BRIDGING VISION-LANGUAGE-ACTION MODELS AND WHOLE-BODY CONTROLLER VIA STRUCTURED ACTION INTERFACES

Anonymous authors
Paper under double-blind review

ABSTRACT

Quadrupedal loco-manipulation systems remain underexplored. Extending humanoid vision-language-action (VLA) pipelines is not straightforward: this composite embodiment has no direct kinematic counterpart from which motion references can be retargeted, raising two open questions: how an arm-body-coordinated whole-body controller should be derived, and how it should be bridged with a VLA policy using an appropriate action representation. In this paper, firstly, we propose an arm-body-coordinated whole-body controller to serve as an embodiment-specific motion generator. The use of such a native tracking interface inevitably introduces irreducible tracking errors, making it unsuitable for direct VLA integration. Thus, we transfer this generated coordination into a structured controller with substantially lower error, coordinating the upper and lower body explicitly through the degrees of freedom at the arm mount, and we subsequently learn a VLA policy using this structured interface. The resulting system, *QuadArm-VLA*, spans a wide range of mobile, contact-rich, and constrained manipulation tasks on a Unitree Go2 quadruped equipped with an ARX X5 arm. Our results provide practical guidance for acquiring coordinated motion and constructing VLA action interfaces for embodiments without a natural kinematic counterpart, and we will open-source our system codes to aid future research.

1 INTRODUCTION

Recent humanoid loco-manipulation systems combine high-level visuomotor planning with low-level control to execute language-conditioned mobile manipulation tasks (Jiang et al., 2026; Wei et al., 2026). Their success depends in part on exposing an action interface that the low-level controller can track accurately while maintaining balance. Many existing systems construct this interface from human motion: motion-tracking controllers are trained on retargeted motion-capture (MoCap) data (Luo et al., 2026; Liao et al., 2026; Hu et al., 2026), while other approaches encode retargeted motion or ego-centric human videos into latent representations that communicate upper- and lower-body coordination (Jiang et al., 2026; Lu et al., 2025; Xue et al., 2025). These approaches either exploit the kinematic correspondence between humans and humanoids or assume that suitable whole-body motion references can be constructed for the target embodiment.

We focus on building a comparable system for a quadruped equipped with a robotic arm, a composite embodiment with the potential for a larger manipulation workspace and improved terrain traversal but with no direct kinematic counterpart in nature. Directly applying a humanoid control pipeline would leave two interfaces unspecified: *how to obtain embodiment-specific references that capture arm-body coordination*, and *how to expose this coordination as an action space that a VLA can predict and execute precisely*. These interface-construction problems distinguish quadrupedal whole-body VLA learning from directly transferring an existing humanoid policy, and we organize our contributions around resolving them.

We first address how such references can be obtained at all. We train a whole-body reinforcement learning (RL) policy that maps desired end-effector poses and base velocities to joint angles. We use this policy as an embodiment-specific motion generator: through interaction with the simulated robot dynamics, it produces feasible trajectories containing coordinated arm configurations and body

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

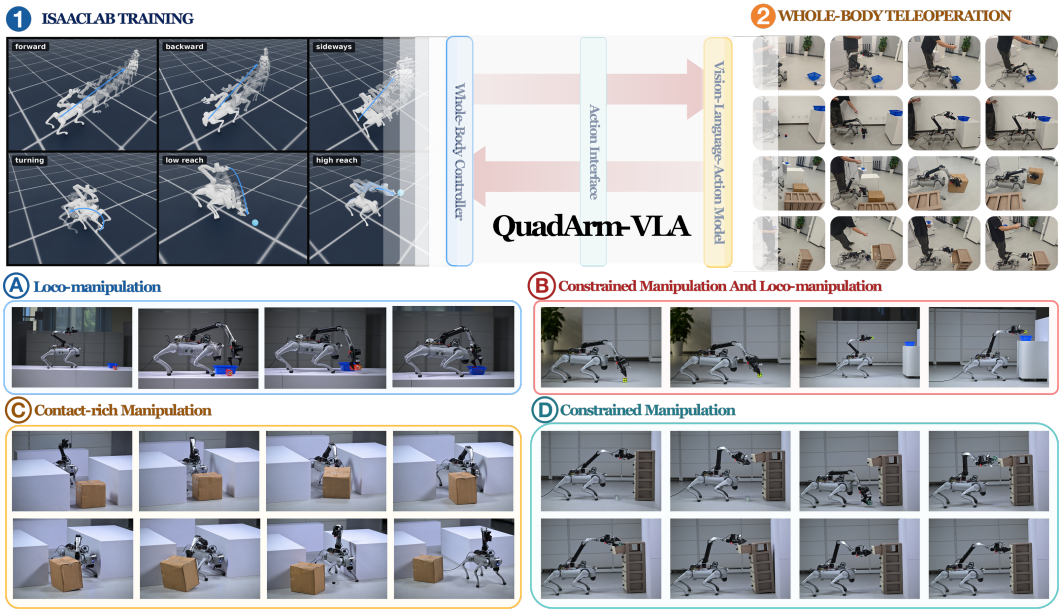


Figure 1: **Overview of QuadArm-VLA.** We acquire embodiment-specific arm-body coordination through large-scale IsaacLab training and collect whole-body demonstrations through teleoperation. A pretrained vision-language-action model is then adapted to the resulting controller interface. QuadArm-VLA supports (A) mobile pick-and-place, (B) small-object transport and elevated placement, (C) contact-rich box pushing around an obstacle, and (D) drawer opening, object placement, and drawer closing. For more demonstrations, see project website: quadarm-vla.site.

postures. We identify three design choices that improve this "MoCap policy": training with simultaneous locomotion and manipulation, using separate value estimates for these two tasks (Fu et al., 2023), and augmenting the command distribution with reachable end-effector poses (Portela et al., 2025). With these choices, our whole-body controller achieves state-of-the-art (SOTA) tracking precision. However, we identify two critical issues that hinder the direct application of this controller: tracking precision and tracking latency.

We therefore transfer the generated coordination into a structured decoupled whole-body controller for precise execution. The decoupled action interface design is consistent with common practice in the humanoid literature (Lu et al., 2025; Jiang et al., 2026; Xue et al., 2025); however, in our setting the coordination being decoupled is not given by a MoCap reference but discovered by the whole-body policy, and we further show that this coordination is physically grounded through the degree of freedom at the arm mount. Despite this risk, we show that the discovered coordination can be losslessly re-expressed in decoupled form while enabling substantially more precise execution, reducing end-effector tracking error from $5.87\,\mathrm{cm}$ to $0.48\,\mathrm{cm}$.

This structured action interface is precise and low-dimensional by construction, which makes it a suitable target for a whole-body VLA to predict directly, an interface prior quadrupedal manipulation systems lack. We fine-tune a VLA model to predict our structured interface directly, and provide insights into adapting a VLA model to this new embodiment, particularly regarding the choice of execution interface and the failure modes introduced when inference latency causes the robot to fall. We instantiate the complete system on a Unitree Go2 equipped with an ARX X5 arm and evaluate it on mobile pick-and-place, contact-rich manipulation, and constrained manipulation tasks. In summary, our contributions are fourfold:

- **A framework for obtaining arm-body coordination.** We formulate whole-body reinforcement learning as a means of generating coordinated motion references for a quadrupedal manipulator, and identify simultaneous loco-manipulation training, separate value estimates, and reachable-pose augmentation as effective design choices for improving this coordination.

- **A structured action interface for precise execution.** We show that the discovered arm-body coordination can be re-expressed in a structured, decoupled format, reducing tracking error by approximately 90% and exposing an action interface with low tracking error suitable for the downstream VLA system.
- **A VLA system built on this interface.** We introduce *QuadArm-VLA*, a whole-body VLA system predicting the structured action interface directly to achieve quadrupedal loco-manipulation.
- **Extensive real-world evaluation.** Through experiments across mobile, contact-rich, and constrained manipulation tasks, we show the resulting system is capable of performing a wide range of real-world tasks.

2 Related Works

Whole-body controllers. Whole-body control coordinates manipulation and locomotion while maintaining dynamic stability. Humanoid controllers commonly obtain this coordination by retargeting human motion and training reinforcement-learning policies to track the resulting references (He et al., 2025; Luo et al., 2026; Liao et al., 2026). Other systems control the upper body through inverse kinematics and condition the lower-body policy on upper-body motion or learned latent representations (Lu et al., 2025; Jiang et al., 2026; Xue et al., 2025). These approaches benefit from the structural correspondence between human and humanoid motion.

Quadrupedal manipulators lack a directly retargetable human counterpart. Prior methods therefore discover arm-base coordination through reinforcement learning, using unified (Fu et al., 2023), mutually conditioned (Pan et al., 2025), or hierarchical tracking policies (Liu et al., 2025; Ha et al., 2025; Wang et al., 2025; 2026). Their action interfaces commonly combine end-effector targets with base-motion commands, leaving the low-level policy to resolve the corresponding whole-body configuration. Our work follows this stream and employ RL to train a whole-body controller that discovers feasible arm-body coordination, and take a step further to transfer this coordination into a structured controller designed for precise execution.

Whole-body VLA systems. General-purpose VLA models map visual observations, language instructions, and proprioceptive state to continuous robot actions (Black et al., 2026; Intelligence et al., 2025; NVIDIA et al., 2025). Recent humanoid systems extend this formulation to loco-manipulation through different low-level action interfaces. LeVERB (Xue et al., 2025) communicates with a reactive controller through latent motion instructions; WholeBodyVLA (Jiang et al., 2026) predicts latent actions that encode coordination; and OpenHLM (Hu et al., 2026) predicts a joint-based whole-body action space learned from robot teleoperation. These systems demonstrate that the interface between the VLA and the whole-body controller determines which coordination information can be represented and reproduced.

