia/Cosmos-Reason2-8B, 2025. Accessed: 2026-05-17.

[28] W. Peebles and S. Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023.

[29] Y. Ze, Z. Chen, J. P. Araújo, Z.-a. Cao, X. B. Peng, J. Wu, and C. K. Liu. Twist: Teleoperated whole-body imitation system. *arXiv preprint arXiv:2505.02833*, 2025.

[30] Z. Fu, Q. Zhao, Q. Wu, G. Wetzstein, and C. Finn. Humanplus: Humanoid shadowing and imitation from humans. *arXiv preprint arXiv:2406.10454*, 2024.

[31] T. Zhang, B. Zheng, R. Nai, Y. Hu, Y.-J. Wang, G. Chen, F. Lin, J. Li, C. Hong, K. Sreenath, and Y. Gao. Hub: Learning extreme humanoid balance. *arXiv preprint arXiv:2505.07294*, 2025.

[32] Q. Liao, T. E. Truong, X. Huang, Y. Gao, G. Tevet, K. Sreenath, and C. K. Liu. Beyondmimic: From motion tracking to versatile humanoid control via guided diffusion. *arXiv preprint arXiv:2508.08241*, 2025.

[33] W. Zeng, S. Lu, K. Yin, X. Niu, M. Dai, J. Wang, and J. Pang. Behavior foundation model for humanoid robots. *arXiv preprint arXiv:2509.13780*, 2025.

[34] Z. Chen, M. Ji, X. Cheng, X. Peng, X. B. Peng, and X. Wang. Gmt: General motion tracking for humanoid whole-body control. *arXiv preprint arXiv:2506.14770*, 2025.

[35] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

[36] P. Intelligence, A. Amin, R. Aniceto, A. Balakrishna, K. Black, K. Conley, G. Connors, J. Darpinian, K. Dhabalia, J. DiCarlo, et al. $\pi^*_{0.6}$: A VLA That Learns From Experience. *arXiv preprint arXiv:2511.14759*, 2025.

[37] P. Intelligence, A. Amin, R. Aniceto, A. Balakrishna, G. Balke, K. Black, G. Bokinsky, S. Cao, T. Charbonnier, et al. $\pi_{0.7}$: A Steerable Generalist Robotic Foundation Model with Emergent Capabilities. *arXiv preprint arXiv:2604.15483*, 2026.

[38] R. Hoque, P. Huang, D. J. Yoon, M. Sivapurapu, and J. Zhang. Egodex: Learning dexterous manipulation from large-scale egocentric video. *arXiv preprint arXiv:2505.11709*, 2025.

[39] K. Liu, C. Guan, Z. Jia, Z. Wu, X. Liu, T. Wang, S. Liang, P. Chen, P. Zhang, H. Song, et al. Fastumi: A scalable and hardware-independent universal manipulation interface with dataset. *arXiv preprint arXiv:2409.19499*, 2024.

[40] F. Lin, Y. Hu, P. Sheng, C. Wen, J. You, and Y. Gao. Data scaling laws in imitation learning for robotic manipulation. In *International Conference on Learning Representations*, volume 2025, pages 54877–54910, 2025.

[41] J. Engel, K. Somasundaram, M. Goesele, A. Sun, A. Gamino, A. Turner, A. Talattof, A. Yuan, B. Souti, B. Meredith, et al. Project aria: A new tool for egocentric multi-modal ai research. *arXiv preprint arXiv:2308.13561*, 2023.

[42] Apple Inc. Apple vision pro technical specifications. https://www.apple.com/sg/apple-vision-pro/specs/, 2026. Accessed: 2026-05-18.

[43] Meta Platforms, Inc. Meta quest 3. https://www.meta.com/quest/quest-3/, 2026. Accessed: 2026-05-18.

[44] K. Grauman, A. Westbury, L. Torresani, K. Kitani, J. Malik, T. Afouras, K. Ashutosh, V. Baiyya, S. Bansal, B. Boote, et al. Ego-exo4d: Understanding skilled human activity from first-and third-person perspectives. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19383–19400, 2024.

[45] K. Grauman, A. Westbury, E. Byrne, Z. Chavis, A. Furnari, R. Girdhar, J. Hamburger, H. Jiang, M. Liu, X. Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18995–19012, 2022.

[46] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

[47] Y. J. Ma, S. Sodhani, D. Jayaraman, O. Bastani, V. Kumar, and A. Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. *arXiv preprint arXiv:2210.00030*, 2022.

[48] R. Yang, Q. Yu, Y. Wu, R. Yan, B. Li, A.-C. Cheng, X. Zou, Y. Fang, X. Cheng, R.-Z. Qiu, et al. Egovla: Learning vision-language-action models from egocentric human videos. *arXiv preprint arXiv:2507.12440*, 2025.

[49] S. Ye, J. Jang, B. Jeon, S. J. Joo, J. Yang, B. Peng, A. Mandlekar, R. Tan, Y.-W. Chao, B. Y. Lin, et al. Latent action pretraining from videos. In *International Conference on Learning Representations*, volume 2025, pages 28213–28239, 2025.

[50] Q. Bu, Y. Yang, J. Cai, S. Gao, G. Ren, M. Yao, P. Luo, and H. Li. Univla: Learning to act anywhere with task-centric latent actions. *arXiv preprint arXiv:2505.06111*, 2025.

[51] C. Yuan, R. Zhou, M. Liu, Y. Hu, S. Wang, L. Yi, C. Wen, S. Zhang, and Y. Gao. Motiontrans: Human vr data enable motion-level learning for robotic manipulation policies. *arXiv preprint arXiv:2509.17759*, 2025.

[52] R.-Z. Qiu, S. Yang, X. Cheng, C. Chawla, J. Li, T. He, G. Yan, D. J. Yoon, R. Hoque, L. Paulsen, et al. Humanoid policy˜ human policy. *arXiv preprint arXiv:2503.13441*, 2025.

[53] ChangingTek Robotics Technology (Suzhou) Co., Ltd. CTAG2F90-D Electric Parallel Gripper. https://en.changingtek.com/diandong/147. Accessed: 2026-05-29.

