

第一页为封面页

参赛学生姓名: Ava Ruike Zhou

中学: Western Academy of Beijing

省份: 北京市

国家/地区: 中国

指导老师姓名: 徐葳

指导老师单位: 清华大学

论文题目: Align Before Acting: Task-Agnostic Post-Training
for Vision-Language-Action Models

Align Before Acting: Task-Agnostic Post-Training for Vision-Language-Action Models

Ava Ruike Zhou

Abstract

Adapting a pretrained vision-language-action (VLA) model to a new robot requires learning both how the robot moves and what a downstream task requires. When these objectives are learned only from task demonstrations, scarce data must repeatedly teach embodiment-specific motion conventions alongside task behavior. We introduce EMBARK (Embodiment Motion Bootstrapping for Adaptation to Robot Kinematics), a two-stage post-training framework that separates these objectives. Stage A uses task-agnostic motion procedurally generated in an embodiment-matched simulator built from the robot’s hardware description. Once the simulator is configured, these data can be generated at low marginal human collection cost, without per-trajectory teleoperation or downstream task annotation, to align the target robot’s state/action interface, kinematic constraints, and reachable workspace. Stage B then adapts the aligned checkpoint to downstream tasks using few-shot demonstrations. The resulting Stage A checkpoint can be reused across tasks, turning embodiment alignment into a shared investment rather than a repeated per-task cost.

We evaluate EMBARK with X-VLA and GR00T N1.7 on a physical SO-101 arm and a Franka Panda in LIBERO. On held-out SO-101 trajectories, task-agnostic warmup reduces joint-angle MAE by 15.7–33.1% for X-VLA and 6.9–22.5% for GR00T N1.7 relative to matched Real-Only adaptation. In closed-loop LIBERO-90 evaluation with two demonstrations per task, warmup improves test success by 23.1 percentage points early in adaptation and retains a 4.7-point advantage at the final evaluation. In a small paired physical LEGO-sorting study, the success point estimate increases from 60% to 70% at the lower demonstration budget and from 70% to 75% at the higher budget.

Although the physical study is exploratory, these results indicate that embodiment-matched simulation provides a reusable optimization prior that improves motion prediction, accelerates few-shot task adaptation, and reduces the repeated burden of learning embodiment mechanics from task demonstrations. This separation opens a path toward more efficient multi-task post-training and broader reuse of embodiment alignment across future robot-learning systems.

Keywords: vision-language-action models; embodiment alignment; robot adaptation; sim-to-real transfer; few-shot learning

Contents

1	Introduction	3
2	Related Work	4
3	A Three-Component Framework for VLA Adaptation	6
3.1	From Observations to Actions	6
3.2	Our Proposed Functional Framework	6
3.3	Implications for Task-Agnostic Adaptation	7
4	EMBARK: Two-Stage Embodiment Alignment and Few-Shot Task Adaptation	7
4.1	Decoupling Embodiment Alignment from Task Learning	7
4.2	Simulation Data Generation for SO-101 and Franka Panda	9
4.3	Model-Specific Post-Training Configurations	11
4.4	Toward Generalist VLA Deployment	12
5	Experiments	12
5.1	Experimental Setup	12
5.1.1	Physical SO-101 tasks	13
5.1.2	LIBERO tasks	13
5.2	Adaptation on a Physical Robot: SO-101	13
5.2.1	X-VLA	14
5.2.2	GR00T N1.7	16
5.2.3	Closed-Loop Evaluation	17
5.3	Adaptation in LIBERO: Franka Panda	18
5.3.1	Task-agnostic motion warmup on LIBERO-90	19
5.3.2	Comparison with LIBERO-Spatial	20
5.4	Ablation Studies	21
5.4.1	Two-Stage Training versus Direct Co-Training	21
5.4.2	Trainable Components during Warmup	21
6	Conclusion	22
A	Architectural Analysis of VLAs	23
B	Stage A Instruction Pools	26
	References	28

1 Introduction

Robotic arms are increasingly used for tasks such as picking up objects, sorting materials, and helping with manufacturing. However, an LLM-based system that understands an instruction may still be unable to control a particular robotic arm reliably. Vision–language–action (VLA) models seek to turn broad visual and linguistic knowledge into general-purpose robot policies. Pretraining on heterogeneous robot trajectories, web data, and human videos has produced models that can interpret diverse scenes and instructions and can be adapted to new manipulation tasks [1], [2], [3], [4], [5], [6]. However, semantic generality does not imply control readiness. A pretrained VLA may recognize the relevant objects and understand the requested behavior while remaining poorly calibrated to a target robot’s camera geometry, proprioceptive state, kinematics, action conventions, workspace, and contact behavior. Bridging this *embodiment gap* is therefore a central post-training problem for reliable deployment.

Conventional task-specific fine-tuning addresses this gap with demonstrations of the target task. Each demonstration must then teach both *what* the robot should do and *how* the target embodiment senses and moves. For example, a demonstration of picking up a cup teaches the model not only where the cup is and how to complete the task but also how to use camera observations to guide the particular robot’s joint movements and gripper approach. This coupling is inefficient when one robot must acquire many skills: kinematics, sensor–robot geometry, state and action semantics, and low-level motion regularities are largely shared across tasks, yet they are repeatedly learned from scarce task demonstrations. This observation motivates a different question: can embodiment alignment be learned once from inexpensive, task-agnostic simulation data and then reused for few-shot adaptation to multiple downstream tasks?

We propose EMBARK (Embodiment Motion Bootstrapping for Adaptation to Robot Kinematics), a two-stage task-agnostic post-training framework. Our approach follows a functional view of a VLA system as comprising a VLM backbone for semantic understanding, an embodiment interface for state and action grounding, and a generative action module for continuous control. In Stage A, we generate task-agnostic motion in a simulator built for the target robot and use it to align the embodiment interface. The simulator uses the robot’s hardware description and a calibrated observation/action interface. Once configured, it produces additional motion trajectories without per-trajectory teleoperation or downstream task annotation, providing alignment data at low marginal human collection cost. The resulting checkpoint captures the robot’s state/action contract, kinematic constraints, and reachable workspace through the embodiment interface, without using downstream task labels. In Stage B, we use few-shot target-task adaptation to add task semantics, target-domain appearance, and interaction details. A single Stage A checkpoint can initialize models for many Stage B tasks, turning embodiment alignment from a recurring per-task cost into an amortized, once-per-embodiment investment. This separation reflects what stays the same when tasks change: the robot’s joints, camera geometry, and reachable workspace. Stage A provides a reusable starting point for how the robot senses and moves; Stage B teaches the objects, goals, and interaction sequences of a particular task from a limited number of demonstrations.

We evaluate task-agnostic motion warmup using two VLA models, X-VLA and GR00T N1.7, and two robot platforms, the physical SO-101 arm and the Franka Panda in LIBERO. Across multiple tasks and demonstration budgets, we measure action-prediction accuracy and closed-loop task success rates. These complementary evaluations test both how accurately the models predict

robot movements and how reliably they complete tasks. Each robot is aligned independently; we do not test zero-shot transfer of a Stage A checkpoint between robots.

The results show consistent benefits across the evaluated settings. On SO-101 data, warmup reduces overall joint-angle mean absolute error (MAE) by 15.7–33.1% for X-VLA and 6.9–22.5% for GR00T N1.7 compared with matched Real-Only adaptation, while also improving fingertip-position and end-effector orientation accuracy. On the Franka Panda, warmup improves task success on both LIBERO-90 and LIBERO-Spatial. With two demonstrations per task on LIBERO-90, it increases the test success rate by 23.1 percentage points early in training and retains a 4.7-percentage-point advantage at the end of training. On LIBERO-Spatial, the final gain with five demonstrations per task is smaller, at 1.0 percentage point. These results suggest that warmup provides a useful starting point for adaptation, particularly early in training and when learning diverse tasks from few demonstrations, but does not replace task-specific adaptation.

The results across two models and two robot embodiments provide evidence that the approach is not limited to one model or robot. They support task-agnostic embodiment alignment as a reusable component of VLA post-training rather than an isolated data augmentation strategy for one deployment setting.