For quadrupedal manipulators, despite extensive efforts to make them autonomous, recent works are still constrained to specific tasks. This is due to either entangling of low-level controller and high-level planner (Ha et al., 2025) or task-specific design heuristics (Liu et al., 2025; Wang et al., 2025; 2026; Zhang et al., 2026), hindering the generalizability of such systems. Identifying this limitation, our work connects learned arm-body coordination, precise execution, and language-conditioned control to establish a generalizable system for quadrupedal manipulators.

3 Method

3.1 Arm-body Coordination in a Whole-body Controller

This composite embodiment has no natural kinematic counterpart, so we need a controller that (i) exposes a simple, efficient interface for human teleoperation and (ii) incorporates the leg joints into the arm's kinematic chain to enable active arm-body coordination.

To this end, we design a whole-body reinforcement learning policy π_{wbc} (Fig. 2A) that jointly controls the full 18-DoF system, whose leg and arm joint angles we denote $q^L \in \mathbb{R}^{12}$ and $q^A \in \mathbb{R}^6$. At each control step, π_{wbc} receives a command

$$c_t = \left(v_x^*,\, v_y^*,\, \omega_z^*,\, p_{\mathrm{ee}}^*,\, R_{\mathrm{ee}}^*\right), \tag{1}$$

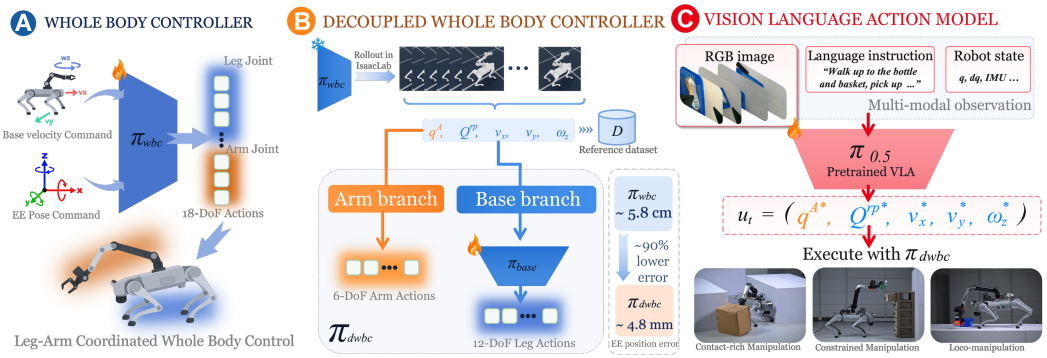


Figure 2: System overview. (A) The whole-body RL controller π_{wbc} jointly controls all 18 DoFs, serving as both the teleoperation interface and the motion generator for training the decoupled controller. (B) The decoupled controller π_{dwbc} re-expresses this coordination via a fixed-gain arm branch and a learned base branch π_{base}, cutting end-effector tracking error by $\sim$90%. (C) The pretrained VLA $\pi_{0.5}$ is fine-tuned to predict the structured action interface u_t, executed by π_{dwbc}.

specifying the desired base planar velocity and yaw rate together with a target end-effector position and orientation expressed relative to the robot's current base position and heading. The policy outputs a residual joint offset $a = (a^L, a^A) \in \mathbb{R}^{18}$ around a default posture $q_0 = (q_0^L, q_0^A)$, converted into joint position targets tracked by fixed-gain PD executors at each joint. We use three complementary designs that promote leg-arm coordinated behavior and optimize tracking precision.

First, π_{wbc} is trained to track a nonzero base-velocity command $(v_x^*, v_y^*, \omega_z^*)$ simultaneously with the end-effector target, rather than tracking it with the base held stationary; this joint training of locomotion and manipulation grounds precise end-effector tracking in an actively stabilized, moving base. Second, following (Fu et al., 2023), we split the reward $r_t = r_t^L + r_t^A$ into a base/leg channel and an arm/end-effector channel, each with its own value estimate V_L, V_A and advantage estimate $\hat{A}_t^L, \hat{A}_t^A$, disentangling credit assignment between locomotion and manipulation; the leg and arm branches nonetheless share the combined advantage $\hat{A}_t = \hat{A}_t^L + \hat{A}_t^A$ in the PPO objective, retaining only separate probability ratios ρ_t^L, ρ_t^A, so both branches still optimize toward the same joint objective. Third, targets $(p_{\text{ee}}^*, R_{\text{ee}}^*)$ are drawn from a precomputed reachable-pose set with local perturbation, $p' = p + R\delta p$, $R' = R\exp([\delta\theta]_\times)$ with $\delta p, \delta\theta$ being small random translations and rotations (Portela et al., 2025), augmented with hand-designed difficult targets that prompt leg-joint readjustment—without breaking foot contact—to extend the effective reach.

π_{wbc} achieves SOTA end-effector tracking precision among RL-based whole-body controllers for this embodiment, but the VLA trained directly on this interface underperforms because of its residual error and response delay. We therefore use π_{wbc} instead in two roles in the final system: as the teleoperation interface for data collection, and as a motion generator whose rollouts train the decoupled tracking policy of Sec. 3.2.

3.2 Decoupled Whole-Body Controller

This motivates a narrower question than how to control this composite embodiment: can the coordination in π_{wbc} be *re-expressed* through an interface that is easier to track precisely? For the arm, a fixed-gain joint-position controller already tracks joint targets with high precision and negligible latency, so allocating additional RL capacity to it yields no benefit; tracking the base is far harder, since it involves rich contact interaction and must continuously counteract the reaction forces and center-of-mass shifts induced by the arm's own motion. This coordination in fact has a clear physical meaning for our embodiment: the base's roll and pitch are exactly the degrees of freedom π_{wbc} learns to exploit when reaching a difficult target. We carry this idea into the re-expressed interface by letting base posture act on the leg joints as the arm's demand—so that even though the resulting policy outputs leg actions alone, it still operates on the same leg-arm kinematic chain underlying π_{wbc}'s whole-body coordination and inherits the coupled whole-body dynamics that coordination depends on. What we decouple is only how actions are generated, not this kinematic chain itself.

Guided by this, π_{dwbc} (Fig. 2B) consists of two parts: an arm branch executed directly by a fixed-gain joint-position controller, and a learned policy π_{base} that controls the legs. Given a structured action interface

$$u_t = \left(q^{A*},\ Q^{rp*},\ v_x^*,\ v_y^*,\ \omega_z^*\right), \tag{2}$$

where $Q^{rp} = \text{yaw}(Q)^{-1} \otimes Q$ denotes the base orientation quaternion Q with yaw removed, the arm branch tracks the absolute joint target $q^{A*} \in \mathbb{R}^6$ directly, and π_{base}, conditioned on a short history of proprioception that includes the arm's realized joint state q^A, outputs leg joint targets trained to track $(v_x^*, v_y^*, \omega_z^*)$ and Q^{rp*}.

Rollout collection. We build π_{base}'s training references by rolling out the frozen π_{wbc}, recording at each step its realized arm joint angles q^A, yaw-free base orientation Q^{rp}, and the velocity command it receives, into a reference set

$$\mathcal{D} = \{(v_{x,e,k}^*,\ v_{y,e,k}^*,\ \omega_{z,e,k}^*,\ q_{e,k}^A,\ Q_{e,k}^{rp})\}_{e,k}, \tag{3}$$

where (e, k) indexes the episode and timestep at which each quantity was recorded.

Training. π_{base} is trained online with PPO: each episode is initialized from a state sampled from $\mathcal{D}$, and the recorded velocity command and orientation are replayed as π_{base}'s tracking targets, while the recorded arm joint angles are continuously replayed to drive the arm's position controller; π_{base} interacts with the resulting simulated dynamics at every step. The reward combines velocity- and orientation-tracking terms, each active only when its command channel is non-trivial, with gait and regularization terms that keep the base stably supported.

3.3 QUADARM-VLA

Adapting a pretrained VLA to this composite embodiment (Fig. 2C) requires adapting its interface on both ends: on the input side, the observation must give a complete account of a proprioceptive state the model has not seen during pretraining; on the output side, the action head must be retargeted to the structured action interface of Sec. 3.2. We describe the resulting observation interface, the training procedure that supervises it, and the resulting action interface in turn.

Input interface. At each timestep the VLA receives front and wrist RGB images $I_t^{\text{front}}, I_t^{\text{wrist}}$, a language instruction ℓ, and a proprioceptive state

$$s_t = \left(r_t,\ p_t,\ \psi_t,\ q_t^{A\top},\ g_t,\ u_{t-1}^{\top}\right) \in \mathbb{R}^{13}. \tag{4}$$

r_t, p_t are the current arm-mount roll and pitch—the coordination channel around which the interface of Sec. 3.2 is built—ψ_t is the base yaw relative to the start of the episode, q_t^A is the realized arm joint configuration, g_t is the gripper position, and u_{t-1} is the previously issued base-velocity command. Together these channels form the observation $o_t = (I_t^{\text{front}}, I_t^{\text{wrist}}, \ell, s_t)$, giving a complete description of the coordination state introduced by this composite embodiment that $\pi_{0.5}$'s original training distribution does not otherwise cover.