[54] S. Bai, Y. Cai, R. Chen, K. Chen, X. Chen, Z. Cheng, L. Deng, W. Ding, C. Gao, C. Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Appendices

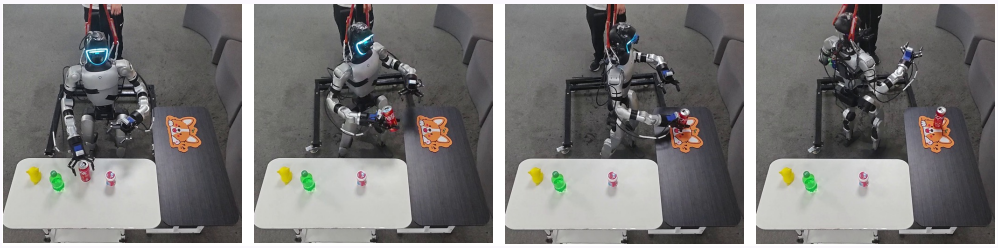
A The HLM-12 Benchmark

A.1 The 12 Tasks

We present the detailed settings of our HLM-12 benchmark. For each task, we provide the corresponding **Language Prompt**, **Detailed Process**, **Evaluation Rubric**, and the **Capability Stressed**.

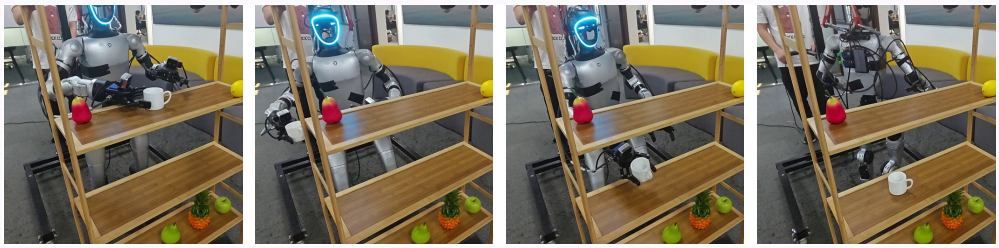
Task 1: `Cola Placement`

- **Language Prompt:** Walk forward to the table, put the red can on the orange mouse pad with the right hand, turn around and walk back.
- **Detailed Process:** The robot needs to walk to an appropriate position in front of the white table, locate the cola can placed randomly among other distractor objects, and carefully grasp it without pushing it away or causing it to fall. It then turns about 90 degrees to the left using its feet and waist to face the mouse pad, and places the can onto it. Finally, the robot turns around and walks back to its initial location.
- **Evaluation Rubric:** 3 points in total:
  - The robot successfully walks to the correct position in front of the table and later returns to its initial location. (1 pt)
  - The robot picks up the correct object (the cola can). (1 pt)
  - The cola can is placed exactly on the target location (the orange mouse pad). (1 pt)
- **Capability Stressed:** *Pick-and-place with locomotion*.



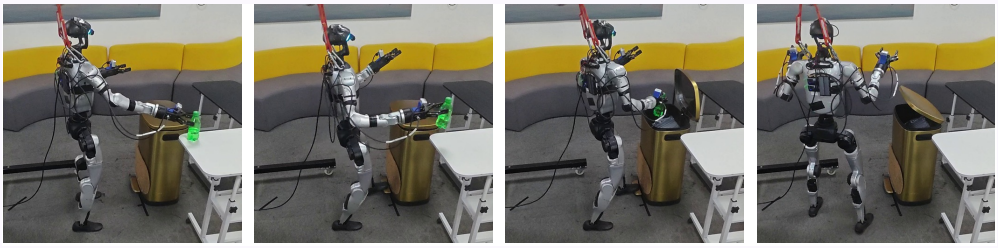
Task 2: `Shelf Cup Transfer`

- **Language Prompt:** Walk forward to the shelf, put the white cup from the upper level to the lower level with the right hand, turn around and walk back.
- **Detailed Process:** The robot needs to walk to an appropriate position in front of the shelf, locate and grasp the white cup placed randomly among other distractor objects on the upper level, then bend down and carefully place it onto the lower level of the shelf. Finally, the robot turns around and walks back to its initial location.
- **Evaluation Rubric:** 3 points in total:
  - The robot successfully walks to the correct position in front of the shelf and later returns to its initial location. (1 pt)
  - The robot picks up the correct object (the white cup). (1 pt)
  - The white cup is successfully placed on the lower level of the shelf. (1 pt)
- **Capability Stressed:** *Whole-body workspace extension*.



Task 3: `Bottle Disposal`

- **Language Prompt:** Walk forward to the dust bin, throw the green bottle into the dust bin, turn around and walk back.
- **Detailed Process:** The robot needs to walk to an appropriate position in front of the dust bin, locate and grasp the green bottle among other distractor objects on the table to the right, then press the pedal with the left foot to open the dust bin lid, and place the bottle completely inside the bin. Finally, the robot turns around and walks back to its initial location.
- **Evaluation Rubric:** 4 points in total:
  - The robot successfully walks to the correct position in front of the dust bin and later returns to its initial location. (1 pt)
  - The robot picks up the correct object (the green bottle). (1 pt)
  - The robot successfully presses the pedal to keep the lid open. (1 pt)
  - The bottle is successfully placed inside the bin. (1 pt)
- **Capability Stressed:** *Using body parts as manipulators*.

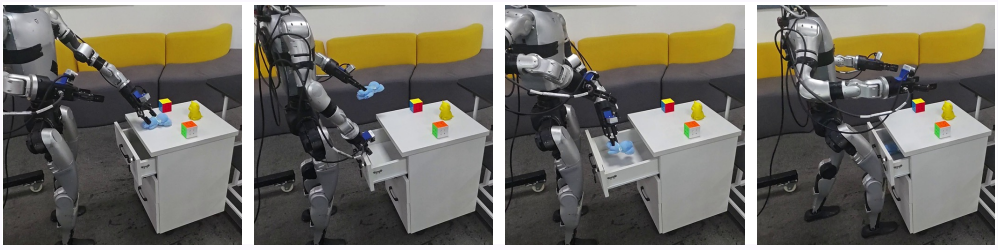


Task 4: `Jar Opening`

- **Language Prompt:** Walk forward to the white table, open the red jar with both hands, turn around and walk back.
- **Detailed Process:** The robot needs to walk to an appropriate position in front of the white table, locate the red jar placed randomly among other distractor objects, grasp the upper lid with the right hand and the lower body with the left hand, then separate the two parts by pulling them apart with both hands. After opening the jar, the robot places both parts back onto the table. Finally, the robot turns around and walks back to its initial location.
- **Evaluation Rubric:** 3 points in total:
  - The robot successfully walks to the correct position in front of the table and later returns to its initial location. (1 pt)
  - The robot uses the right hand to grasp the correct part of the correct object (the upper lid of the red jar). (1 pt)
  - The robot uses the left hand to grasp the correct part (the lower body of the red jar), and the jar is successfully opened. (1 pt)
- **Capability Stressed:** *Loco-manipulation under environmental constraint*.