Our contributions are:

- We propose a functional view of VLA systems with three components: a VLM backbone for semantic understanding, an embodiment interface for grounding robot state and actions, and a generative action module for continuous control. This view helps identify where task-agnostic post-training can be applied and reused.
- We formulate VLA adaptation as a reusable two-stage procedure: task-agnostic embodiment alignment followed by few-shot target-task adaptation. This procedure separates knowledge about how a robot senses and moves from knowledge about the task it must perform.
- We develop a general data acquisition strategy for embodiment alignment: a robot’s hardware description is used to configure an embodiment-matched simulator that generates task-agnostic motion under policy-compatible interfaces, without per-trajectory teleoperation or task labels. The resulting data and aligned checkpoint can be reused across downstream tasks. We implement and validate this strategy on two distinct robot platforms using `sim-lerobot` for SO-101 and `sim-libero` for the Franka Panda, providing evidence that the method is applicable across embodiments.
- We demonstrate consistent improvements across X-VLA and GR00T N1.7, the SO-101 and Franka Panda, and complementary open-loop and closed-loop evaluations. These results provide initial evidence that task-agnostic motion warmup can generalize beyond a single model or robot.

2 Related Work

Separating motor knowledge from task learning. TAP shares our motivation to learn how to move before learning what task to perform, but develops this idea as a VLA pretraining strategy: it aims to acquire transferable motor knowledge from robot interactions without task-language labels, reducing reliance on language-annotated expert demonstrations [7]. It first predicts actions

from current and future observations through inverse-dynamics pretraining, then fine-tunes the same model on language-conditioned tasks. Our work addresses a subsequent adaptation problem: given an already pretrained VLA, how can it be prepared for a particular robot before target-task fine-tuning? We use motion generated in an embodiment-matched simulator to post-train the policy with current images, proprioception, and low-level motion instructions. Both stages retain the same policy input and output structure, while the instructions change from commanded motions to downstream task goals. The resulting checkpoint provides a shared initialization for task adaptation on that robot.

Related methods acquire and use motor knowledge in different ways. Learning Action Priors pretrains a standalone state-action module before visual-language alignment [8], whereas we adapt an existing VLA directly with multimodal motion data. JoyAI-RA 0.5 supplies a separately pretrained world model’s frozen dynamics features to action learning [9]; our warmup requires no additional world-model branch. ICWM adapts through task-agnostic probing context at inference time without parameter updates [10], whereas we store the adaptation in model parameters through offline training before task fine-tuning.

Simulation and synthetic robot data. Simulation and generative models provide scalable alternatives to manual robot data collection. MimicGen transforms a small number of human demonstrations into large task-specific datasets across new scene configurations [11]; DreamGen generates synthetic videos of target behaviors with a video world model and recovers pseudo-action trajectories [12]; and RialTo builds task-specific digital twins for real-to-sim-to-real policy learning [13]. Sim-and-real co-training jointly mixes simulated and physical demonstrations and characterizes the data composition and optimization choices required for effective transfer [14]. These approaches use simulation mainly to supplement real demonstrations with task or scene data, so task-level sim-to-real fidelity is important. EMBARK gives simulation a different role: Stage A uses the target robot’s hardware description to teach its state/action conventions and kinematic constraints, rather than to reproduce a downstream task or its appearance. It therefore does not require task-level or photorealistic fidelity, but it does require consistent robot, interface, and motion models. Stage B then learns task semantics and target-domain interactions from real task demonstrations.

Generalist VLAs and cross-embodiment learning. Vision-language-action models transfer semantic and visual knowledge from vision-language pretraining into robot control, with representative systems including RT-2, Octo, OpenVLA, π_0 , $\pi_{0.5}$, GR00T, and X-VLA [1], [2], [3], [6], [15], [16], [17]. Large heterogeneous corpora such as Open X-Embodiment enable policies to share data across tasks and robot platforms [18]. RoboCat adapts a multi-embodiment generalist agent to new robots and tasks, while CrossFormer and HPT learn shared policies or representations from heterogeneous embodiments and sensor-action interfaces [19], [20], [21]. SpatialVLA improves cross-robot transfer through spatial representations and adaptive action grids learned from language-conditioned robot demonstrations [22]. These works amortize learning through broad multi-robot pretraining. We instead study a complementary downstream problem: specializing an already pretrained VLA to a target robot using inexpensive, task-agnostic motion before scarce target-task demonstrations are introduced.

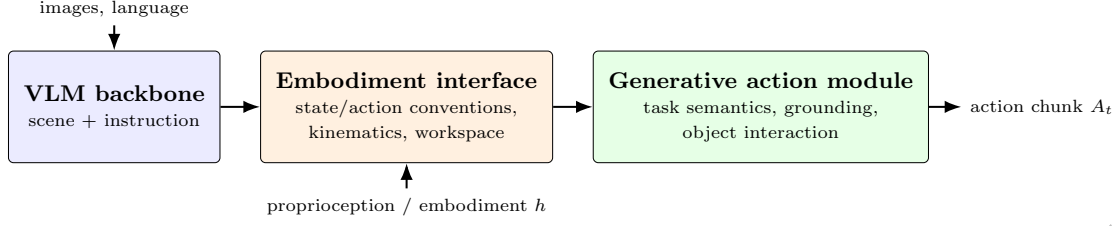


Figure 1: Our proposed three-component functional framework for VLA adaptation. A VLM backbone interprets the scene and the instruction; an embodiment interface aligns embodiment-specific physics and action conventions; a diffusion or flow-matching transformer generates continuous motor chunks.

3 A Three-Component Framework for VLA Adaptation

We propose a functional framework that distinguishes three components of VLA adaptation: semantic understanding, embodiment grounding, and continuous action generation. By identifying reusable robot knowledge and where to apply task-agnostic post-training, this framework provides the conceptual foundation for our two-stage adaptation method.

3.1 From Observations to Actions

A vision-language-action (VLA) model is a single policy that maps onboard sensory observations and a natural-language instruction to robot motor commands [1].

$$\text{Robot Action} = \text{Policy}_{\theta}(\text{Images, Language, Robot State}).$$

Formally, at time t the policy consumes a multimodal observation $\mathbf{o}_t$, including one or more camera views $\mathbf{I}_t^{1:K}$, the current proprioceptive state $\mathbf{q}_t$, and a language goal ℓ . It emits an *action chunk* $\mathbf{A}_{t:t+H-1} = [\mathbf{a}_t, \mathbf{a}_{t+1}, \dots, \mathbf{a}_{t+H-1}]$ over a prediction horizon H [4], [5], [6]. A VLA policy (π_{θ}) can be written as

$$\mathbf{A}_{t:t+H-1} = \pi_{\theta}(\mathbf{I}_t^{1:K}, \ell, \mathbf{q}_t). \quad (1)$$

VLA pretraining combines broad visual and language knowledge with robot demonstrations. However, this does not ensure reliable control of a particular robot. We distinguish three functional components to identify where robot-specific adaptation can occur.

3.2 Our Proposed Functional Framework

Figure 1 illustrates our proposed decomposition. We define the three components by their functional roles rather than by a fixed arrangement of network layers, allowing different VLA architectures to be analyzed within the same framework. The following equations formalize these roles.

- **Semantic understanding identifies what is in the scene and what the instruction requests.** The VLM backbone maps multi-view images and language into a semantic representation,

$$\mathbf{z}_t^{\text{sem}} = f_{\text{VLM}}(\mathbf{I}_t^{1:K}, \ell). \quad (2)$$

This describes the requested behavior, but not how the particular robot should execute it.

- **Embodiment grounding encodes how the target robot senses and moves.** The embodiment information $\mathbf{e}$ describes the robot’s joint and action spaces, kinematic structure, sensor geometry, control mode, and workspace. It may be provided explicitly or encoded implicitly through the interface design and learned parameters. The embodiment interface combines this information with the current proprioceptive state:

$$\mathbf{z}_t^{\text{emb}} = P_\phi(\mathbf{q}_t, \mathbf{e}). \quad (3)$$

The resulting representation guides action generation toward motions appropriate for the robot, without guaranteeing physical feasibility.

- **Continuous action generation turns semantic and embodiment representations into executable motion.** A generative action module predicts a continuous action chunk,

$$\hat{\mathbf{A}}_{t:t+H-1} = G_\theta(\mathbf{z}_t^{\text{sem}}, \mathbf{z}_t^{\text{emb}}). \quad (4)$$

The resulting trajectory specifies *how* to realize the instruction over the prediction horizon.