Training. Training pairs (o_t, A_t) are built directly from demonstrations collected via π_{wbc} (Sec. 3.1): we record the arm joint angles, mount orientation, base command, and gripper position the robot actually realizes over the following H steps, forming the label chunk $A_t \in \mathbb{R}^{H\times 12}$ whose structure we specify below. We filter the resulting samples to discard those with desynchronized camera or state timestamps, extended segments in which the robot remains static and carries no useful supervision, and samples whose label chunk contains abrupt discontinuities or oscillatory artifacts. We then fine-tune $\pi_{0.5}$ (Intelligence et al., 2025) with a conditional flow-matching objective conditioned on o_t: for a ground-truth chunk A we sample noise ϵ and time $\tau \in [0, 1]$, form $X_\tau = \tau\epsilon + (1-\tau)A$, and train the model to predict the vector field

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{\tau,\epsilon}\left[\left\|v_\theta(X_\tau, \tau, o_t) - (\epsilon - A)\right\|_2^2\right]. \tag{5}$$

Keeping π_{base} (Sec. 3.2) balanced, and letting the base start and stop precisely during mobile manipulation, both require a continuous, responsive stream of commands, which makes closing the gap between chunks a necessity rather than a refinement: following a train-time action-conditioning scheme (Black et al., 2025), we additionally train the model to treat a random-length clean prefix of A as given context and predict only the remaining steps.

Output interface. The model predicts a chunk of H future commands

$$A_t = \big(\hat{u}_t, \ldots, \hat{u}_{t+H-1}\big), \quad \hat{u}_{t+i} = \big(q^A_{t+i+1},\, r_{t+i+1},\, p_{t+i+1},\, v^*_{x,t+i},\, v^*_{y,t+i},\, \omega^*_{z,t+i},\, g_{t+i+1}\big), \tag{6}$$

identical in form to the structured interface u_t of Sec. 3.2 with the gripper target appended: predicted arm joint targets are tracked by the fixed-gain arm controller, and predicted $(r, p, v^*_x, v^*_y, \omega^*_z)$ by π_{base}. At each control step the robot executes the next command in the chunk currently in flight; once a bounded number of steps have executed, it requests a new chunk conditioned on the yet-unexecuted suffix of the current one.

4 Experiments

Our experiments are organized around four research questions that follow the structure of our approach. **Q1:** How can embodiment-specific arm-body coordination be learned effectively? **Q2:** What interface should connect the high-level policy and low-level controller? **Q3:** What are the requirements to adapt a pretrained VLA to quadrupedal loco-manipulation? **Q4:** Can the resulting system support diverse real-world tasks and efficient adaptation to new skills?

4.1 Experiment Setup

Controller training. We model a Unitree Go2 equipped with an ARX X5 arm in IsaacLab (Mittal et al., 2025) and train two low-level policies. After training the arm-body coordination policy, we rollout this policy to collect coordinated arm-joint and arm-mount trajectories. We then train the structured tracking controller from these references. Detailed command distributions, training hyperparameters and simulation setup are provided in Appendix Section A.

Hardware platform. Real-world experiments use the corresponding Go2–ARX X5 embodiment with one front camera and one wrist camera. The VLA predicts six arm-joint targets, arm-mount roll and pitch, planar base-velocity and yaw-rate commands, and a gripper target. Arm-joint targets are executed by the fixed-gain arm controller, while the learned base controller tracks the commanded posture and locomotion. Detailed sensing, computing, and control-frequency specifications are provided in Appendix Section B.

Evaluation tasks. We evaluate the complete system on four tasks covering distinct coordination requirements. In *mobile pick-and-place*, the robot approaches a bottle, grasps it, and places it into a basket. In *open drawer*, it opens a drawer, places a bottle inside, and closes the drawer. In *push box*, it pushes a box around an obstacle through successive forward and lateral contacts. In *carry cube*, it grasps a small cube, carries it while walking, and places it on an elevated surface. Visual task description and detailed progress metrics are provided in Appendix Section D.

4.2 Q1: Learning Effective Arm–Body Coordination

Whole-body tracking. We compare against RoboDuet (Pan et al., 2025) on our reachable command set and its original test set (denote as "out-of-distribution (OOD) command set"). For notational convenience, let LM denote simultaneous loco-manipulation training, RS reachable-set augmentation, 2H the two-head value function, and RD RoboDuet.

As shown in Table 1, our method outperforms RoboDuet on every metric under both command distributions. On the reachable set, it reduces end-effector position and orientation errors by 70.4% and 80.7%, respectively. The advantage persists on the RoboDuet OOD set. Reachable-set sampling substantially improves RoboDuet's manipulation accuracy but degrades its locomotion tracking; our complete method remains better than RD+RS on all four metrics.

The ablations identify complementary roles for the three design choices. Removing simultaneous loco-manipulation training or the two-head value function markedly increases both end-effector errors, indicating that coordination benefits from jointly experiencing arm and base motion while maintaining separate learning signals for the two objectives. Reachable-set augmentation further reduces end-effector position and orientation errors, although it trades off some velocity-tracking accuracy, particularly in yaw.

Structured reference replay. Table 2 compares the original end-effector-commanded whole-body controller (EE-WBC) with the structured replay controller on reachable command set. Decoupling

Table 1: Whole-body end-effector tracking errors (lower is better).

Method	Lin. (m/s)	Ang. (rad/s)	EE Pos. (m)	EE Ori. (rad)
Reachable command set				
Ours	0.0964	0.0906	**0.0587**	**0.2650**
Ours w/o LM	0.1542	0.0953	0.1908	0.6100
Ours w/o RS	**0.0957**	**0.0690**	0.0628	0.3169
Ours w/o 2H	0.1171	0.0996	0.1554	0.5803
RD	0.1412	0.4161	0.1982	1.3705
RD+RS	0.1975	0.3843	0.0833	0.2913
RoboDuet OOD command set				
Ours	**0.1037**	**0.0961**	**0.1846**	**0.7440**
RD	0.1447	0.4212	0.2638	1.2310
RD+RS	0.2123	0.3947	0.2347	1.1549

Table 2: End-effector tracking errors on the reachable pose set (lower is better).

Controller	Lin. (m/s)	Ang. (rad/s)	EE Pos. (m)	EE Ori. (rad)
EE-WBC	0.0964	0.0906	0.0587	0.2650
Structured replay	**0.0859**	**0.0568**	**0.0048**	**0.0087**

arm execution from learned locomotion reduces mean end-effector position error from 5.87 cm to 4.8 mm and orientation error from 0.2650 rad to 0.0087 rad, corresponding to reductions of 91.8% and 96.7%, respectively. It also reduces linear- and angular-velocity errors by 10.9% and 37.3%. These results show that the coordination discovered by EE-WBC can be replayed considerably more precisely when arm kinematics are handled separately by the native arm controller, while reinforcement learning trains leg joints for body-orientation and locomotion tracking.

4.3 Q2: Choosing the VLA–Controller Interface

Evaluation protocol. We investigate two action interfaces, corresponding to two controllers introduced in Section 3. The *EE-command interface* predicts end-effector targets executed by the original whole-body controller, whereas the *structured joint interface* predicts arm-joint targets and arm-mount orientation for the structured decoupled controller. We evaluate both interfaces over eight trials on three tasks. We decompose each task into ordered stages and record task progress as the fraction of stages completed in a trial, where completing every stage would indicate full-task success.

Results. The structured joint interface substantially improves tasks that require precise arm configuration, as shown in Figure 3. Across the total 96 stages, it achieves 74.0% progress, compared with 41.7% for the EE-command interface. Across the 24 complete trials, full-task suc-

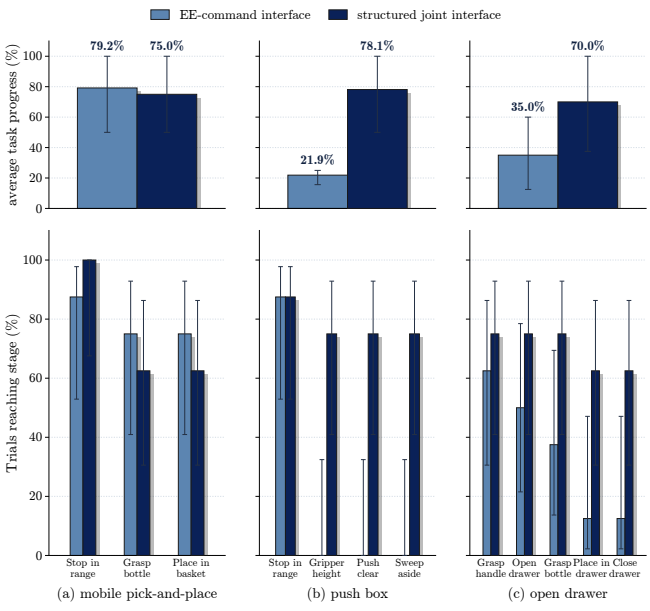


Figure 3: VLA performance under the EE-command and structured joint interfaces. Top: average task progress. Bottom: the percentage of trials reaching each ordered task stage. Error bars show 95% confidence intervals.

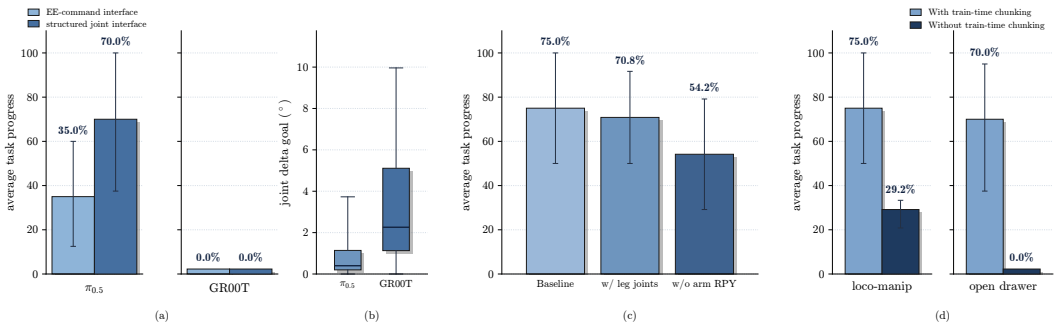


Figure 4: Ablations for adapting pretrained VLAs to quadrupedal manipulation. (a) Open drawer task progress for $\pi_{0.5}$ and GR00T N1.7 under the EE-command and structured joint interfaces. (b) Distribution of consecutive joint-target changes produced by each backbone; smaller changes indicate smoother commands. (c) Effect of proprioceptive state design on mobile pick-and-place. (d) Effect of train-time chunking on mobile pick-and-place and open drawer. Error bars show 95% confidence intervals.

cess increases from 29.2% to 66.7%. The largest difference occurs in contact-rich box pushing: the structured interface achieves 78.1% progress and 75.0% success, whereas the EE-command interface achieves 21.9% progress and never completes the task.