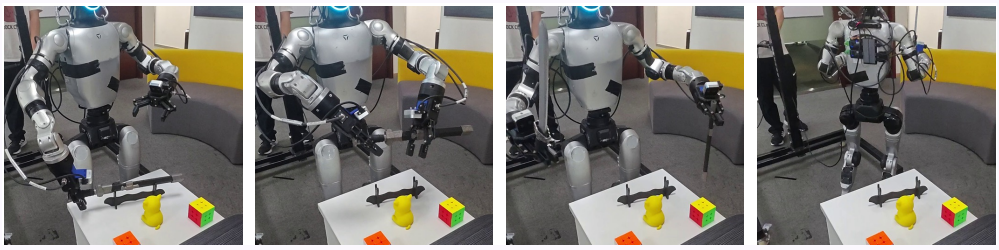
Task 5: `Toy Stowing`

- **Language Prompt:** Walk forward to the drawer, pick up the blue toy bear with the left hand, open the drawer with the right hand and drop the blue toy bear inside, close the drawer with the right leg, turn around and walk back.
- **Detailed Process:** The robot needs to walk to an appropriate position in front of the drawer, locate and grasp the blue toy bear among other distractor objects with the left gripper, then close the right gripper, insert it into the small gap of the drawer, and pull the drawer open. After placing the toy bear inside the drawer, the robot uses the right leg to gently push the drawer closed. Finally, the robot turns around and walks back to its initial location.
- **Evaluation Rubric:** 5 points in total:
  - The robot successfully walks to the correct position in front of the drawer and later returns to its initial location. (1 pt)
  - The robot picks up the correct object (the blue toy bear). (1 pt)
  - The robot successfully inserts the right gripper into the gap and pulls the drawer open. (1 pt)
  - The toy bear is placed exactly inside the drawer. (1 pt)
  - The drawer is gently pushed closed using the right leg. (1 pt)
- **Capability Stressed:** *Using body parts as manipulators*.



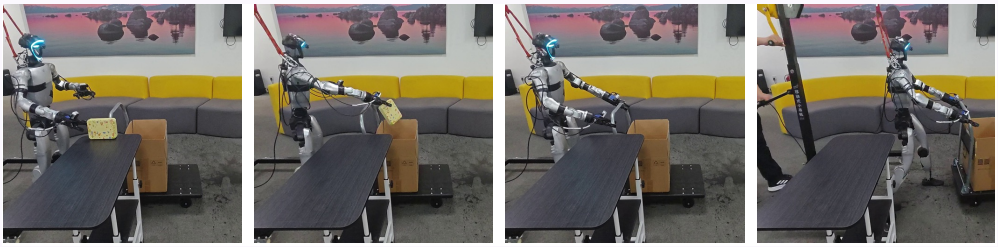
Task 6: `Sword Extraction`

- **Language Prompt:** Walk forward to the white drawer, pull out the sword with both hands, turn around and walk back.
- **Detailed Process:** The robot needs to walk to an appropriate position in front of the white cabinet, locate the sword on the rack, grasp the hilt with the right hand and lift the sword, then grasp the scabbard with the left hand and pull the hilt and scabbard apart with both hands to extract the sword. Afterward, the robot turns around and walks back to its initial location while holding the sword.
- **Evaluation Rubric:** 4 points in total:
  - The robot successfully walks to the correct position in front of the cabinet and later returns to its initial location. (1 pt)
  - The robot uses the right hand to grasp the correct part of the correct object (the hilt of the sword). (1 pt)
  - The robot uses the left hand to grasp the correct part (the scabbard of the sword). (1 pt)
  - The robot successfully pulls the sword out of the scabbard. (1 pt)
- **Capability Stressed:** *Loco-manipulation under environmental constraint*.



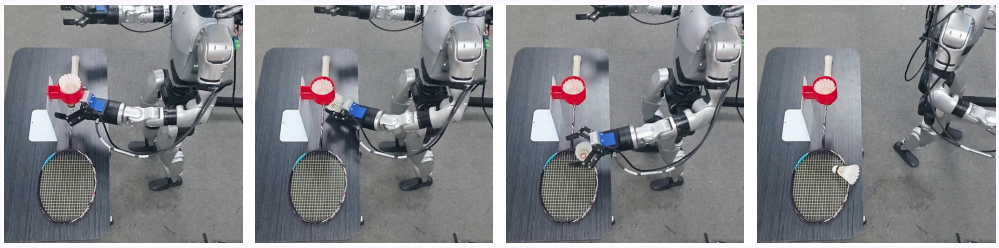
Task 7: `Cart Pushing`

- **Language Prompt:** Walk forward to the cart, throw the box onto the cart, then push the cart away.
- **Detailed Process:** The robot needs to walk to an appropriate position in front of the cart, locate and grasp the box on the table to the right, and drop it into the large box on the cart. It then grasps the left and right sides of the cart handle sequentially with the corresponding hands. Afterward, the robot pushes the cart forward to carry the contents away.
- **Evaluation Rubric:** 4 points in total:
  - The robot successfully walks to the correct position in front of the cart and then successfully pushes the cart forward. (1 pt)
  - The robot uses the right hand to grasp the correct object (the yellow iron box). (1 pt)
  - The iron box is placed exactly into the large box on the cart. (1 pt)
  - The robot successfully grasps both sides of the cart handle. (1 pt)
- **Capability Stressed:** *Loco-manipulation under environmental constraint*.



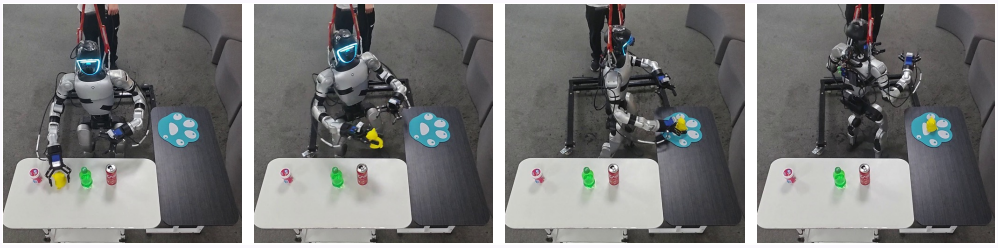
Task 8: `Shuttlecock Setup`

- **Language Prompt:** Walk forward to the black table, pull out one shuttle and put it on the racket with the left hand, turn around and walk back.
- **Detailed Process:** The robot needs to walk to an appropriate position in front of the black table, locate the upside-down shuttlecock dispenser placed on the tabletop, grasp the bottom shuttlecock with the left gripper and pull it downward to remove it, then place it onto the racket face on the left side. Finally, the robot turns around and walks back to its initial location.
- **Evaluation Rubric:** 3 points in total:
  - The robot successfully walks to the correct position in front of the table and later returns to its initial location. (1 pt)
  - The robot uses the left hand to locate the shuttlecock dispenser and pull out one shuttlecock successfully. (1 pt)
  - The robot places the shuttlecock exactly onto the racket face. (1 pt)
- **Capability Stressed:** *Loco-manipulation under environmental constraint*.