3.3 Implications for Task-Agnostic Adaptation

The key implication of our framework is that embodiment-dependent knowledge need not be learned only from task-specific demonstrations. For a fixed robot configuration, joint structure, action conventions, and many state–action relationships are shared across tasks, even when task goals and interaction sequences differ. This motivates learning an embodiment-aligned initialization from task-agnostic motion in Stage A, before adapting it to downstream tasks in Stage B. Reuse does not require the embodiment interface to remain fixed: Stage B can refine it rather than relearn these shared relationships from scratch. Section 4 develops this procedure, which we evaluate experimentally.

Detailed mathematical analysis and comparison across multiple VLA models are provided in Appendix A.

4 EMBARK: Two-Stage Embodiment Alignment and Few-Shot Task Adaptation

4.1 Decoupling Embodiment Alignment from Task Learning

Given a pretrained VLA policy π_{θ_0} , conventional *task-specific fine-tuning* adapts the policy to each target task and robot setup using in-domain demonstrations [3], [23]. These demonstrations teach both task behavior and how the robot senses and moves, motivating a separate warmup stage for knowledge shared across tasks.

We instead propose EMBARK (Figure 2), a *task-agnostic post-training* approach that applies the three-component framework from Section 3 to decouple reusable embodiment alignment from target-task learning. Stage A uses task-agnostic simulation data to align the embodiment interface

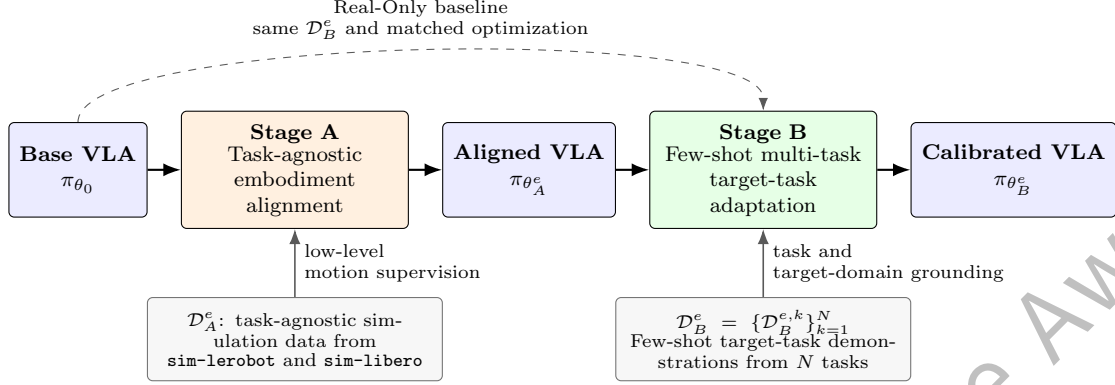


Figure 2: Overview of EMBARK. Stage A uses task-agnostic simulation data to align the target embodiment interface and produces a checkpoint shared across downstream tasks. Stage B aggregates few-shot target-task demonstrations from N tasks to learn task semantics and target-domain interactions. The dashed path denotes the matched Real-Only adaptation from the pretrained checkpoint.

Component	Stage A	Stage B
Vision encoder	Frozen	Updated
Language encoder	Frozen	Frozen
Embodiment interface	Updated	Updated
Generative action module	Model-dependent	Updated

Table 1: Trainable components for task-agnostic embodiment alignment (Stage A) and few-shot target-task adaptation (Stage B). The generative action module is frozen in GR00T Stage A; X-VLA also updates its action-generating transformer.

in Figure 1 for a target embodiment e , producing the embodiment-specific checkpoint $\pi_{\theta_A^e}$. Stage B then fine-tunes the aligned checkpoint on few-shot demonstrations collected in the target deployment environment to produce $\pi_{\theta_B^e}$. The resulting Stage A checkpoint can therefore be reused across multiple tasks, while Stage B supplies the task- and domain-specific behavior that is absent from the task-agnostic alignment data.

$$\pi_{\theta_0} \xrightarrow[\mathcal{D}_A^e]{\text{task-agnostic embodiment alignment}} \pi_{\theta_A^e} \xrightarrow[\{\mathcal{D}_B^{e,k}\}]{\text{few-shot target-task adaptation}} \pi_{\theta_B^e}. \quad (5)$$

Table 1 summarizes the trainable components in each stage. The language encoder remains frozen throughout both stages.

Stage A: task-agnostic embodiment alignment using simulation datasets. Stage A uses randomized, procedurally generated task-agnostic motion trajectories in simulation to align the embodiment interface of a pretrained VLA to a target embodiment. For each robot, we generate $\mathcal{D}_A^e$ from its hardware description, including its URDF. This configuration allows the simulator to produce the motion corpus without per-trajectory teleoperation or downstream task annotation. Each trajectory contains synchronized multi-view images, proprioceptive states, low-level motion instructions, and action chunks covering free-space motion, near-table motion, and simple gripper or contact interactions. The Stage A instructions omit object categories and downstream task

goals. The data match the top/wrist- camera, state/action, joint-order, and control-frequency interfaces used by the real robot. This construction trains the embodiment interface to represent the target robot’s state/action conventions and kinematic feasibility, including its reachable workspace, without downstream task demonstrations. Visual features and task-specific grounding and contact behavior are adapted in Stage B using demonstrations collected in the target deployment environment. In our implementation, `sim-lerobot` generates task-agnostic data for the SO-101 in Isaac Sim [24], whereas `sim-libero` generates both task-agnostic and task-conditioned data for the Franka Panda in LIBERO [25]. We use these generated task-agnostic data to post-train the pretrained GR00T N1.7 [5] and X-VLA [6] checkpoints. Model-specific configurations are discussed in Section 4.3.

Stage B: few-shot target-task adaptation. For a set of N downstream tasks in the target deployment environment, we collect a few-shot demonstration dataset $\mathcal{D}_B^{e,k}$ for each task $k \in \{1, \dots, N\}$. We then aggregate these datasets as $\mathcal{D}_B^e = \{\mathcal{D}_B^{e,k}\}_{k=1}^N$ and use the resulting multi-task dataset to further fine-tune the common Stage A checkpoint π_{θ_A} . These demonstrations teach task semantics, language grounding, and target-domain appearance, as well as task-specific contact behavior beyond the basic interaction primitives in Stage A. To isolate the benefit of simulation initialization, we train a *Real-Only* baseline from the pretrained checkpoint π_{θ_0} on the same aggregated dataset. The two branches use identical Stage B data, trainable components, optimization settings, and numbers of Stage B steps; they differ only in whether Stage B starts from the pretrained checkpoint or the Stage A checkpoint. This protocol amortizes embodiment alignment across the task set while preserving the task identity of each demonstration.

4.2 Simulation Data Generation for SO-101 and Franka Panda

Stages A and B share the same policy interface but differ in instruction semantics and data domain. Following the notation in Section 3, we instantiate the multimodal observation $\mathbf{o}_t$ in our experiments as

$$\mathbf{o}_t = (\mathbf{I}_t^{\text{top}}, \mathbf{I}_t^{\text{wrist}}, \mathbf{q}_t, \ell_t), \quad (6)$$

where $\mathbf{I}_t^{\text{top}}$ and $\mathbf{I}_t^{\text{wrist}}$ are synchronized camera images, $\mathbf{q}_t$ is the proprioceptive state, and ℓ_t is a language instruction. The state $\mathbf{q}_t$ describes the robot’s current configuration: arm joint and gripper positions for SO-101, and end-effector pose and finger positions for the Franka Panda. For each robot, Stage A and Stage B use matching state and action representations. The top and wrist camera images instantiate the multi-view input $\mathbf{I}_t^{1:K}$, while $\mathbf{q}_t$ and ℓ_t provide the proprioceptive and language inputs to the embodiment grounding and semantic understanding components, respectively.

Each task-agnostic action segment is paired with an instruction describing the intended low-level motion. These instructions are distinct from the task instructions used at test time. For example, a generated Stage A instruction may be “move the gripper forward,” whereas a downstream task instruction may be “pick up the blue block and place it near the red block”; the complete motion-only instruction pool and its paraphrases are provided in Appendix B. For the two embodiments evaluated in Section 5, we generate simulation data using Isaac Sim for SO-101 and LIBERO for Franka Panda. Figure 3 shows scenes that may look very different from, and less realistic than,



Figure 3: Representative simulated scenes used for task-agnostic motion data generation. The varied object layouts and appearances provide visual diversity without defining a downstream task goal.

the real scenes. This is acceptable because Stage A only requires matching the robot’s kinematics and interfaces, which greatly simplifies scene design.