The stage-level results localize these improvements. Both interfaces usually position the robot within operating range for box pushing, but the EE-command interface never reliably establishes the required gripper height, manifesting as unable to place gripper between the box and the robot to establish pushing behavior.

Mobile pick-and-place presents a different pattern as this task tolerates larger tracking errors (as compared with open drawer) and repeated grasp attempts (as compared with push box). The EE-command interface actually outperforms the structured joint interface during grasp phases. The advantage of the structured interface therefore emerges primarily when success depends on accurately reproducing the demonstrated arm-body configuration.

Failure analysis. In box pushing, the EE-command interface positions the robot within operating range in seven of eight trials but never reaches the gripper-height stage. The task requires a vertically downward gripper pose, whereas the executed pose tends to point forward, causing the terminal link to strike the box and become blocked. The structured interface removes this systematic execution failure. Some failure cases for structured joint interface in open drawer task actually stems from high-level imperfections: VLA skipping drawer opening (due to lack of memory) or placing the bottle too close to the drawer boundary that hinders closure.

4.4 Q3: Adapting a Pretrained VLA to Quadrupedal Manipulation

Backbone choice. We post-train $\pi_{0.5}$ (Intelligence et al., 2025) and GR00T N1.7 (NVIDIA et al., 2025) on open drawer task using either EE commands or the structured joint interface (refer to Appendix Section C.2 for post-training details). As shown in Figure 4(a), $\pi_{0.5}$ achieves 35.0% progress with EE commands and 70.0% with structured joint actions. Surprisingly, GR00T makes no measurable progress under either interface, and we observe fast arm movement during real-world evaluation that destabilize the base. Panel (b) quantifies the observed instability. $\pi_{0.5}$ produces small consecutive changes in its joint targets, whereas GR00T has a roughly $5\times$ larger median change. These abrupt commands excite the quadruped base and produce visible body oscillation, which is detrimental in open drawer task where precise manipulation is required to grab the handle. $\pi_{0.5}$ produces substantially smoother commands and is therefore used for the remaining experiments.

Proprioceptive state. We next compare the proposed input state (arm joints, gripper position, mount roll/pitch, episode-relative base yaw, and previous base-velocity commands) with variants that add all 12 leg-joint positions or remove the arm-mount orientation. As shown in Figure 4(c), it achieves 75.0% progress on mobile pick-and-place. Adding the leg joints reduces progress to 70.8%, indicating that gait variables handled internally by the low-level controller introduce unnecessary high-frequency variation. Removing mount orientation reduces progress further to 54.2%. This variant removes body-orientation feedback, impairing arm-body workspace adjustment.

Temporal continuity. We use train-time chunking to maintain smooth, uninterrupted commands (Black et al., 2025). Without chunking, inference gaps either halt the moving quadruped or force it to execute stale commands, causing instability or overshooting. Abrupt arm commands also disturb the compliant base during stationary manipulation. Accordingly, removing chunking reduces mobile pick-and-place progress from 75.0% to 29.2% and drawer-manipulation progress from 70.0% to zero in Figure 4(d).

Together, these results identify three requirements for quadrupedal VLA adaptation: a suitable backbone for smooth action prediction, proprioception aligned with the exposed control interface, and chunked execution that bridges inference latency.

4.5 Q4: Multi-Task Mid-Training for Efficient Adaptation

Adaptation protocol. We investigate whether related loco-manipulation experience improves adaptation to a held-out task. The robot must approach a small cube, grasp it, transport it while walking, and place the cube into a high basket. *Direct adaptation* ($\pi_{0.5}$) fine-tunes $\pi_{0.5}$ on target-task demonstrations, whereas *mixture-initialized adaptation* (mid-trained $\pi_{0.5}$) first mid-trains the same checkpoint on mobile pick-and-place and open drawer data. We hypothesize that open drawer task provides prior about small object manipulation, and mobile pick-and-place task helps learn the carrying the placing process.

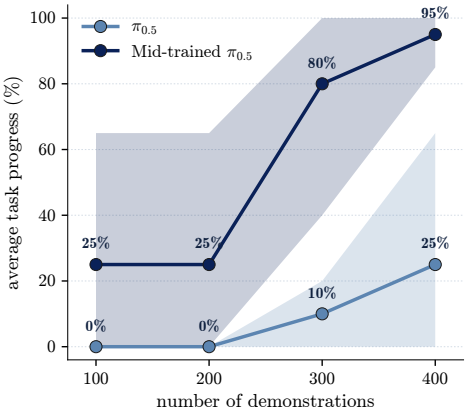


Figure 5: Adaptation efficiency on the held-out cube-transport task. Direct adaptation starts from $\pi_{0.5}$, while mixture-initialized adaptation starts from a checkpoint mid-trained on mobile pick-and-place and drawer manipulation. Curves show mean task progress over five trials; shaded regions show one standard deviation.

Sample efficiency. As shown in Figure 5, direct adaptation makes no measurable progress with 100 or 200 demonstrations and reaches only 25% progress with 400 demonstrations. Mixture initialization is able to increase to 80% with 300 demonstrations, and reaches near 100% with 400 demonstrations. At 300 demonstrations, four of five mixture-initialized trials complete the entire task, while none of the directly adapted trials do so.

The failure stages clarify this difference. Directly adapted policies often begin walking without grasping or contact the cube at an inaccurate location, causing it to be dropped, pushed away, or squeezed from the gripper. With 400 demonstrations, the only incomplete mixture-initialized trial reaches the stable-stop stage but fails during final placement. Mid-training therefore transfers base positioning, grasping, transport, and stabilization behaviors, leaving target-specific grasp localization and placement as the principal adaptation requirements.

Real-world task coverage. Representative rollouts in Figure 1 show mobile pick-and-place, drawer manipulation, contact-rich box pushing, and small-object transport. Together with the quantitative results, these rollouts demonstrate prehensile, constrained, contact-rich, and mobile manipulation on a quadrupedal manipulator.

5 Conclusion

We have presented QuadArm-VLA, a whole-body VLA system for quadrupedal loco-manipulation that addresses the fundamental challenge of operating composite embodiments without a natural kinematic counterpart. Our approach proceeds in three stages: first, we train a whole-body RL policy with arm-body coordination; second, we re-express this discovered coordination into a structured controller; and third, we fine-tune a VLA model to predict this structured interface directly, enabling extensive loco-manipulation capabilities on a Unitree Go2 with an ARX X5 arm.

AI use statement

In this work, we used generative AI tools to aid or polish the writing of this paper. We have not used generative AI tools for any other task. Every number reported in the paper was taken from an eval-

uation log or result file and cross-checked by the authors against the source. We have reviewed all AI-assisted writing in this paper. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Kevin Black, Allen Z Ren, Michael Equi, and Sergey Levine. Training-time action conditioning for efficient real-time chunking. *arXiv preprint arXiv:2512.05964*, 2025.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2026.

Zipeng Fu, Xuxin Cheng, and Deepak Pathak. Deep whole-body control: Learning a unified policy for manipulation and locomotion. In *Proceedings of The 6th Conference on Robot Learning*, pp. 138–149, 2023.

Huy Ha, Yihuai Gao, Zipeng Fu, Jie Tan, and Shuran Song. Umi-on-legs: Making manipulation policies mobile with manipulation-centric whole-body controllers. In *Proceedings of The 8th Conference on Robot Learning*, pp. 5254–5270, 2025.

Tairan He, Zhengyi Luo, Xialin He, Wenli Xiao, Chong Zhang, Weinan Zhang, Kris M. Kitani, Changliu Liu, and Guanya Shi. Omnih2o: Universal and dexterous human-to-humanoid whole-body teleoperation and learning. In *Proceedings of The 8th Conference on Robot Learning*, pp. 1516–1540, 2025.

Yingdong Hu, Haodong Zhu, Boyuan Zheng, Yihang Hu, Tong Zhang, Zunhao Chen, Junming Zhao, Ruiqian Nai, and Yang Gao. Openhlm: An empirical recipe for whole-body humanoid loco-manipulation. *arXiv preprint arXiv:2606.22174*, 2026.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: A Vision-Language-Action Model with Open-World Generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Haoran Jiang, Jin Chen, Qingwen Bu, Li Chen, Modi Shi, Yanjie Zhang, Delong Li, Chuanzhe Suo, wang chuang, zhihui peng, and Hongyang Li. Wholebodyvla: Towards unified latent vla for whole-body loco-manipulation control. In *International Conference on Learning Representations*, pp. 157438–157461, 2026.

Qiayuan Liao, Takara E Truong, Xiaoyu Huang, Yuman Gao, Guy Tevet, Koushil Sreenath, and C Karen Liu. Beyondmimic: From motion tracking to versatile humanoid control via guided diffusion. *Science Robotics*, 11(117):eadx8924, 2026.

Minghuan Liu, Zixuan Chen, Xuxin Cheng, Yandong Ji, Ri-Zhao Qiu, Ruihan Yang, and Xiaolong Wang. Visual whole-body control for legged loco-manipulation. In *Proceedings of The 8th Conference on Robot Learning*, pp. 234–257, 2025.