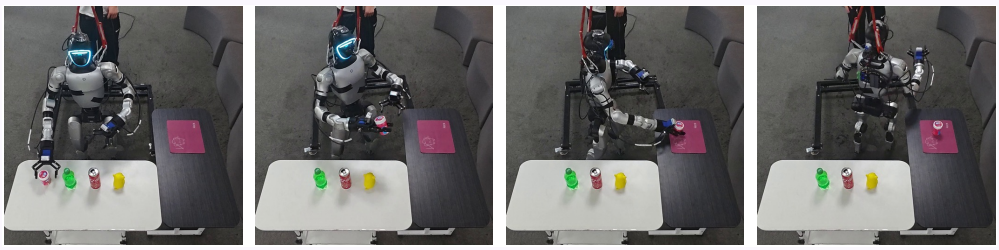
Task 9: `Pig Placement`

- **Language Prompt:** Walk forward to the table, put the yellow toy pig on the blue mouse pad with the right hand, turn around and walk back.
- **Detailed Process:** The robot needs to walk to an appropriate position in front of the white table, locate the yellow toy pig placed randomly among other distractor objects, and carefully grasp it. It then turns about 90 degrees to the left using its feet and waist to face the mouse pad, and places the toy pig onto it. Finally, the robot turns around and walks back to its initial location.
- **Evaluation Rubric:** 3 points in total:
  - The robot successfully walks to the correct position in front of the table and later returns to its initial location. (1 pt)
  - The robot picks up the correct object (the yellow toy pig). (1 pt)
  - The toy pig is placed exactly on the target location (the blue mouse pad). (1 pt)
- **Capability Stressed:** *Pick-and-place with locomotion*.



Task 10: `Gum Can Placement`

- **Language Prompt:** Walk forward to the table, put the pink gum can on the maroon mouse pad with the right hand, turn around and walk back.
- **Detailed Process:** The robot needs to walk to an appropriate position in front of the white table, locate the pink gum can placed randomly among other distractor objects, and carefully grasp it. It then turns about 90 degrees to the left using its feet and waist to face the mouse pad, and places the gum can onto it. Finally, the robot turns around and walks back to its initial location.
- **Evaluation Rubric:** 3 points in total:
  - The robot successfully walks to the correct position in front of the table and later returns to its initial location. (1 pt)
  - The robot picks up the correct object (the pink gum can). (1 pt)
  - The gum can is placed exactly on the target location (the maroon mouse pad). (1 pt)
- **Capability Stressed:** *Pick-and-place with locomotion*.

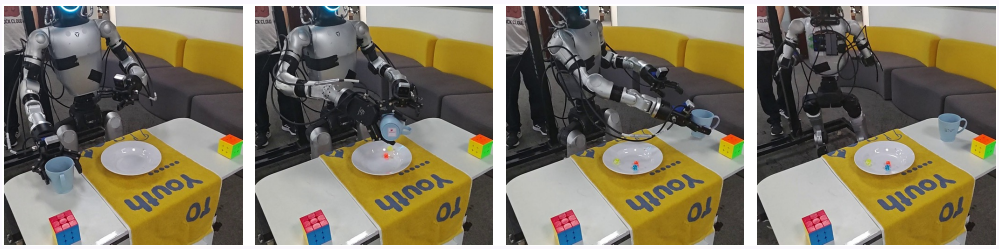


Task 11: `Shelf Cube Transfer`

- **Language Prompt:** Walk forward to the shelf, put the cube from the upper level to the lower level with the right hand, turn around and walk back.
- **Detailed Process:** The robot needs to walk to an appropriate position in front of the shelf, locate and grasp the cube placed randomly among other distractor objects on the upper level, then bend down and carefully place it onto the lower level of the shelf. Finally, the robot turns around and walks back to its initial location.
- **Evaluation Rubric:** 3 points in total:
  - The robot successfully walks to the correct position in front of the shelf and later returns to its initial location. (1 pt)
  - The robot picks up the correct object (the cube). (1 pt)
  - The cube is successfully placed on the lower level of the shelf. (1 pt)
- **Capability Stressed:** *Whole-body workspace extension*.

Task 12: `Pouring`

- **Language Prompt:** Walk forward to the white table, pour the contents from the blue cup into the white plate with the right hand, turn around and walk back.
- **Detailed Process:** The robot needs to walk to an appropriate position in front of the white table, locate the blue cup placed randomly among other distractor objects, grasp it with the right hand, and pour the dice inside into the white plate placed on the towel at the center of the table. The robot then places the empty cup onto the left side of the table while turning around to the left, and finally walks back to its initial location.
- **Evaluation Rubric:** 4 points in total:
  - The robot successfully walks to the correct position in front of the table and later returns to its initial location. (1 pt)
  - The robot uses the right hand to grasp the correct object (the blue cup). (1 pt)
  - The robot pours the dices successfully into the white plate. (1 pt)
  - The robot places the empty cup on the left side of the table. (1 pt)
- **Capability Stressed:** *Loco-manipulation under environmental constraint*.



A.2 Overall Evaluation Protocol

We adopt a shared, rigorous evaluation protocol: Every (policy, task) pair is evaluated in the real world over five independent rollouts. Note that we use five rollouts rather than more because each loco-manipulation trial requires resetting both the scene and the robot to its home pose, making it

substantially slower than a stationary-manipulation trial. Across the five rollouts the target object is placed in different positions, and each rollout introduces a different layout of distractor objects. For each task, the same five initial scene configurations (recorded by photographs) are used across all policies to ensure fair comparisons. Rather than recording only binary success/failures, we record task progress by assigning points based on the evaluation rubrics of each task introduced before. The final score of a specific rollout appears as a task progress fraction in $[0, 1]$. For example, in the `Cola Placement` task, if the robot does everything well except that the cola can is placed outside the mouse pad, it loses 1 point and gets $2/3 = 67\%$ task progress. Compared with the binary success rate, task progress captures more nuanced failure modes. We report standard errors alongside the mean.