SO-101 Task-Agnostic Motion Data with Isaac Sim. We develop `sim-lerobot`, an Isaac Sim data generator for the physical SO-101 arm used in our experiments. The simulated arm is constructed from its publicly available URDF description [26]. We calibrate the camera intrinsics and extrinsics in the physical setup and transfer them to the simulator. We also align the arm-table geometry by executing ten reference poses on the physical robot and matching the measured distances between the gripper base and the tabletop. Together with inverse kinematics, this calibrated configuration defines the reachable workspace used for data generation.

The generator produces task-agnostic motion in three complementary modes. *FM-air* samples target poses throughout the reachable workspace and executes free-space translations and rotations. *FM-table* varies the end-effector position and approach orientation near the tabletop. *FM-contact* performs simple interaction primitives, including approach, grasp, lift, lower, and release. Initial states are sampled to favor underrepresented workspace and joint-space regions. Consecutive waypoints are constrained in joint space and connected with smooth position and velocity profiles. The three modes are balanced at the episode level.

The recorder exports synchronized top and wrist images, joint states, arm and gripper commands, timestamps, and diagnostic signals in the same LeRobot format [27] as the physical SO-101 data, preserving the joint order and action conventions across simulation and real-world deployment.

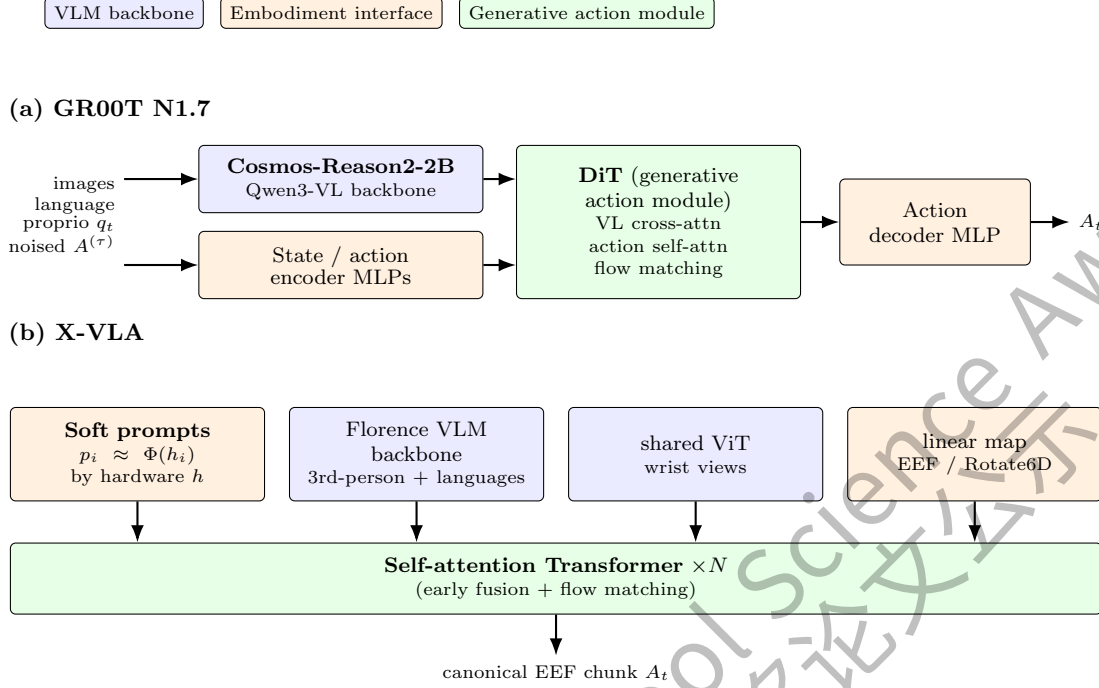


Figure 4: GR00T N1.7 and X-VLA as two instantiations of the three-component view in Figure 1. Blue is the VLM backbone. Orange is the embodiment interface: per-body state/action MLPs and an action decoder in GR00T N1.7; a soft-prompt library and a canonical EEf/Rotate6D linear map in X-VLA. Green is the generative action module: a flow-matching DiT with cross-attention in GR00T N1.7, and vanilla self-attention blocks in X-VLA.

Franka Panda Simulation Data with LIBERO. We develop `sim-libero`, a generator for the Franka Panda in the LIBERO environment. This generator follows a canonical policy interface at 20 Hz: an agent view and a wrist view at 256×256 resolution, an eight-dimensional state containing end-effector position, axis-angle orientation, and two finger positions, and a seven-dimensional relative OSC_POSE action containing translation, axis-angle rotation, and gripper opening. Simulation episodes are converted to the same LeRobot schema [27] as the official demonstrations before training. Raw MuJoCo actions, joint states, privileged poses, contacts, rewards, state-machine phases, and success signals are stored separately for diagnostics and never exposed to the policy.

The principal Stage A source is task-agnostic Franka Panda motion. We generate this corpus in `libero_object` and `libero_goal` scenes. Each episode samples a scene placement and executes a low-level translation, rotation, approach, or gripper primitive around the reset end-effector pose. Smooth interpolation and explicit workspace and table clearance bounds keep the trajectory executable. As with SO-101, every segment is assigned a motion-level instruction with no object category or task goal. The LIBERO scenes from `libero_object` and `libero_goal` serve only as diverse visual and geometric carriers.

4.3 Model-Specific Post-Training Configurations

We instantiate the three-component VLA view in Figure 4 with GR00T N1.7 and X-VLA. Both models follow the training scope in Table 1, with the model-specific embodiment interfaces and

optimization settings described below.

GR00T N1.7. GR00T N1.7 combines a Cosmos-Reason2-2B (Qwen3-VL) VLM backbone with a flow-matching DiT generative action module [5]. Its embodiment interface includes per-body state/action projectors and an action decoder. We use 30,000 Stage A steps at 10^{-4} and 5,000 Stage B steps at 5×10^{-5} . The category-specific weight slots in GR00T N1.7 act as embodiment prompts but differ from X-VLA’s soft-prompt tokens.

X-VLA. X-VLA uses a shared Florence/wrist-ViT representation, a canonical EEf-Rotate6D action interface, and domain-specific soft prompts [6]. Its embodiment interface includes these soft prompts and the canonical EEf/Rotate6D linear map. Because embodiment grounding also involves the state-action mapping, Stage A additionally updates the action-generating transformer while keeping the vision and language encoders frozen. Section 5.4.2 examines this choice. We use 30,000 Stage A steps and 5,000 Stage B steps at 10^{-4} .

4.4 Toward Generalist VLA Deployment

EMBARK aims to reduce the cost of adapting a pretrained VLA to a target robot by sharing motion warmup across downstream tasks. Stage B remains necessary for task goals, target-domain appearance, and contact-rich behavior. Our LIBERO-90 experiment uses one Stage A checkpoint to initialize a single Stage B policy trained jointly on 90 tasks. This tests the benefit of a shared initialization under few-shot supervision, rather than the separate adaptation and composition of individual skills.

Looking ahead, the aligned initialization could also support separately adapted skills, such as picking up a plate or opening a drawer. A VLM or LLM could then select and combine these skills toward a broader goal such as “clean the kitchen.” Demonstrating this possibility would require further experiments on skill adaptation, sequencing, and recovery from failures; the current work evaluates adaptation to the specified tasks, not arbitrary long-horizon task composition.

5 Experiments

Our experiments ask two questions. First, does task-agnostic motion warmup make adaptation to a robot embodiment more data- or optimization-efficient? Second, is the benefit consistent across downstream tasks, VLA backbones, and physical and simulated evaluation settings? We report physical-robot results and closed-loop simulation results separately because they use different metrics and support different claim strengths.

5.1 Experimental Setup

Unless stated otherwise, both embodiments follow the Stage A and Stage B training scope in Table 1.

5.1.1 Physical SO-101 tasks

We adapt X-VLA and GR00T N1.7 to the SO-101. The real data comprise a single-tile Pick-and-Place task, in which the robot picks up a tile and places it in a box, and a six-instruction LEGO sorting task, in which the robot sorts LEGO blocks by color according to instructions. Each suite has a fixed held-out test set of 50 episodes, disjoint from all training subsets. Training budgets are nested, so larger budgets only add demonstrations; both backbones use identical episode lists. Results are averaged across multiple random seeds. The primary comparison is between Real-Only adaptation and the same adaptation initialized with task-agnostic motion warmup.