Chenhao Lu, Xuxin Cheng, Jialong Li, Shiqi Yang, Mazeyu Ji, Chengjing Yuan, Ge Yang, Sha Yi, and Xiaolong Wang. Mobile-television: Predictive motion priors for humanoid whole-body control. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5364–5371, 2025.

Zhengyi Luo, Ye Yuan, Tingwu Wang, Chenran Li, Fernando Castañeda, Sirui Chen, Zi-Ang Cao, Jiefeng Li, David Minor, Qingwei Ben, et al. Sonic: Supersizing motion tracking for natural humanoid whole-body control. *Science Robotics*, 11(117):eaed4592, 2026.

Mayank Mittal, Pascal Roth, James Tigue, Antoine Richard, Octi Zhang, Peter Du, Antonio Serrano-Muñoz, Xinjie Yao, René Zurbrügg, Nikita Rudin, Lukasz Wawrzyniak, Milad Rakhsha, Alain Denzler, Eric Heiden, Ales Borovicka, Ossama Ahmed, Iretiayo Akinola, Abrar Anwar, Mark T. Carlson, Ji Yuan Feng, Animesh Garg, Renato Gasoto, Lionel Gulich, Yijie Guo, M. Gussert, Alex

Hansen, Mihir Kulkarni, Chenran Li, Wei Liu, Viktor Makoviychuk, Grzegorz Malczyk, Hammad Mazhar, Masoud Moghani, Adithyavairavan Murali, Michael Noseworthy, Alexander Poddubny, Nathan Ratliff, Welf Rehberg, Clemens Schwarke, Ritvik Singh, James Latham Smith, Bingjie Tang, Ruchik Thaker, Matthew Trepte, Karl Van Wyk, Fangzhou Yu, Alex Millane, Vikram Ramasamy, Remo Steiner, Sangeeta Subramanian, Clemens Volk, CY Chen, Neel Jawale, Ashwin Varghese Kuruttukulam, Michael A. Lin, Ajay Mandlekar, Karsten Patzwaldt, John Welsh, Huihua Zhao, Fatima Anes, Jean-Francois Lafleche, Nicolas Moënne-Loccoz, Soowan Park, Rob Stepinski, Dirk Van Gelder, Chris Amevor, Jan Carius, Jumyung Chang, Anka He Chen, Pablo de Heras Ciechomski, Gilles Daviet, Mohammad Mohajerani, Julia von Muralt, Viktor Reutskyy, Michael Sauter, Simon Schirm, Eric L. Shi, Pierre Terdiman, Kenny Vilella, Tobias Widmer, Gordon Yeoman, Tiffany Chen, Sergey Grizan, Cathy Li, Lotus Li, Connor Smith, Rafael Wiltz, Kostas Alexis, Yan Chang, David Chu, Linxi "Jim" Fan, Farbod Farshidian, Ankur Handa, Spencer Huang, Marco Hutter, Yashraj Narang, Soha Pouya, Shiwei Sheng, Yuke Zhu, Miles Macklin, Adam Moravanszky, Philipp Reist, Yunrong Guo, David Hoeller, and Gavriel State. Isaac lab: A gpu-accelerated simulation framework for multi-modal robot learning. *arXiv preprint arXiv:2511.04831*, 2025.

NVIDIA, Johan Bjorck, Nikita Cherniadev Fernando Castañeda, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. GR00T N1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

Guoping Pan, Qingwei Ben, Zhecheng Yuan, Guangqi Jiang, Yandong Ji, Shoujie Li, Jiangmiao Pang, Houde Liu, and Huazhe Xu. Roboduet: Learning a cooperative policy for whole-body legged loco-manipulation. *IEEE Robotics and Automation Letters*, 10(5):4564–4571, 2025.

Tifanny Portela, Andrei Cramariuc, Mayank Mittal, and Marco Hutter. Whole-body end-effector pose tracking. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11205–11211, 2025.

Jilong Wang, Javokhirbek Rajabov, Chaoyi Xu, Yiming Zheng, and He Wang. Quadwbg: Generalizable quadrupedal whole-body grasping. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11675–11682, 2025.

Kaijun Wang, Liqin Lu, Mingyu Liu, Jianuo Jiang, Zeju Li, Bolin Zhang, Wancai Zheng, Xinyi Yu, Hao Chen, and Chunhua Shen. Odyssey: Open-world quadrupeds exploration and manipulation for long-horizon tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 18602–18610, 2026.

Songlin Wei, Hongyi Jing, Boqian Li, Zhenyu Zhao, Jiageng Mao, Zhenhao Ni, Sicheng He, Jie Liu, Xiawei Liu, Kaidi Kang, et al. Ψ_0: An Open Foundation Model Towards Universal Humanoid Loco-Manipulation. *arXiv preprint arXiv:2603.12263*, 2026.

Haoru Xue, Xiaoyu Huang, Dantong Niu, Qiayuan Liao, Thomas Kragerud, Jan Tommy Gravdahl, Xue Bin Peng, Guanya Shi, Trevor Darrell, Koushil Sreenath, et al. Leverb: Humanoid whole-body control with latent vision-language instruction. *arXiv preprint arXiv:2506.13751*, 2025.

Yutian Zhang, Siyuan Ma, Liwen Yang, Yang Li, Ce Hao, Haozhen Chi, Dong Wei, Qiaojun Yu, and Dibo Hou. Fwbc-vla: Force-aware whole-body compensation for contact-rich loco-manipulation. *arXiv preprint arXiv:2609.03889*, 2026.

A Whole-Body Controller Training Details

A.1 Command Distribution

A.1.1 Stage 1: Reachable Pose Set Generation

We construct an offline end-effector pose set using the Go2–ARX X5 model in Isaac Lab. The robot base is fixed at its nominal height of $0.32\,\mathrm{m}$, with the legs commanded to their default standing configuration and self-collisions enabled. We sample the six arm joints using a scrambled Sobol sequence with seed 0, mapped to the following joint ranges:

$$q^A_{\min} = (-77,\ 120,\ 90,\ -90,\ -50,\ -90)^\circ,$$
$$q^A_{\max} = (77,\ 180,\ 165,\ 90,\ 50,\ 90)^\circ,$$

intersected with the joint limits of the robot model. Sampling is performed in batches of 1,024 configurations. For each candidate, we initialize the joint positions with zero velocity, apply position targets, and advance the simulation for two steps of $0.005\,\mathrm{s}$. We reject candidates whose final net contact-force magnitude exceeds $1\,\mathrm{N}$ on any of the six arm links, and extract the remaining end-effector positions and orientations in the robot base frame.

To reduce overrepresentation of densely sampled regions, we additionally filter candidates by proximity in both position and orientation. After accepting the first 300 candidates that pass the contact check, we reject a new pose if at least two previously accepted poses lie within $0.05\,\mathrm{m}$ and have a relative rotation angle below 22°. This filter is applied sequentially, including to candidates within the same batch. Sampling continues until 77,777 poses have been accepted. We store each accepted pose together with its sampled arm configuration, yielding the reference set from which end-effector commands are subsequently drawn. Fig. 6 visualizes the position component of a subset of these poses.

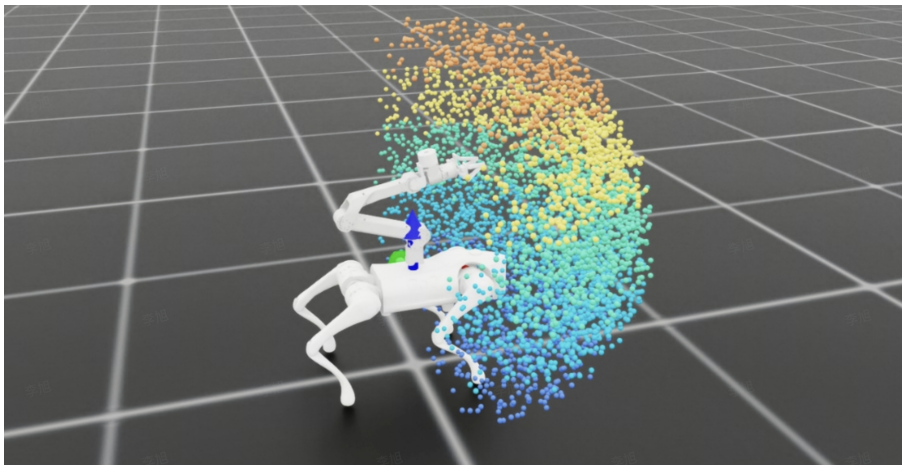


Figure 6: Reachable end-effector pose set for the Go2–ARX X5 with a fixed base. Each point shows the end-effector position of a sampled pose, with colors indicating height from low (blue) to high (orange). A subset of the 77,777 accepted poses is shown for clarity; orientations are stored in the set but omitted from this visualization.

A.1.2 Stage 2: Reference Motion Generation

We freeze the trained whole-body controller π_{wbc} and roll it out in 4,096 parallel Isaac Lab environments on flat terrain for 72,000 control steps per environment, under sampled base-velocity and end-effector pose commands covering locomotion and manipulation. External pushes and end-effector force perturbations are disabled during collection. At each control step ($50\,\mathrm{Hz}$), we record the realized arm joint angles q^A, base orientation with yaw removed Q^{rp}, and the commanded base velocity $(v_x^*, v_y^*, \omega_z^*)$. We also retain the robot states needed to initialize subsequent replay episodes. Only complete episodes that reach the time limit without a failure termination are retained; failed

episodes and unfinished trailing segments are discarded. This yields 284,286 reference episodes, with a maximum length of 1,000 steps (20 s), after discarding 15,632 failure-terminated episodes.