B Hardware Setup

B.1 Humanoid Robot

All our loco-manipulation tasks are carried out by a Unitree G1 robot, with two ChangingTek CTAG2F90-D grippers [53] equipped on its wrists to enable grasping. Visual perception is provided by a Unitree SV1-25 fisheye stereo camera mounted on the robot's head, together with two Intel RealSense D405 cameras mounted on the wrists. The associated connection cables are routed and secured along the back of the humanoid to avoid interfering motions. Figure 9 provides detailed illustrations of the hardware configuration.

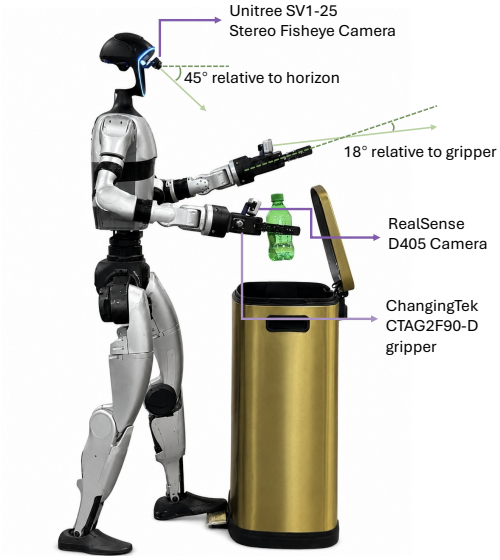


Figure 9: **Humanoid robot hardware.** The Unitree G1 is equipped with wrist-mounted grippers and onboard cameras.

Figure 10: **HuMI hardware.** Handheld grippers and body trackers are used to collect teleoperation-free demonstrations.

B.2 Whole-Body Teleoperation Hardware

For all on-robot teleoperation experiments, we use a PICO4U VR kit [19] consisting of a head-mounted display (HMD), two handheld controllers, and two leg-mounted motion trackers, as shown in Fig. 11. The relative poses of these five devices are used to reconstruct the operator's motion, which is later retargeted to the humanoid robot. The HMD streams real-time visual feedback from the robot's head-mounted camera and two wrist-mounted cameras to support egocentric teleoperation, while the triggers on the handheld controllers command the grippers' opening and closing.

B.3 Teleoperation-Free Data Collection Hardware

For teleoperation-free data collection, we use the HuMI hardware setup [1]. As shown in Fig. 10, the setup consists of two GoPro-mounted UMI-style handheld grippers [16] and five HTC VIVE Ultimate trackers [26], attached to the two grippers, the pelvis, and the two feet. Note that we adjust the grippers' color and shape to roughly match the robot-mounted grippers; however, differences in camera parameters, camera placement, and gripper opening width leave a domain gap. The system records wrist-view RGB observations, gripper widths, and synchronized 6-DoF tracker poses.

During collection, the demonstrator performs the task using the handheld grippers while a laptop records the synchronized tracker poses. Additionally, an online IK preview visualizes the corresponding humanoid motion in real time, helping the collector adjust demonstrations so that the recorded trajectories remain feasible for the robot.

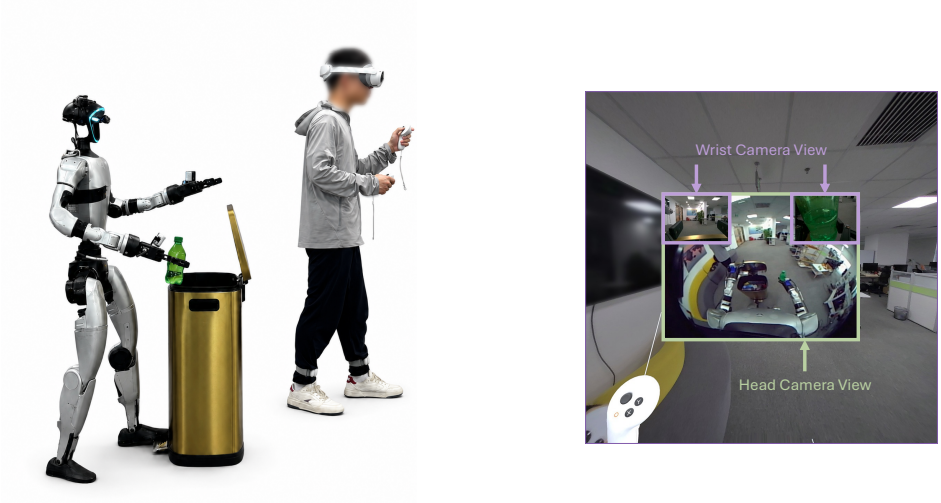


Figure 11: **Whole-body teleoperation scene and HMD snapshot, of the same frame. Left**: the PICO4U kit provides the HMD, two handheld controllers, and two leg trackers for live teleoperation. **Right**: an in-headset egocentric camera view streamed to the operator during teleoperation. The fisheye head view is placed in the center, with the 2 wrist views placed on its top-left and top-right corners.

C Implementation Details

C.1 Low-Level Controller & Teleoperation

This section provides further details on the teleoperation interfaces ablated in §3.1. All four interfaces use the same visual observations, consisting of stereo RGB images from the head camera and stereo RGB images from the two wrist cameras. They also share the same proprioceptive state format: a 32-D body state containing root roll, root pitch, yaw angular velocity, and the 29 measured body joints of the Unitree G1. The two grippers are recorded separately as one scalar for each hand. The main difference across interfaces is therefore the body-action format: each interface exposes a different command space to the operator, which in turn determines what behaviors can be demonstrated and what action representation the VLA must learn.

Decoupled control teleoperation. Decoupled control is widely used in recent humanoid loco-manipulation systems [2, 3, 4, 5, 6, 7, 8]. It separates manipulation and locomotion. The operator commands the waist and two arms for manipulation, while a lower-body controller receives coarse mobility commands and produces the leg motion. This interface is well suited to tasks that can be viewed as mobile manipulation: the robot walks to a workspace, uses its arms, and optionally changes its root height. It is less suitable for behaviors that require direct leg or foot participation,

because the operator does not command individual lower-body joints. The recorded body action is 21-D: 3 waist joint targets, 7 left-arm joint targets, 7 right-arm joint targets, 1 base-height command, and a 3-D planar base-velocity command. The base-velocity command represents forward velocity, lateral velocity, and yaw angular velocity.