Open-loop physical-policy accuracy is measured on held-out demonstrations using overall joint-angle MAE, fingertip-position MAE, end-effector orientation MAE, gripper accuracy, and error along the predicted action horizon. These teacher-forced metrics diagnose embodiment and action alignment, but they are not task success rates. We therefore do not use them alone to claim closed-loop physical competence.

To complement these open-loop diagnostics, we evaluate GR00T N1.7 on LEGO sorting in physical closed-loop trials. This evaluation tests whether improved embodiment and action alignment translates into executable task behavior; the complete protocol is described in Section 5.2.3.

5.1.2 LIBERO tasks

Within LIBERO [25], we use LIBERO-90—the 90-task short-horizon subset of LIBERO-100—for broad coverage of diverse manipulation tasks, and LIBERO-Spatial for a controlled low-shot study. Both use the Franka Panda, relative `OSC_POSE` actions, two camera views, and task-balanced budgets; B_k denotes k demonstrations per target task.

We compare Real-Only adaptation with two-stage task-agnostic motion warmup. Real-Only fine-tunes the released GR00T N1.7 checkpoint on target-task demonstrations, whereas the warmup variant first updates the embodiment interface with workspace-wide task-agnostic simulation motions and then fine-tunes on the same demonstrations.

Closed-loop test success rate is the primary LIBERO metric. All reported LIBERO results, including the learning curves and fixed-step endpoints, use a disjoint test set that is never used for training or checkpoint selection. The reported LIBERO values are means across multiple training seeds evaluated on the same fixed held-out test set. Each LIBERO-90 evaluation uses 450 rollouts (5 per task), while each LIBERO-Spatial test evaluation uses 200 rollouts (20 per task). We report both early and endpoint test success rate to distinguish faster adaptation from a higher converged ceiling.

5.2 Adaptation on a Physical Robot: SO-101

Under the physical-robot protocol in Section 5.1.1, we use 2,127 simulated SO-101 episodes (1.16 million frames) to update each backbone’s embodiment interface in Stage A, followed by Stage B adaptation using demonstrations collected from the physical robot in three real-world scenarios: Pick-and-Place, LEGO-same, and LEGO-rand. Pick-and-Place asks the robot to pick up a tile and place it in a box. Both LEGO sorting settings contain red, green, and blue blocks and use instructions that name a block to pick up and another block to place it near. LEGO-same resets the arm to a canonical initial pose before each instruction, whereas LEGO-rand varies the initial

joint configuration across episodes. The three scenarios are separate data collections, so their available nested budgets differ slightly: B_{50}/B_{180} for Pick-and-Place, B_{50}/B_{170} for LEGO-same, and B_{50}/B_{154} for LEGO-rand. Stage A takes approximately 11 hours on a single A800 GPU for GR00T N1.7 and 5.5 hours on a single A800 GPU for X-VLA. For X-VLA and GR00T N1.7, we compare task-agnostic motion warmup followed by Stage B adaptation with matched Real-Only baselines trained on the same real demonstrations and data budgets.

This paired design tests whether the benefit of task-agnostic motion warmup transfers across two VLA backbones that conform to the three-component framework in Section 3. It also tests whether the open-loop diagnostics defined in Section 5.1.1 provide useful indicators of physical adaptation. On LEGO-rand, the closed-loop results in Section 5.2.3 corroborate the improvements indicated by the open-loop evaluations at both data budgets. In these experiments, the open-loop improvements are directionally consistent with the physical closed-loop results, suggesting that open-loop diagnostics may serve as inexpensive monitoring signals, while closed-loop success remains the definitive measure of deployed task performance.

5.2.1 X-VLA

In this experiment, we measure following metrics. *Overall joint-angle MAE* measures joint-space prediction accuracy, while *fingertip-position MAE* measures whether the arm reaches the correct task-space location before grasping or placing an object. *End-effector orientation MAE* and *gripper accuracy* characterize terminal manipulation: whether the end effector is correctly oriented and whether its open-close decision is correct.

Table 2 exposes a large zero-shot deployment gap. Without physical-task adaptation, X-VLA has 42.49–52.22 degrees of overall joint-angle MAE, 25.66–27.71 cm of fingertip-position MAE, and 113.48–118.80 degrees of end-effector orientation MAE. Its gripper accuracy is only 37.06–42.55%, near chance for the binary open-close diagnostic. Although these teacher-forced metrics are not closed-loop success rates, near-chance gripper decisions and gross geometric error make the pretrained policy operationally uncalibrated and rule out direct deployment.

Real-Only adaptation (Table 3) reduces overall joint-angle MAE to 4.34–7.17 degrees, fingertip-position MAE to 2.60–3.77 cm, and end-effector orientation MAE to 7.89–16.01 degrees, while raising gripper accuracy to 87.78–90.72%. This order-of-magnitude correction shows that post-training is not optional refinement but the mechanism that converts a pretrained model into an embodiment-compatible policy. The resulting question is therefore not whether post-training is needed, but how to make it more data-efficient and more effective when physical demonstrations are costly.

Task-agnostic motion warmup improves all four metrics in every task-budget comparison. Relative to Real-Only adaptation, it reduces overall joint-angle MAE by 15.7–33.1%, fingertip-position MAE by 17.2–29.3%, and end-effector orientation MAE by 16.4–35.9%, while increasing gripper accuracy by 2.4–5.7%. The benefit persists at the larger budgets but generally becomes smaller, indicating that task-agnostic motion warmup provides a better initialization and improves real-data efficiency rather than replacing physical demonstrations. The warmup advantage also persists across the predicted action horizon: for X-VLA on Pick-and-Place at B_{50} , overall joint-angle MAE is reduced from 7.17 to 4.87 degrees at $k = 0$, and from 16.88 to 14.99 degrees at $k = 29$.

Because gripper states can be class-imbalanced, we additionally checked class-balanced metrics.

Metric	Pick-and-Place	LEGO-same	LEGO-rand
Overall Joint-Angle MAE (deg) ↓	52.22	47.51	42.49
Gripper Accuracy (%) ↑	42.55	42.14	37.06
Fingertip-Position MAE (cm) ↓	27.31	25.66	27.71
End-Effector Orientation MAE (deg) ↓	118.51	113.48	118.80

Table 2: Held-out open-loop results for the **raw** pretrained X-VLA checkpoint.

(a) **Pick-and-Place: Pick tile and drop into box**

Metric	B_{50}		B_{180}	
	Real-Only	Warmup → Real	Real-Only	Warmup → Real
Overall Joint-Angle MAE (deg) ↓	7.17	4.87 (−32.1%)	5.14	3.60 (−29.9%)
Gripper Accuracy (%) ↑	89.70	93.06 (+3.7%)	90.72	94.95 (+4.7%)
Fingertip-Position MAE (cm) ↓	3.77	2.67 (−29.3%)	3.05	2.18 (−28.6%)
End-Effector Orientation MAE (deg) ↓	16.01	10.32 (−35.6%)	9.96	7.21 (−27.7%)

(b) **LEGO-same: LEGO sorting (same start point) according to instruction**

Metric	B_{50}		B_{170}	
	Real-Only	Warmup → Real	Real-Only	Warmup → Real
Overall Joint-Angle MAE (deg) ↓	5.58	4.12 (−26.2%)	4.34	3.66 (−15.7%)
Gripper Accuracy (%) ↑	89.07	92.47 (+3.8%)	90.69	93.63 (+3.2%)
Fingertip-Position MAE (cm) ↓	3.07	2.22 (−27.5%)	2.60	2.15 (−17.2%)
End-Effector Orientation MAE (deg) ↓	11.42	7.93 (−30.6%)	8.65	7.23 (−16.4%)

(c) **LEGO-rand: LEGO sorting (random start point) according to instruction**

Metric	B_{50}		B_{154}	
	Real-Only	Warmup → Real	Real-Only	Warmup → Real
Overall Joint-Angle MAE (deg) ↓	6.15	4.11 (−33.1%)	4.44	3.34 (−24.7%)
Gripper Accuracy (%) ↑	87.78	92.75 (+5.7%)	90.63	92.80 (+2.4%)
Fingertip-Position MAE (cm) ↓	3.57	2.72 (−24.0%)	2.76	2.16 (−21.6%)
End-Effector Orientation MAE (deg) ↓	11.44	7.34 (−35.9%)	7.89	5.67 (−28.1%)

Table 3: X-VLA held-out open-loop results on physical SO-101 tasks. Real-Only adaptation is compared with task-agnostic motion warmup followed by adaptation on the same real data. Parenthetical percentages report relative changes from Real-Only.