We then smooth the reference signals separately within locomotion and manipulation segments. Manipulation is identified by $\|(v_x^*, v_y^*, \omega_z^*)\|_2 < 0.01$, with segments shorter than 1 s merged into neighboring segments to avoid treating temporary braking as a task switch. During manipulation, each arm joint angle and the base roll and pitch are independently processed with a second-order Butterworth low-pass filter applied forward and backward for zero phase shift; segments too short for the selected filter are left unchanged. The cutoff frequencies are approximately 0.169 and 0.106 Hz for roll and pitch, and $(0.269,\ 0.463,\ 0.417,\ 0.746,\ 0.397,\ 0.403)$ Hz for the six arm joints, respectively. During locomotion, arm references are held at the last value of the preceding manipulation segment, or the initial value if the episode starts with locomotion, while recorded roll and pitch are left unchanged. Velocity commands and initialization states are preserved. The resulting sequences provide the arm and base-posture references for training π_{base}.

A.2 Training Hyperparameters

Both controllers are trained with PPO and Adam. For π_{wbc}, the actor and critic have separate networks, each with a shared backbone of widths $(512, 256)$ followed by two 128-unit branches. The actor branches output 12 leg and 6 arm action means, while the critic branches output the corresponding two value estimates. For π_{base}, the actor and critic are separate MLPs with hidden widths $(512, 256, 128)$, producing 12 leg action means and one value estimate, respectively. All hidden layers use ELU activations, and both policies receive a five-step observation history. Gaussian policies use learnable, state-independent log standard deviations initialized to zero and clipped to $[-20, 2]$. Table 3 summarizes the optimization settings; the learning rate is linearly annealed by the training scripts.

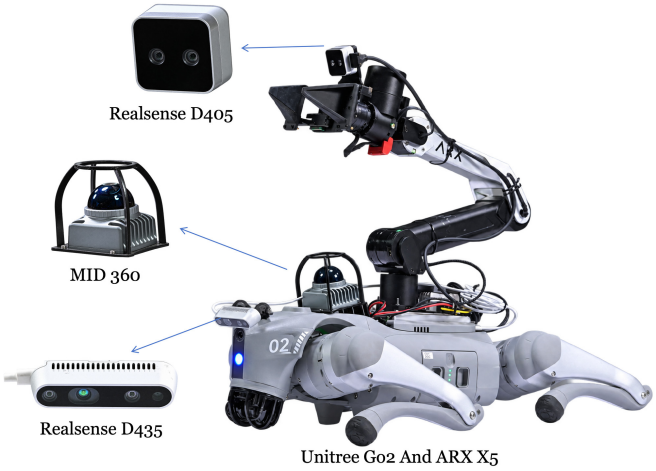
Table 3: Network and PPO settings for the two controllers. Action scales convert policy outputs into joint-position offsets in radians.

Hyperparameter	π_{wbc}	π_{base}
Actor outputs / critic outputs	$18/2$	$12/1$
Observation history	5 steps	5 steps
Leg / arm action scale	$0.25/0.20$	0.25/direct arm targets
Parallel environments	4,096	4,096
Training steps per environment	72,000	72,000
Rollout length per update	24	24
PPO updates	3,000	3,000
Epochs per update	5	5
Mini-batches per epoch	4	4
Samples per mini-batch	24,576	24,576
Learning rate (initial $\rightarrow$ final)	$3\times10^{-4} \rightarrow 10^{-6}$	$3\times10^{-4} \rightarrow 10^{-6}$
Discount factor γ	0.99	0.99
GAE parameter λ	0.95	0.95
Policy ratio clipping	0.2	0.2
Value clipping	0.2	0.2
Entropy loss coefficient	0.01	0.01
Value loss coefficient	2.0	1.0
Maximum gradient norm	1.0	1.0
Reward scaling factor	0.1	0.1
Time-limit bootstrapping	Enabled	Enabled
Random seed	42	42

Table 4 lists the reward weights after all environment-configuration overrides. The two controllers share most gait and regularization terms; π_{wbc} additionally tracks the end-effector pose and regularizes arm actions, whereas π_{base} tracks the recorded base orientation while the arm follows replayed joint targets. Task-dependent terms retain their command-based activation conditions. Both controllers use a 3 Hz trot reference with a stance fraction of 0.5 and a target swing-foot clearance of 0.07 m.

B TELEOPERATION DATA COLLECTION

Robot and sensors. Our platform combines a Unitree Go2 quadruped with a 6-DoF ARX X5 arm and a gripper (Fig. 7). A front-mounted RealSense D435 and a wrist-mounted RealSense D405 provide the two RGB views used for demonstration collection. A MID-360 LiDAR supplies measurements for LiDAR–inertial odometry. An onboard NVIDIA Orin runs the whole-body policy and ROS 2 control stack. Policy inference runs at $50\,\mathrm{Hz}$, while the arm and leg SDK control loops run at $500\,\mathrm{Hz}$.



(a) Robot and sensor placement



(b) PICO VR interface

Figure 7: Hardware for teleoperation data collection. (a) The Go2–ARX X5 platform with front and wrist cameras and a MID-360 LiDAR; insets show the corresponding sensors. (b) The PICO headset and handheld controllers used to command locomotion, end-effector motion, and gripper opening.

VR command interface. We read PICO controller poses and inputs through XRoboToolkit and publish commands over ROS 2 at $30\,\mathrm{Hz}$. The left joystick controls planar base velocity, and the horizontal axis of the right joystick controls yaw rate. The right trigger sets the gripper opening from 0 to $0.08\,\mathrm{m}$. The right grip button acts as a clutch: pressing it establishes reference poses for the hand controller and the robot end effector. While held, the controller's displacement and relative rotation, expressed in its reference frame after conversion from PICO to ROS coordinates, are composed with the end-effector reference to form the target pose, with unit translation scale. Releasing the grip stops hand-motion updates and re-anchors the end-effector target, allowing the operator to reposition the controller. The target frame follows base translation in the horizontal plane and base yaw, with its vertical origin fixed at clutch activation. These pose targets and base-velocity commands are executed by π_{wbc}.

Demonstration recording. We record robot state at $50\,\mathrm{Hz}$, including all 18 joint positions, gripper position, arm-mount orientation, realized and commanded end-effector poses, and base-velocity commands. The two cameras record RGB independently at 30 frames per second, at 640×480 resolution for the wrist view and 424×240 for the front view. Camera roles are bound to device serial numbers. State samples and image arrivals are timestamped using the ROS clock and saved per episode, enabling subsequent temporal alignment of the asynchronous streams.

C VLA TRAINING DETAILS

C.1 DATA PROCESSING

We convert raw teleoperation logs into LeRobot datasets (**?**) using a shared processing pipeline. All sensor and control streams are synchronized at $50\,\mathrm{Hz}$. Each sample contains front-view and wrist-view RGB observations, robot proprioception, a language instruction, and the operator command.

Table 4: Configured reward weights before the global reward scaling in Table 3. A dash indicates an absent term. The contact-force and contact-velocity shaping terms return non-positive values, so their positive weights still penalize gait mismatch.

Reward term	π_{wbc}	π_{base}
Planar velocity tracking	2.5	1.5
Yaw-rate tracking	1.875	0.75
End-effector pose tracking	2.0	—
Base roll/pitch tracking	—	1.5
Feet air time	2.5	2.5
Gait contact-force shaping	2.0	2.0
Gait contact-velocity shaping	2.0	2.0
Swing-foot clearance error	-4.0	-4.0
Flat-base orientation penalty	-2.5	-2.5
Leg posture deviation during locomotion	-0.25	-0.25
Foot-contact reward during manipulation	1.0	1.0
Standing leg posture deviation	-0.2	-0.2
Base tilt tolerance during manipulation	0.75	0.75
Base height error	-5.0	-5.0
Vertical base velocity	-2.0	-2.0
Base roll/pitch angular velocity	-0.05	-0.05
Leg action rate	-0.01	-0.01
Arm action rate	-0.08	—
Leg joint torque	-2×10^{-4}	-2×10^{-4}
Rear-hip joint torque (additional)	-3×10^{-4}	-3×10^{-4}
Arm joint torque	-5×10^{-4}	—
Leg joint acceleration	-2.5×10^{-7}	-2.5×10^{-7}
Arm joint acceleration	-10^{-5}	—
Undesired body contacts	-10.0	-10.0
Foot sliding	-0.3	-0.3
Leg joint position limits	-5.0	-5.0
Arm joint position limits	-5.0	—

Before conversion, we remove invalid control cycles, incomplete episodes, and frames with missing camera observations. The resulting datasets contain one action chunk per observation and are used directly by both $\pi_{0.5}$ and GR00T N1.7.

Commanded end-effector actions. For the EE-command interface, we use the commanded VR pose rather than the measured robot pose as the action target. Let $\mathbf{T}^{W}_{E,0}$ denote the realized end-effector pose at the beginning of a chunk and let $\widehat{\mathbf{T}}^{W}_{E,k}$ denote the commanded pose at future step k. We express every target in the coordinate frame anchored at $\mathbf{T}^{W}_{E,0}$:

$$\mathbf{T}^{E_0}_{E,k} = \left(\mathbf{T}^{W}_{E,0}\right)^{-1} \widehat{\mathbf{T}}^{W}_{E,k}, \quad k \in \{0, \ldots, H-1\}. \tag{7}$$

This transformation removes global position and heading while preserving the commanded motion relative to the robot state at the beginning of the chunk. The relative rotation is converted from a scalar-first quaternion to an axis-angle vector. Each action step contains the three-dimensional relative position, three-dimensional rotation vector, gripper command, and future planar base command, yielding a 10-dimensional action.

Structured joint actions. For the structured joint interface, each action step contains the arm-mount roll and pitch targets; six arm-joint targets; the gripper command; and the planar base commands (v_x, v_y, ω_z). This produces a 12-dimensional action that can be routed directly to the structured controller.