VR 3-point teleoperation. VR 3-point teleoperation follows the sparse head-and-hands interface adopted by prior whole-body tracking systems such as OmniH2O and SONIC [12, 13]. It uses a sparse task-space interface. The operator specifies the poses of the two wrists and a head/neck reference point, together with a planar navigation command. A motion planner then infers the lower-body motion needed to realize the planar navigation command. This format is lightweight and natural for arm-centric tasks, especially reaching and grasping, but it provides only indirect control over torso, knees, and feet. The recorded body action is 24-D: a 3-D planar base-velocity command, 9 position coordinates for the three tracked points, and 12 orientation coordinates for the same three points. Each tracked point is represented by a 3-D position and a 4-D quaternion, ordered as left wrist, right wrist, and head reference point.

Joint-based whole-body teleoperation. Joint-based whole-body teleoperation follows recent efforts that collect full-body humanoid demonstrations through motion retargeting [29, 30, 31]. In our implementation, we capture human motion with a PICO VR setup [19] and retarget it to the robot online using GMR [20]. Unlike sparse keypoint interfaces, it gives the operator direct control over the robot's full kinematic chain, including arms, waist, legs, and root motion. This makes it suitable for the whole-body behaviors: squatting to expand the reachable workspace, coordinating locomotion with manipulation, and using a foot or leg as an active manipulator. The recorded body action is 32-D: root roll, root pitch, yaw angular velocity, and 29 robot joint targets. The 29 joint targets consist of 14 arm joints, 12 leg joints, and 3 waist joints, in the Unitree G1 joint order used by the low-level controller.

SMPL-based whole-body teleoperation. SMPL-based whole-body teleoperation uses the SMPL human-body representation [21], which is also supported as an input format by recent whole-body tracking controllers [13]. It keeps the action in a human-body representation instead of immediately converting the demonstration into robot joint targets. This representation is natural when demonstrations come from human motion capture and is useful for testing whether a human-pose action space can serve as a policy target. However, it is substantially higher-dimensional than the robot-joint representation, and many of its coordinates are redundant. The recorded body action is 81-D: root roll, root pitch, yaw angular velocity, a 6-D wrist/arm refinement term, and 72 SMPL joint coordinates from 24 human-body joints.

C.2 Whole-Body VLA Policy Design

This section provides further details on the design choices ablated in §3.2 (Fig. 5). We organize the discussion into three families: action and proprioception interface, pretraining ablations, and one-step action generation.

Action and proprioception interface. We ablate four design choices around the interface between the VLA and the humanoid's action and proprioceptive state.
Action projection initialization. $\pi_{0.5}$'s action expert contains two linear projection layers: an input projection that maps the noisy action chunk from action-space dimension to the transformer embedding dimension, and an output projection that maps back. The pretrained projections support up to 32 action dimensions, but our whole-body action vector is 34-D (32 dims from §3.1 plus two parallel-jaw gripper dimensions; a dexterous hand would push this higher), so both projections must be resized. *Random re-initialization* re-initializes both weight matrices from scratch at the new 34-D shape, discarding the pretrained projection weights entirely. *Weight surgery* (default) copies the pretrained weights into the first 32 rows/columns of the new matrices and Xavier-initializes only the remaining new entries, preserving the learned action representation for faster, more stable training.

Action ordering. $\pi_{0.5}$'s pretrained action layout is bimanual, with arms and grippers interleaved per side: [left arm (7), left gripper (1), right arm (7), right gripper (1)]. The humanoid adds 18 dimensions (legs, waist, root pose) that have no counterpart in this layout, so we must decide where to place them. *Pretrained bimanual ordering with humanoid joints appended* (default) keeps the pretrained slots fixed and concatenates the new dimensions at the end, giving the 34-D vector [left arm (7), left gripper (1), right arm (7), right gripper (1), left leg (6), right leg (6), waist (3), root roll/pitch + yaw rate (3)]. *Humanoid-native ordering* instead places the root and lower body first, [root roll/pitch + yaw rate (3), left leg (6), right leg (6), waist (3), left arm (7), right arm (7), left gripper (1), right gripper (1)]. This is more natural from a humanoid-control perspective but moves the pretrained dimensions to new positions in the action vector.
Absolute vs. relative action targets. For the 29 actuated joints of the Unitree G1 (dual-arm 14 + dual-leg 12 + waist 3), an action chunk admits two parameterizations. *Absolute targets* (default) expresses each action as absolute joint positions. *Relative targets* expresses each action in the chunk relative to the first state of that chunk ($a_t - s_0$).
Proprioceptive input. The humanoid's proprioceptive state is a 34-D vector with the same layout as the action vector. *With proprioception* (default) feeds this vector into the VLA alongside vision and language, giving the policy direct access to its own body pose, which is otherwise hard to recover since the head- and wrist-mounted cameras do not cleanly observe the lower body. *Without proprioception* removes this input; the policy must infer body pose from vision alone.

Pretraining ablations. We compare three backbone initializations to isolate the contribution of robot pretraining.
$\pi_{0.5}$ (default). $\pi_{0.5}$ [22] is one of the strongest open-source robot VLA models. Architecturally it consists of a pretrained VLM (PaliGemma) plus an action expert that handles robotics-specific inputs and outputs. The full network is pretrained on data from multiple robots, high-level semantic prediction tasks, web data, and other sources to enable broadly generalizable real-world manipulation. Its robot data, however, consists primarily of static or wheeled dual-arm platforms; no humanoid data is included.
PaliGemma. PaliGemma [23] is the open-source vision-language model on which $\pi_{0.5}$ is built. In this ablation we initialize the VLM portion of the VLA with PaliGemma weights and randomly initialize the action expert, removing all robot-specific pretraining while retaining vision-language representations.
Random initialization. The entire network (VLM and action expert) is randomly initialized with no pretraining of any kind.

One-step action generation. We compare two ways to produce an action chunk in a single forward pass of the action expert.
One-step flow matching. Training follows the standard flow-matching recipe of $\pi_{0.5}$. At inference, instead of integrating the learned vector field over multiple steps, we integrate from $\tau = 0$ to $\tau = 1$ in a single step (integration step count set to 1). No retraining is required; this is purely an inference-time change.
Drifting model. The drifting model [25] is a generative model that learns a pushforward map evolving during training, removing the need for iterative inference and naturally admitting one-step generation. The original paper applies it to robotic control as Drifting Policy, which matches or exceeds the state-of-the-art Diffusion Policy [35] (which uses 100-step diffusion-based inference) on single-task visuomotor imitation. We extend it to the multi-task VLA setting by replacing the flow-matching action expert with a drifting-model action expert, keeping the rest of the architecture and the VLM backbone unchanged.