For the X-VLA Pick-and-Place B_{50} setting, *balanced accuracy* is 89.75% for Real-Only and 92.74% after warmup. The corresponding *ordinary accuracy* values are 89.70% and 93.06%, respectively. The similar improvement in balanced accuracy suggests that the gain is not solely due to predicting the majority gripper state.

By broadening the initial-state distribution, LEGO-rand requires the same visual-language command to be realized through different trajectories. Consistent gains in both LEGO settings therefore suggest that task-agnostic motion warmup improves adaptation beyond memorizing a canonical approach path. Section 5.4 examines the training schedule and Stage A trainable components underlying these gains.

Metric	Pick-and-Place	LEGO-same	LEGO-rand
Overall Joint-Angle MAE (deg) ↓	11.18	50.67	35.23
Gripper Accuracy (%) ↑	50.95	50.83	49.57
Fingertip-Position MAE (cm) ↓	6.15	31.40	30.12
End-Effector Orientation MAE (deg) ↓	22.63	99.10	83.16

Table 4: Held-out open-loop results for the **raw** pretrained GR00T N1.7 checkpoint.

(a) **Pick-and-Place: Pick tile and drop into box**

Metric	B_{50}		B_{180}	
	Real-Only	Warmup → Real	Real-Only	Warmup → Real
Overall Joint-Angle MAE (deg) ↓	3.42	3.18 (−6.9%)	3.19	2.80 (−12.3%)
Gripper Accuracy (%) ↑	92.99	93.22 (+0.2%)	93.94	94.21 (+0.3%)
Fingertip-Position MAE (cm) ↓	1.88	1.73 (−8.3%)	1.78	1.55 (−13.1%)
End-Effector Orientation MAE (deg) ↓	5.64	5.12 (−9.3%)	5.33	4.55 (−14.5%)

(b) **LEGO-same: LEGO sorting (same start point) according to instruction**

Metric	B_{50}		B_{170}	
	Real-Only	Warmup → Real	Real-Only	Warmup → Real
Overall Joint-Angle MAE (deg) ↓	3.25	3.01 (−7.5%)	3.41	2.65 (−22.5%)
Gripper Accuracy (%) ↑	93.63	93.47 (−0.2%)	95.45	95.35 (−0.1%)
Fingertip-Position MAE (cm) ↓	1.98	1.60 (−18.9%)	2.10	1.42 (−32.6%)
End-Effector Orientation MAE (deg) ↓	6.22	5.31 (−14.7%)	5.85	4.71 (−19.6%)

(c) **LEGO-rand: LEGO sorting (random start point) according to instruction**

Metric	B_{50}		B_{154}	
	Real-Only	Warmup → Real	Real-Only	Warmup → Real
Overall Joint-Angle MAE (deg) ↓	2.67	2.42 (−9.2%)	2.53	2.20 (−13.0%)
Gripper Accuracy (%) ↑	91.15	92.23 (+1.2%)	94.69	93.55 (−1.2%)
Fingertip-Position MAE (cm) ↓	1.62	1.39 (−14.2%)	1.71	1.32 (−22.5%)
End-Effector Orientation MAE (deg) ↓	3.94	3.43 (−13.0%)	4.16	3.32 (−20.3%)

Table 5: GR00T N1.7 held-out open-loop results on physical SO-101 tasks. Real-Only adaptation is compared with task-agnostic motion warmup followed by adaptation on the same real data. Parenthetical percentages report relative changes from Real-Only.

5.2.2 GR00T N1.7

To test whether the warmup benefit is specific to X-VLA, we repeat the matched comparison between Real-Only and motion-warmup-initialized adaptation with GR00T N1.7, which follows the same three-component view. The compared branches use the same real-data budgets and both unfreeze the vision encoder during physical-task adaptation.

The pretrained-checkpoint results across all three physical tasks are summarized in Table 4. GR00T N1.7 exhibits the same qualitative deployment gap. Zero-shot gripper accuracy is 49.57–50.95%, essentially chance, while the two LEGO evaluations have 35.23–50.67 degrees of overall joint-angle MAE, 30.12–31.40 cm of fingertip-position MAE, and 83.16–99.10 degrees of end-effector orientation MAE. Pick-and-Place errors are smaller but still far above the adapted results. The

same failure in two distinct pretrained VLAs shows that the post-training bottleneck is not X-VLA-specific.

Table 5 shows a consistent improvement in continuous action geometry. Across all six task-budget comparisons, task-agnostic motion warmup reduces overall joint-angle MAE by 6.9–22.5%, fingertip-position MAE by 8.3–32.6%, and end-effector orientation MAE by 9.3–20.3%. Gripper accuracy changes by at most 1.2% relative: it improves at both Pick-and-Place budgets and at LEGO-rand B_{50} , and decreases slightly in the other three comparisons. Thus the main effect is a more accurate reaching and pose prior, rather than a uniform change in the binary gripper decision.

The gain spans two task families, two LEGO data collections, and both low- and high-data regimes. Its recurrence in X-VLA and GR00T N1.7 supports our hypothesis that task-agnostic motion warmup transfers across backbones that expose the same three functional components, rather than relying on an X-VLA-specific mechanism.

Section 5.2.3 next tests whether these improvements translate into higher physical closed-loop success.

5.2.3 Closed-Loop Evaluation

The preceding open-loop evaluations measure action prediction under teacher-forced observations. They do not expose errors that accumulate when a policy acts on its own observations, nor failures caused by real sensing, latency, contact, and grasp uncertainty. We therefore test whether the same method ordering persists during physical closed-loop execution.

We conduct the evaluation with GR00T N1.7 on the physical SO-101. Each language-conditioned LEGO-sorting scene contains one red, one green, and one blue block. An instruction names a source and destination color, for example, “pick up the blue block and place it near the red block,” while the third block serves as a distractor. The policy observes fixed top- and wrist-camera views together with the robot state, and controls the SO-101 arm and parallel gripper until task completion or a 60-second timeout.

We compare Real-Only and task-agnostic-motion-warmup policies at B_{50} and B_{154} , corresponding to 50 and 154 physical demonstrations. At each budget, the two policies are evaluated on the same 20 scenes with matched block layouts, instructions, initial arm poses, and policy-sampling seeds. This four-policy matrix yields 80 physical rollouts. A rollout succeeds only if the specified block is grasped, transported, and released near the specified destination. Wrong-object manipulation, missed grasps, dropped blocks, and failure to release before timeout are failures; outcomes are assigned from the recorded physical execution rather than final 2D proximity alone.

Because each scene configuration must be reproduced at least four times, once for each policy variant, small differences in manual block placement or arm pose could otherwise confound the paired comparison. To enforce matched initial conditions, every trial uses automated scene verification followed by a scripted robot reset. The operator interface in Figure 5 detects the three blocks in the live top-camera view, overlays their prescribed positions, and enables the rollout only after the observed layout is within tolerance. A reset controller then returns the SO-101 to the prescribed initial joint pose. This procedure ensures that each policy is evaluated from the same object layout and robot state.

Table 6 reports the paired results. Task-agnostic motion warmup raises the success point estimate from 60% to 70% at B_{50} and from 70% to 75% at B_{154} . In the paired comparison,



Figure 5: Operator interface for reproducing the initial block layout in physical closed-loop trials. Colored overlays identify the detected blocks and their prescribed positions; the trial is enabled only after the scene passes the position-tolerance check.

Real-data budget	Real-Only	Warmup $\rightarrow$ Real	Difference
B_{50}	12/20 (60%)	14/20 (70%)	+10 pp
B_{154}	14/20 (70%)	15/20 (75%)	+5 pp

Table 6: Physical closed-loop success on matched SO-101 LEGO-sorting scenes. Each budget uses 20 paired scenes. Task-agnostic motion warmup improves the point estimate at both budgets.

warmup converts three Real-Only failures into successes and regresses on one scene at B_{50} , for a net gain of two successes. At B_{154} , four failures become successes while three successful Real-Only trials regress, for a net gain of one. Figure 6 shows representative paired gains: warmup corrects a wrong-object outcome at B_{50} and a failure to release at B_{154} .