Predicting base velocities. When the Unitree low-level control service is active, the high-level sport service is disabled and does not provide measured base velocities. We therefore formulate

base-velocity prediction as sequence forecasting: conditioned on the previous velocity command and current visual observations, the VLA predicts the next velocity command. Empirically, we find that this formulation converges and reproduces the stopping behavior of the human teleoperator.

Proprioceptive observations. The proprioceptive state is 13-dimensional: arm-mount roll and pitch, episode-relative base yaw; six measured arm-joint positions; gripper position; and the previous planar base command. We use the previous base command as a velocity surrogate as mentioned before.

Action chunks. Future actions are precomputed during dataset conversion and stored as fixed-length arrays. $\pi_{0.5}$ uses a horizon of 50 steps, corresponding to 1 s at 50 Hz, whereas GR00T N1.7 uses its native horizon of 40 steps.

C.2 VLA Post-Training

We post-train $\pi_{0.5}$ (Intelligence et al., 2025) and GR00T N1.7 (NVIDIA et al., 2025) using their native training stacks, augmented with loaders for the precomputed action chunks and optional train-time real-time chunking (RTC) (Black et al., 2025). Unless stated otherwise, all models receive the two RGB observations, language instruction, proprioceptive state, and an action chunk in the selected representation.

Train-time RTC. RTC simulates the action prefix that may already have been executed while the next policy query is processed. For each training sample, we draw a prefix length

$$d \sim \mathcal{U}\left(d_{\min}, d_{\max}\right), \tag{8}$$

replace the first d positions of the noisy action input with their clean targets, and mask their contribution to the denoising loss. The remaining positions follow the backbone's original noise-interpolation procedure. We use uniformly distributed delay in the reported experiments.

$\pi_{0.5}$. We initialize $\pi_{0.5}$ from the public base checkpoint or a specified intermediate checkpoint. Unless otherwise stated, we fine-tune the complete model for 3,000 optimization steps with AdamW, a learning rate of 10^{-5}, bfloat16 computation, and a batch size of 8 per GPU. For multi-task training, task i is sampled with probability

$$p_i = \frac{n_i^{0.43}}{\sum_j n_j^{0.43}}, \tag{9}$$

where n_i is the number of frames available for task i. This sublinear weighting reduces domination by the largest dataset while retaining a preference for tasks with more demonstrations.

GR00T N1.7. We initialize GR00T from its public N1.7 checkpoint and use the same 3,000-step schedule, per-device batch size as with $\pi_{0.5}$. Training uses bfloat16 computation with FP32 master parameters. We freeze the language and vision backbones and optimize the multimodal projector, diffusion transformer, and VLLN modules.

Deployment. $\pi_{0.5}$ and GR00T receive the same camera mapping, language instruction, and proprioceptive schema used during training. When prefix-conditioned execution is enabled, the deployment client also sends the executed-prefix length and previously issued action chunk.

D Real-World Evaluation

D.1 Tasks And Evaluation Protocol

Task 1: Mobile Pick-and-Place

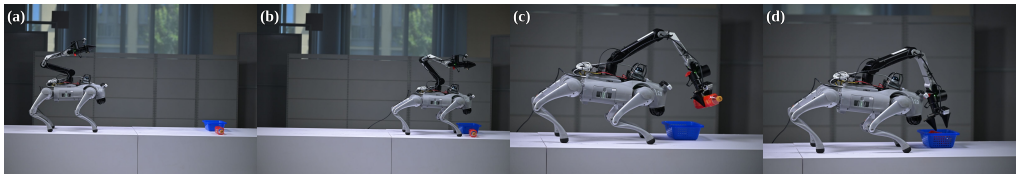
Language Prompt: Walk up to the bottle and basket, pick up the bottle from the ground, and put it into the basket.

Task Description: The robot needs to locomote to an appropriate position near the bottle and basket, accurately locate and grasp the bottle placed on the ground, then lift and carefully place it into the basket. The robot must maintain whole-body balance throughout the manipulation process while coordinating its mobile base and upper-body joints to complete the task.

Evaluation Rubric: 3 stages in total:

- The robot successfully walks up to the correct position near the bottle and basket (b).
- The robot picks up the correct object (the bottle) from the ground (c).
- The bottle is successfully placed into the basket (d).

Task Category: loco-manipulation.



Task 2: Mobile Box Pushing

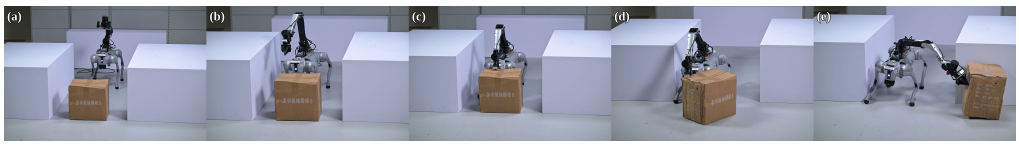
Language Prompt: Clear the corridor by pushing the box behind the obstacle.

Task Description: The robot approaches the box and orients its body to establish stable contact. It then applies continuous pushing force to translate the box behind the obstacle while regulating contact dynamics. To execute the push, the robot leverages lower-body locomotion and upper-body force generation to maintain postural stability and guide the box along the desired trajectory.

Evaluation Rubric: 4 stages in total:

- The robot successfully approaches and stops at a suitable operating position near the box (b).
- The robot positions its end-effector at an appropriate height for pushing (c).
- The robot successfully pushes the box away while maintaining stable contact (d).
- The box is successfully pushed sideways behind the obstacle, clearing the corridor (e).

Task Category: contact-rich manipulation.



Task 3 : Drawer Manipulation

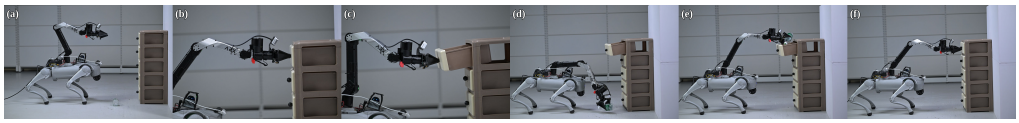
Language Prompt: Open the top drawer, pick up the bottle from the ground, put it into the drawer, then close the drawer.

Task Description: The robot grasps the handle, and pulls the top drawer open along its sliding rails. It then fetches the bottle from the floor, places it inside the opened drawer, and slides the drawer shut. To complete this sequence, the robot must coordinate its mobile base, arm, and end-effector to operate within the confined workspace imposed by the drawer mechanism.

Evaluation Rubric: 5 stages in total:

- The robot successfully grasps the circular handle of the drawer (b).
- The robot successfully opens the top drawer (c).
- The robot successfully grasps the bottle from the ground (d).
- The bottle is successfully placed into the drawer (e).
- The drawer is successfully closed (f).

Task Category: constrained manipulation.



Task 4 : Mobile Object Transport

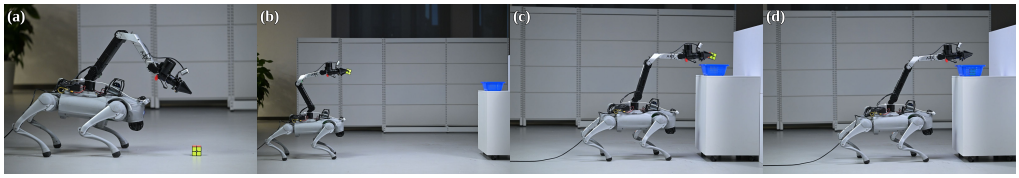
Language Prompt: Pick up the cube from the ground, carry it forward, and put it into the basket.

Task Description: The robot grasps the cube from the ground and lifts it. It then locomotes forward while maintaining a stable grasp to prevent the cube from slipping or falling, and finally positions the cube over the basket and releases it inside. During transport, the robot must dynamically balance its locomotion and manipulation to maintain grasp stability against inertial disturbances.

Evaluation Rubric: 3 stages in total:

- The robot successfully grasps the cube from the ground (b).
- The robot successfully carries the cube forward without dropping it (c).
- The cube is successfully placed into the basket (d).

Task Category: constrained loco-manipulation.



D.2 Task Completion Process

We evaluate each trial using a cumulative task-progress score. Each task is divided into ordered milestones, and each attained milestone contributes one point, counted at most once per trial. A trial therefore receives partial credit even if the complete task is not finished. The maximum scores are 3 for mobile pick-and-place, 4 for box pushing, 5 for drawer manipulation, and 4 for cube transport; a trial that reaches no milestone receives 0 points. Table 5 specifies the criteria.

For a task with K milestones, the raw score and normalized progress of trial i are

$$S_i = \sum_{j=1}^{K} m_{i,j}, \qquad P_i = \frac{S_i}{K},$$

where $m_{i,j} = 1$ if milestone j is attained and 0 otherwise. Average task progress over N trials is $\frac{1}{N}\sum_{i=1}^{N} P_i$, reported as a percentage. This metric retains differences in partial completion that a binary trial-level outcome would discard.

Table 5: Task-progress scoring criteria. Each numbered milestone contributes one point.

Task	Max. score	Milestones
Mobile pick-and-place	3	1. Stop at a suitable position for manipulating the bottle. 2. Successfully grasp the bottle with the gripper. 3. Place the bottle into the basket.
Cube transport	4	1. Contact the cube during a grasp attempt. 2. Securely grasp the cube. 3. Come to a stable stop at the placement location after transport. 4. Place the cube into the basket.
Box pushing	4	1. Stop at a suitable position for manipulating the box. 2. Position the gripper at an appropriate height for pushing. 3. Correctly push the box away. 4. Correctly push the box laterally to clear the passage.
Drawer manipulation	5	1. Successfully grasp the round drawer handle. 2. Open the drawer. 3. Successfully grasp the bottle. 4. Place the bottle inside the drawer. 5. Close the drawer.