C.3 Heterogeneous Co-Training

Stationary teleoperation demonstrations. Stationary teleoperation uses the same robot and joint-based whole-body interface as our full loco-manipulation demonstrations, but collects only the in-place manipulation portion of a task. During collection, we start after the robot has reached the

workspace and stop before it needs to leave it, avoiding the approach and departure motions that require locomotion. The robot therefore stays at the same floor location, while the demonstration still contains arm and gripper manipulation and on-the-spot posture adjustments, such as torso and height changes (e.g., `Shelf Cube Transfer`) or in-place turns (e.g., `Pig Placement`). This preserves same-embodiment manipulation data while making collection substantially easier than full loco-manipulation teleoperation, since the operator does not need to teleoperate the robot through the walking portions of each demonstration.

Stationary HuMI demonstrations. HuMI collects the same stationary manipulation phases as above, but without the humanoid in the loop, making it faster and cheaper than on-robot teleoperation. After collection, we run offline IK on the recorded tracker trajectories to convert them into whole-body humanoid joint targets. We then package the resulting trajectories in the same state and action format as our joint-based teleoperation data, allowing HuMI episodes to be mixed with teleoperated episodes without any model-side change. For the visual observations, we rectify the GoPro wrist images to roughly reduce the fisheye mismatch, but their wrist views remain noticeably wider than the robot-mounted RealSense views, leaving a substantial visual gap. Finally, to approximate the 0.2 s future-frame preview latency used in our teleoperation pipeline, we shift the HuMI actions forward by the same amount when forming state-action pairs (with perfect tracking under this latency, a command at the current time corresponds to the robot state 0.2 s later).

C.4 Hyperparameters

Table 3 lists the default hyperparameters for training `OpenHLM`. Training on 4 A800 GPUs takes approximately 24 hours. For data augmentation, we apply random crop and random rotation to non-wrist camera observations (head-mounted camera), and random brightness, random contrast, and random saturation to all observation images.

At inference, the VLA generates a 50-step action chunk at 30 Hz and sends whole-body commands to the low-level SONIC controller. We execute up to 25 actions before running inference again, meaning inference runs every 5/6 seconds.

Hyperparameter	**Value**
Optimizer	AdamW
Optimizer momentum	$\beta_1 = 0.9,\ \beta_2 = 0.95$
Peak learning rate	1×10^{-4}
Learning rate schedule	Cosine decay
Warmup steps	1k
Training steps	30k
Batch size	128
Action horizon	50
Observation resolution	224×224

Table 3: **Training hyperparameters.** Default configuration for `OpenHLM` VLA training.

D System-Level Comparison on Long-Horizon Tasks

D.1 Task Details & Evaluation Protocol

We provide the detailed settings of the long-horizon task set used for the system-level comparison in Section 4.

Long-horizon Task: `Fruit Arrangement`

- **Language Prompt:** Pick the {fruit 1} with the right hand, pick the {fruit 2} with the left hand, and place them on the shelf.
- **Detailed Process:** The robot is required to pick up two designated fruits from a set of distractor fruits using both hands, and then place them into two target containers on the top level of a shelf. Specifically, the robot first picks {fruit 1} from a low coffee table at a height of 40 cm using its right hand, and then picks {fruit 2} from a medium-height table at a height of 70 cm using its left hand. The target containers are placed on the top level of the shelf at a height of 95 cm. These surfaces span a wide range of heights within the Unitree G1 humanoid's whole-body reachable workspace. The robot must turn and walk to appropriate positions beside each platform before performing the corresponding manipulations, so that the target objects are brought into its reachable range. The robot starts several steps away from the first table and is required to return to the starting location after completing all manipulation steps, forming a closed-loop task sequence for experimental convenience.
- **Evaluation Rubric:** The task is evaluated with a total of 8 points:
  - The robot successfully walks to an appropriate position beside the low coffee table at a height of 40 cm. (1 pt)
  - The robot kneels down and correctly picks up {fruit 1} among the distractor fruits on the low coffee table using its right hand. (1 pt)
  - The robot stands up and successfully walks to an appropriate position beside the medium-height table at a height of 70 cm. (1 pt)
  - The robot correctly picks up {fruit 2} among the distractor fruits on the medium-height table using its left hand. (1 pt)
  - The robot turns toward the shelf and adjusts to an appropriate pose for placing the fruits. (1 pt)
  - The robot raises its right hand above the top level of the shelf and releases {fruit 1} into the blue plate. (1 pt)
  - The robot raises its left hand above the top level of the shelf and releases {fruit 2} into the orange basket. (1 pt)
  - The robot turns around and returns to the starting location. (1 pt)
- **Capability Stressed:** *Long-horizon compositional loco-manipulation*.

The mentioned 'fruit 1' and 'fruit 2' are chosen from {banana, peach, mangosteen, lemon, tomato}, producing $P(5, 2) = 20$ tasks in total, and during data collection, we collect 6 demos for each pair, with each of their language prompts filled by the specific fruit names, and thus produce 120 demos in total. Except that our `OpenHLM` (HuMI co-training) adopts only 36 demos (6 pairs) from them and replaces the remaining 84 demos involving held-out fruits (*i.e.*, lemon and tomato) by HuMI demonstrations.

As for evaluation, for each method we test 10 out of 20 pairs that are uniformly chosen as listed in Table 4. The scene configuration including layout of distractor fruits are recorded to ensure a fair comparison across methods. The performance is evaluated by the average task progress of these 10 rollouts.

rollout index	0	1	2	3	4	5	6	7	8	9
fruit 1	banana	peach	mangosteen	lemon	tomato	banana	peach	mangosteen	lemon	tomato
fruit 2	peach	mangosteen	banana	peach	banana	lemon	tomato	lemon	tomato	mangosteen

Table 4: Chosen pairs for long-horizon task evaluation. Each fruit appears as 'fruit 1' and 'fruit 2' twice. Held-out fruits (*i.e.*, lemon and tomato) of `OpenHLM` (HuMI co-training) are highlighted.