Across 80 physical rollouts, warmup therefore retains the same method ordering at both real-data budgets as in the open-loop evaluation. This experiment provides deployment-level corroboration rather than a precise estimate of the effect size: larger prospectively frozen evaluations across training seeds, task families, and physical robots remain future work.

5.3 Adaptation in LIBERO: Franka Panda

The preceding experiments showed that task-agnostic motion warmup improves adaptation for SO-101 with both X-VLA and GR00T N1.7 in open- and closed-loop evaluation. We next test whether this benefit extends beyond a single robot embodiment by applying EMBARK to the kinematically distinct Franka Panda in LIBERO, using GR00T N1.7 as the common VLA architecture.

Before evaluating the proposed warmup, we validate our GR00T N1.7 post-training implementation within the same LeRobot framework [27]. Starting from the pretrained checkpoint, we reproduce the official LIBERO-Spatial adaptation protocol for 5,000 steps and compare the resulting policy with the official GR00T N1.7 checkpoint post-trained on LIBERO-Spatial for the

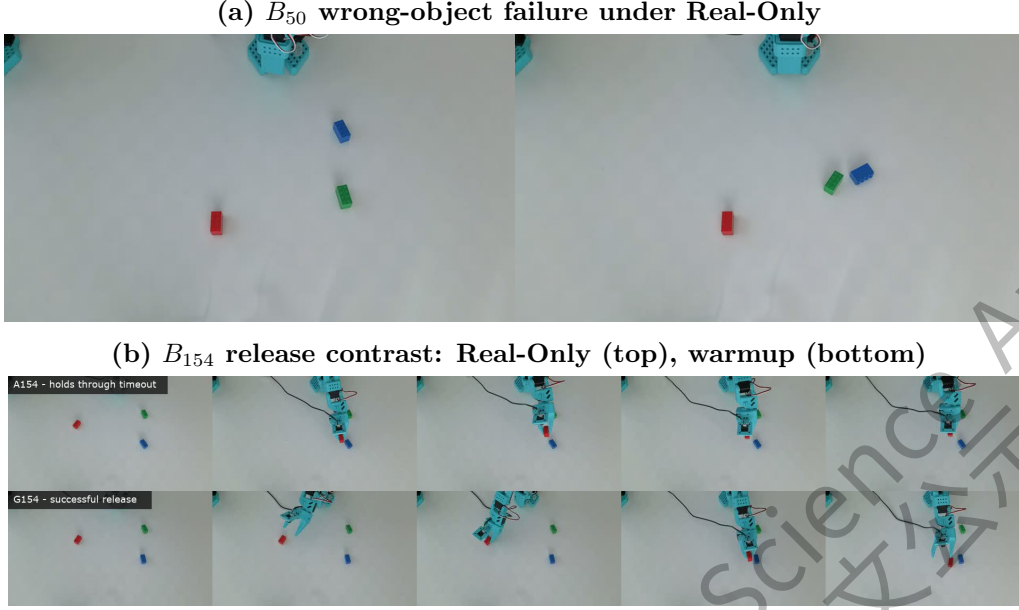


Figure 6: Representative closed-loop outcomes. (a) Initial (left) and final (right) views of a Real-Only B_{50} rollout. The policy fumbles the requested blue block, grasps the nearby green distractor, and releases green away from the red destination. (b) Temporal sequences for the same B_{154} scene: Real-Only grasps and transports red but holds it through timeout (top), whereas the warmup policy completes the release near blue (bottom).

Stage B budget	Real-Only	Warmup $\rightarrow$ Real	Stage B budget	Real-Only	Warmup $\rightarrow$ Real
B_1	58.9%	59.1%	B_1	62.5%	69.0%
B_2	68.2%	72.9%	B_2	80.5%	83.5%
B_5	78.9%	82.2%	B_5	91.5%	92.5%

Table 7: LIBERO-90 test success rate. After 5,000 Stage B optimizer steps, with 1, 2, or 5 demonstrations per task.

Table 8: LIBERO-Spatial test success rate. After 5,000 Stage B optimizer steps, with 1, 2, or 5 demonstrations per task.

Franka Panda. This official checkpoint succeeds in 192/200 rollouts (96.0%), while our reproduction achieves 193/200 (96.5%); by contrast, the pretrained checkpoint achieves 0/200. This close agreement validates our implementation and confirms that the resulting task performance arises from adaptation rather than zero-shot transfer.

5.3.1 Task-agnostic motion warmup on LIBERO-90

LIBERO-90 comprises 90 language-conditioned manipulation tasks distributed across kitchen, living-room, and study scenes. It includes many objects and scene layouts, articulated fixtures such as drawers, cabinets, stoves, and microwaves, stacking, pushing, and varied placement relations. It is therefore more diverse in perception, contact modes, motion range, and task semantics without making long-horizon planning the defining difference. This breadth makes it a suitable testbed for low-shot adaptation across a heterogeneous task distribution.

For task-agnostic motion warmup, Stage A uses a simulation corpus of 3,000 episodes (610.4K

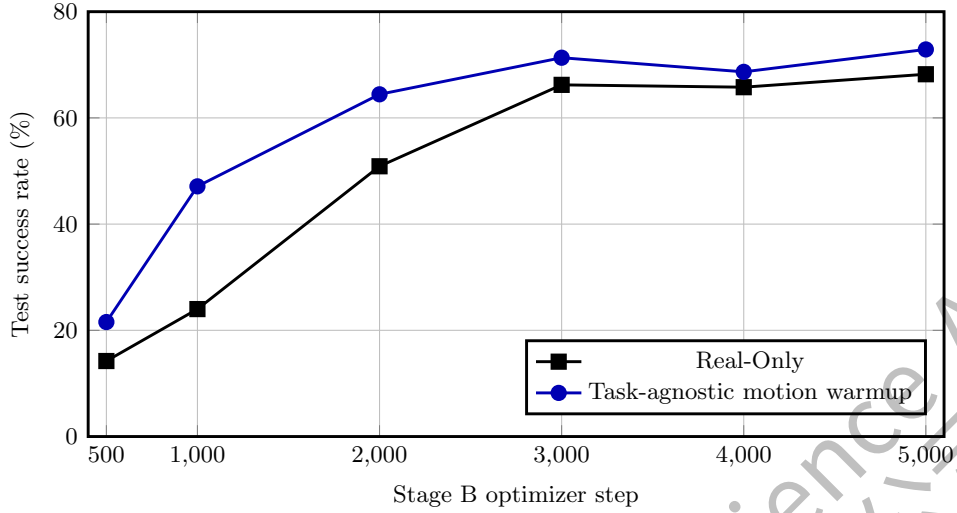


Figure 7: LIBERO-90 B_2 test-success curve. Each point evaluates 450 rollouts over the same 90 tasks and five held-out test states per task. These states are not used for training or checkpoint selection.

frames) collected across 39 randomized scene configurations. The trajectories contain free-space motions and grasping interactions executed by the Franka Panda. The resulting Stage A checkpoint provides a shared initialization for Stage B.

We consider three Stage B budgets, B_1 , B_2 , and B_5 , which provide one, two, or five demonstrations per task, respectively (90, 180, or 450 demonstrations in total). This protocol tests whether task-agnostic motion warmup improves adaptation when the target tasks are diverse and supervision for each individual task is severely limited.

Table 7 reports endpoint test success after 5,000 Stage B steps. Task-agnostic motion warmup provides little benefit at B_1 (59.1% versus 58.9% for Real-Only), but improves success by 4.7 and 3.3 percentage points at B_2 and B_5 , respectively. These results indicate that warmup complements rather than replaces target-task demonstrations: its benefit emerges once Stage B has enough supervision to ground the diverse task objectives. To examine how adaptation evolves with additional optimization, Figure 7 traces the test success rate over the Stage B training trajectory at B_2 . Task-agnostic motion warmup improves the test success rate throughout the reported trajectory, with the largest gap at 1,000 optimizer steps.

Task-agnostic motion warmup raises the test success rate from 24.0% to 47.1% at step 1,000 and from 50.9% to 64.4% at step 2,000. The gap narrows to 4.7 percentage points by step 5,000, supporting faster target-task adaptation at a fixed real-data budget more strongly than a higher converged ceiling.