For cube transport, contact and secure grasping are cumulative milestones. An imprecise grasp attempt that contacts the cube but causes it to slip or be pushed away as the gripper closes earns 1 point. A secure grasp also satisfies the contact milestone and thus earns 2 points in total. Reaching a stable stop after transport increases the score to 3, and placement in the basket yields the maximum of 4.

Figs. 12–15 provide visual examples of the task stages. Frames are ordered from left to right along the top row, followed by the bottom row. Red boxes mark selected evaluation checkpoints; the complete scoring criteria are given in Table 5.

Ⓐ Loco-manipulation

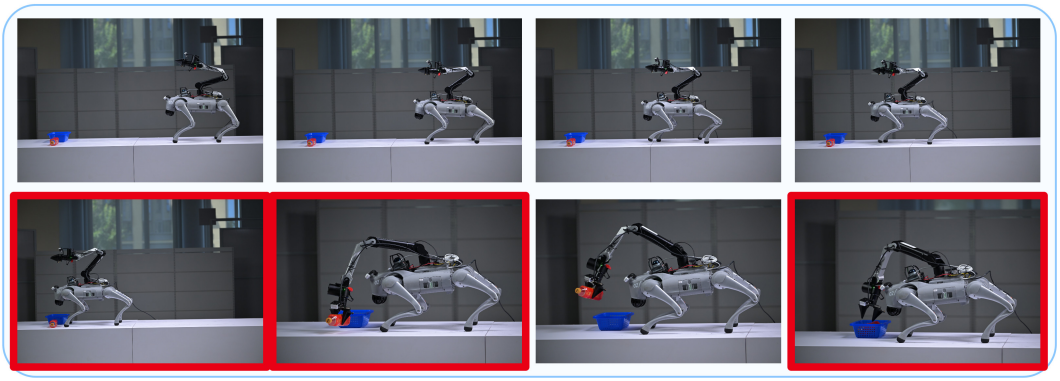


Figure 12: Evaluation checkpoints for mobile pick-and-place: stopping at a suitable manipulation position, grasping the bottle, and placing it in the basket (3 points in total).

B Constrained Manipulation And Loco-manipulation

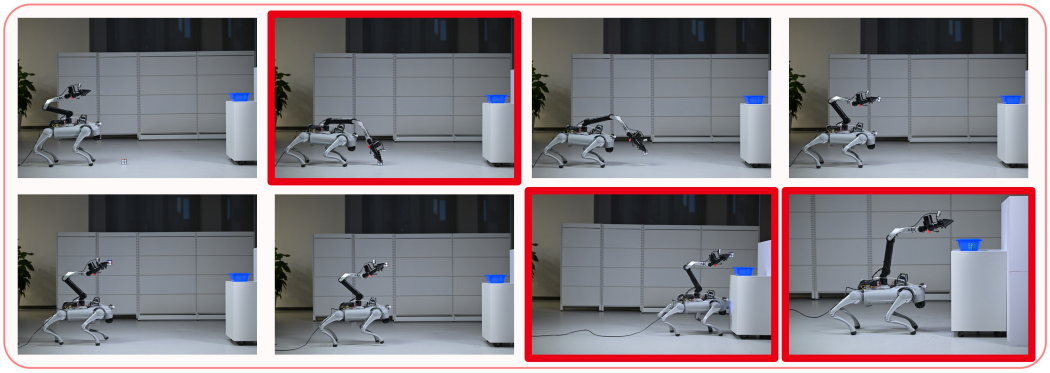


Figure 13: Evaluation checkpoints for cube transport: contact, secure grasping, stable stopping after transport, and placement in the basket (3 points in total). Secure grasping earns cumulative credit for the first two milestones.

C Contact-rich Manipulation

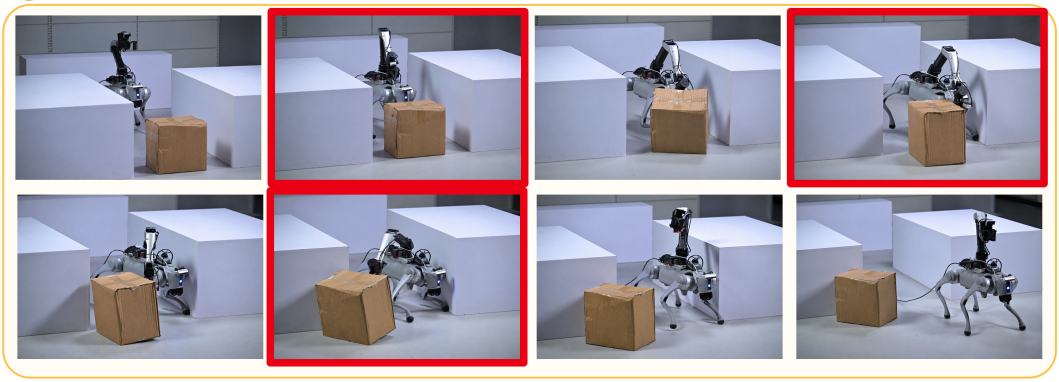


Figure 14: Evaluation checkpoints for box pushing: stopping at a suitable manipulation position, establishing the appropriate gripper height, pushing the box away, and pushing it laterally to clear the passage (3 points in total).

D Constrained Manipulation

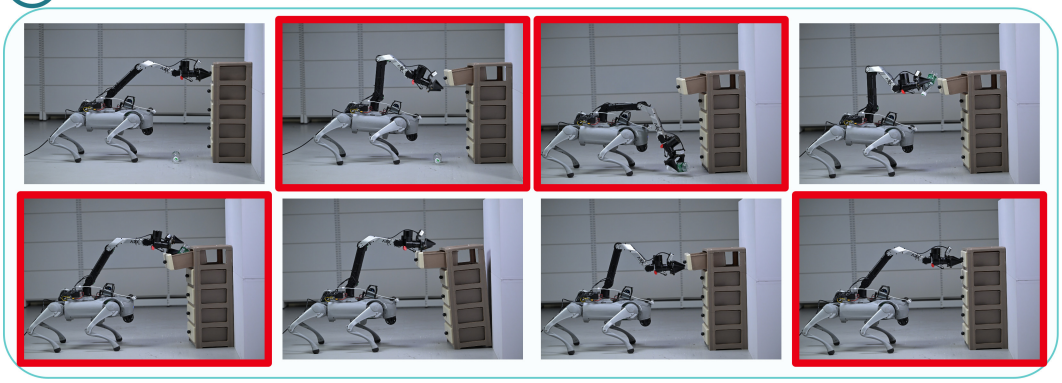


Figure 15: Evaluation checkpoints for drawer manipulation: grasping the round handle, opening the drawer, grasping the bottle, placing it inside, and closing the drawer (4 points in total).

D.3 Failure Cases Analysis

Fig. 16 illustrates three recurring types of failure. First, inaccurate grasp alignment can prevent object acquisition or lead to slipping, especially for small objects and drawer handles. Second, contact-constrained interactions require appropriate motion directions and placement configurations: establishing contact alone does not ensure that a pushing, pulling, or closing motion can be completed. Third, the policy can select an action from the wrong stage of a multi-step task. In the drawer task, it sometimes attempts to pick up the bottle before opening the drawer, suggesting a limitation in tracking task progress without explicit episodic memory. These cases highlight that precise low-level tracking must be complemented by accurate target selection, contact-aware motion prediction, and reliable task sequencing.

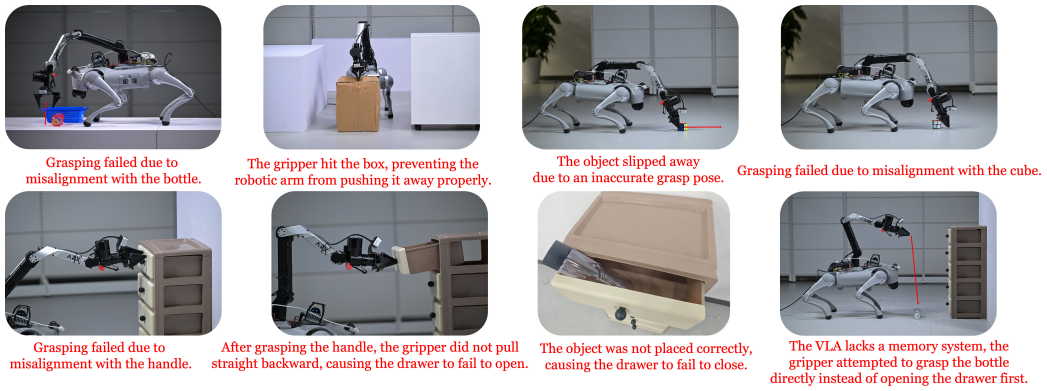


Figure 16: Representative failure cases in real-world experiments. Explanations are provided within the figure.

E Limitations

Our evaluation is limited to a single quadruped–arm embodiment and a controlled set of indoor tasks. Although the tasks cover mobile, constrained, contact-rich, and small-object manipulation, they do not fully test uneven terrain, dynamic disturbances, deformable objects, or extended long-horizon operation. The structured action interface is also embodiment-specific: transferring the system to a different arm placement or robot morphology would require collecting new coordination references and retraining the low-level controller.

The proposed decomposition prioritizes execution precision but may restrict behaviors that require tightly coupled, high-frequency arm-leg reactions. Moreover, because measured base velocity is unavailable under the Unitree low-level service, the VLA conditions on the previous velocity command as a surrogate and must infer motion transitions from visual observations. Failures caused by high-level memory, perception, and object placement remain even when controller tracking is accurate. Extending the system with direct state estimation, broader task and environment diversity, and closed-loop recovery mechanisms constitutes important future work.