D.2 Baselines & Data Collection

GR00T N1.6 [7] is a humanoid VLA built on a Cosmos-2B vision-language backbone [27] and a diffusion-transformer action head. For the released Unitree G1 loco-manipulation setup, the policy observes an egocentric camera stream and proprioception from the legs, waist, arms, and hands. It outputs a decoupled command at each future step: 14-D relative arm commands, 14-D absolute hand commands, 3-D absolute waist commands, a 1-D base-height command, and a 3-D navigation command. Its non-hand command channels correspond to the 21-D decoupled-control interface in §3.1 (arms, waist, base height, and navigation), with the difference that GR00T predicts relative rather than absolute arm commands. The low-level controller then executes these commands while handling balance and locomotion.
Modifications. We preserve GR00T's relative arm representation and the same decoupled body-command interface, but replace the original 14-D hand block with two absolute parallel-gripper width commands for our hardware. We also extend the visual input from the original ego view to three views by adding the left and right wrist cameras. We collect the full loco-manipulation demonstrations for all 20 fruit pairs using this decoupled-control teleoperation setup, yielding 120 demonstrations for fine-tuning.
Training details. We fine-tune from the official GR00T N1.6 pretrained checkpoint for 50k optimization steps with a global batch size of 32, while keeping the official GR00T fine-tuning recipe, including the optimizer, learning-rate schedule, warmup, and 50-step action horizon.

Ψ_0 [8] is a humanoid VLA that first pretrains a Qwen3-VL [54] backbone on large-scale egocentric human videos and then post-trains a flow-based MM-DiT action expert on humanoid robot demonstrations. It outputs a 36-D decoupled command at each future step: 14-D hand commands, 14-D arm commands, 3-D torso-orientation commands, a 1-D base-height command, and a 4-D locomotion command consisting of forward velocity, lateral velocity, yaw rate, and target yaw. These commands are executed by AMO [6], an off-the-shelf RL tracking controller.
Modifications. Using the native Ψ_0 pipeline end-to-end would require recollecting the baseline dataset, since its controller interface differs from the GR00T decoupled-control interface used above. To keep the comparison matched while avoiding an additional round of time-consuming data collection, we adapt Ψ_0 to the same decoupled-control demonstrations used for GR00T N1.6. We keep the 36-D Ψ_0 tensor layout but fill it with our controller fields: the left and right gripper widths occupy one slot in each 7-D hand-command block; the unused hand-action dimensions are masked during training, while the corresponding state dimensions remain zero-padded; the 14-D arm-command block uses the left and right arm commands; the 3-D torso-orientation command and 1-D base-height command use the corresponding controller commands; and the 4-D locomotion command uses our 3-D navigation command, with the target-yaw slot left unused. Thus, the model is trained only on action dimensions present in our controller interface. At deployment, we execute Ψ_0 through the same GR00T decoupled-control stack by converting the predicted 36-D action chunks back into the controller command dictionary.
Training details. We initialize the VLM backbone and action expert from the released Ψ_0 pretrained checkpoints, then fine-tune on the same 120 decoupled-control demonstrations for 50k optimization steps with a global batch size of 64. We follow the released Ψ_0 fine-tuning recipe, including bf16 training, optimizer settings, 30-step action chunks, and RTC.

E Additional Experimental Results

E.1 Per-Task Results on the HLM-12 Benchmark

Figure 12 shows per-task results on the HLM-12 benchmark for the four conditions evaluated in §3.3. All 12 tasks appear individually, plus three aggregate bars (rightmost): Tasks 1–8 (training), Tasks 9–11 (motion-reuse held-out), and Tasks 9–12 (all held-out). Four conditions are compared: 8-task baseline, stationary co-training, HuMI co-training, and 12-task oracle.

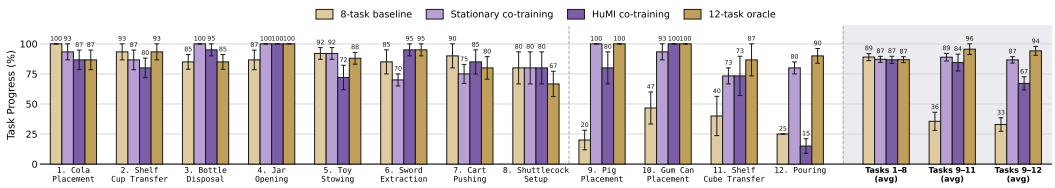


Figure 12: **Per-task breakdown of heterogeneous co-training results.** Task progress for all 12 tasks and three aggregates (rightmost). Four conditions: 8-task baseline, stationary co-training, HuMI co-training, and 12-task oracle. Co-training does not regress training tasks. On held-out tasks, both methods reach near-oracle on motion-reuse tasks (9–11); stationary succeeds on the new-motion task (12), HuMI does not.

On training tasks (1–8), all methods maintain similar performance; co-training does not regress the base policy. On held-out tasks (9–12), the gap is stark. For motion-reuse tasks (9–11), both co-training methods reach near-oracle performance, supplying new semantic understanding (new objects and prompts). On Task 12 (`Pouring`), which requires a new motion (vessel tilt), stationary co-training matches the oracle while HuMI co-training fails, consistent with the visual and action gaps discussed in §3.3.

E.2 Scaling HuMI Demonstrations

Figure 13 examines how HuMI demonstration count affects performance on the three motion-reuse held-out tasks (Tasks 9–11), with the 8-task whole-body teleop set fixed.

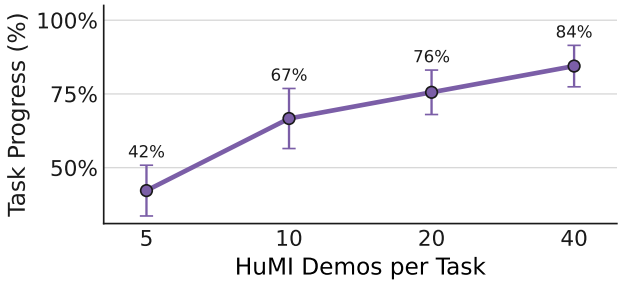


Figure 13: **Scaling HuMI demonstrations per task.** Average task progress on the three motion-reuse held-out tasks (Tasks 9–11) as we vary the number of HuMI demonstrations per task, with the 8-task whole-body teleop set fixed. Performance climbs from 42% at 5 demos to 84% at 40 demos.

Performance scales from 42% at 5 demos to 67% at 10, 76% at 20, and 84% at 40. The largest gain comes between 5 and 10 demos (+25 points); returns diminish thereafter (+9 from 10 to 20, +8 from 20 to 40). Even 5 demonstrations per task provide substantial signal, confirming HuMI's effectiveness at supplying semantic understanding for motion-reuse tasks. These experiments focus on motion-reuse tasks where HuMI already succeeds; scaling HuMI data to enable motion transfer (e.g., Task 12, `Pouring`) is an interesting direction for future work that we plan to explore.