Taken together, Table 7 and Figure 7 show that task-agnostic motion warmup acts primarily as an optimization prior. The results provide initial evidence that EMBARK enables faster adaptation while retaining a persistent improvement in the final test success rate.

5.3.2 Comparison with LIBERO-Spatial

LIBERO-Spatial is a simpler target distribution: its ten tasks reuse a narrow bowl-on-plate manipulation family and primarily vary the requested spatial relation. Although the language instructions

differ, the tasks share closely related goals and manipulation structure, allowing substantial knowledge transfer across tasks in the suite. This setting therefore tests whether task-agnostic motion warmup remains useful when the target-task distribution has limited diversity.

Table 8 summarizes the test success rates. At B_1 , EMBARK improves the test success rate from 62.5% to 69.0%, a gain of 6.5 percentage points. At B_5 , Real-Only adaptation already reaches 91.5%, while EMBARK reaches 92.5%, reducing the gain to 1.0 percentage point. Thus, when the target tasks are relatively homogeneous, a small number of target-task demonstrations can provide enough signal to learn most of the required embodiment prior, leaving limited room for task-agnostic motion warmup to improve the endpoint test success rate.

5.4 Ablation Studies

We examine two design choices in EMBARK using X-VLA on the SO-101 Pick-and-Place task at B_{50} : separating warmup from real-data adaptation, and selecting the components updated during warmup. Both comparisons use the same 50 held-out episodes and the open-loop metrics defined in Section 5.1.1.

5.4.1 Two-Stage Training versus Direct Co-Training

To assess whether adding simulation data alone accounts for the gain, Table 9 compares direct Sim-and-Real co-training with Warmup $\rightarrow$ Real and Real-Only adaptation. Co-training mixes simulation and real data in one stage; the two-stage recipe adds a dedicated real-data adaptation phase.

Metric	Real-Only	Sim-and-Real	Warmup $\rightarrow$ Real
Overall Joint-Angle MAE (deg) $\downarrow$	7.17	6.09	4.87
Gripper Accuracy (%) $\uparrow$	89.70	88.96	93.06
Fingertip-Position MAE (cm) $\downarrow$	3.77	3.74	2.67
End-Effector Orientation MAE (deg) $\downarrow$	16.01	13.14	10.32

Table 9: X-VLA training-strategy comparison on Pick-and-Place at B_{50} .

Compared with direct co-training, Warmup $\rightarrow$ Real reduces joint-angle, fingertip-position, and orientation MAE by 20.1%, 28.7%, and 21.5%, respectively, and improves gripper accuracy by 4.10 percentage points.

These results favor the two-stage recipe and suggest that dedicated real-data adaptation helps refine visual grounding and manipulation behavior under simulation-to-real differences.

5.4.2 Trainable Components during Warmup

X-VLA’s soft prompts condition action generation, but its action-related projections and transformer also participate in mapping proprioception and visual features to motion [6]. Embodiment grounding is therefore not confined to prompt embeddings. We compare updating only soft prompts with updating both soft prompts and the action head during Stage A, keeping vision and language encoders frozen. Both settings use the same Stage B adaptation scope: vision encoder, soft prompts, and action head, with the language encoder frozen.

Metric	Soft-Prompt-Only Warmup $\rightarrow$ Real	Warmup $\rightarrow$ Real (Soft Prompts + Action Head)
Overall Joint-Angle MAE (deg) $\downarrow$	6.33	4.87
Gripper Accuracy (%) $\uparrow$	91.40	93.06
Fingertip-Position MAE (cm) $\downarrow$	3.40	2.67
End-Effector Orientation MAE (deg) $\downarrow$	13.67	10.32

Table 10: X-VLA warmup-component comparison on Pick-and-Place at B_{50} . Both recipes use real-data adaptation after warmup.

Table 10 compares the two warmup configurations using the same simulation data and Stage B adaptation protocol. Updating both soft prompts and the action head reduces joint-angle MAE from 6.33 to 4.87 degrees (23.1%), compared with updating soft prompts alone, and also improves fingertip-position MAE, orientation MAE, and gripper accuracy.

Prompt-only warmup relies on the frozen generator’s existing motion capabilities. Updating the action head additionally allows its state-action mapping and temporal motion distribution to adapt to the target robot. This provides a plausible explanation for the observed gain and supports joint warmup for X-VLA, without establishing that every updated component is necessary or that the gain arises solely from kinematic learning.

6 Conclusion

We introduced EMBARK, a two-stage framework that separates task-agnostic embodiment adaptation from few-shot target-task learning. Stage A uses embodiment-matched simulation motion to update the target robot’s state/action interface and kinematic feasibility constraints; Stage B adds task semantics, visual grounding, and target-domain interactions. Across X-VLA and GR00T N1.7 on the physical SO-101, this warmup consistently reduces held-out motion errors. The paired physical closed-loop LEGO-sorting study remains exploratory and does not establish a statistically reliable success-rate improvement. On LIBERO-90 with two demonstrations per task, warmup raises test success from 24.0% to 47.1% after 1,000 Stage B steps and retains a 4.7-point advantage after 5,000 steps. The smaller LIBERO-Spatial endpoint gain indicates that the prior is most useful early in optimization and under diverse, low-shot target distributions.

The evidence therefore supports embodiment-matched, task-agnostic simulation exposure as a reusable embodiment and optimization prior that complements rather than replaces task supervision. Current limitations include the small physical evaluation, limited-seed LIBERO comparisons on a fixed test set, and the additional data and compute introduced by Stage A. Multi-seed testing, compute-matched and mismatched-embodiment controls, and evaluation across independently acquired task families are important next steps. EMBARK provides a practical direction for reducing repeated physical-data collection: align how a robot acts before asking scarce demonstrations to teach what it should do.

Acknowledgments

The author gratefully thanks Professor Wei Xu for generously providing the computing resources that made the experiments possible.

This work began as a high-school research project motivated by the author’s interest in robotics. The initial goal was to teach a robot to play Mahjong with my grandmother. She grew up in Sichuan, where Mahjong is not merely a game but a favorite way of bringing people together. As she and her friends grew older and found it harder to go out, their gatherings would sometimes be one player short—the familiar Mahjong predicament of *san que yi*. I therefore imagined a robot that could take the fourth seat at the table and keep her company. That idea remains in the physical Pick-and-Place experiment, where the robot manipulates a Mahjong tile. Discussions with Professor Xu broadened the project into a study of robot-arm manipulation and vision-language-action systems. The architectural survey in Appendix A was the first outcome of this exploration. It helped identify embodiment adaptation as a concrete research problem and led to the present EMBARK framework. This work was an independent personal research project conducted by the author. The author independently implemented the software, conducted the experiments and evaluations, and analyzed the results. Professor Xu provided guidance on the research topic and overall direction, as well as feedback on the manuscript and other customary academic supervision, entirely on an unpaid basis.

Disclosure of AI assistance. This project was conducted over approximately seven to eight months. During that period, AI tools were used daily as auxiliary tools in the development and debugging of the experimental and simulator code. The tools used included different versions of Claude (Opus 4.6, Opus 4.7, and Opus 5) and GPT (versions 5.5, 5.6, and 6). Their use was limited to code assistance and debugging; the research question, experimental design, core arguments, experiments, and interpretation of the results were completed by the author. After completing the main body of the manuscript independently, the author used GPT Astra-6 to help organize the manuscript’s logical structure, identify and remove repetitive passages, condense wording, emphasize the main points, check grammar, and refine the academic English. AI-generated text was not used as a substitute for the author’s own thinking or writing, and all suggestions, code, and revisions were reviewed and verified by the author.

A Architectural Analysis of VLAs

VLA architectures are evolving rapidly, but several recent systems can be usefully analyzed through a common three-component functional view, illustrated conceptually in Fig. 1 and instantiated by GR00T N1.7 [5], X-VLA [6], $\pi_{0.7}$ [4], and Qwen-VLA [28]. This structure factors three sources of knowledge that a generalist robot policy must combine: (i) what the world *is* and what the user *asked*, (ii) how *this* body can move, and (iii) which physically feasible motions realize the instruction.

VLM backbone for semantic world understanding. A pretrained VLM backbone encodes multi-view images and the language instruction into a sequence of tokens. Given multi-view camera observations and a task instruction, the VLM backbone extracts semantically meaningful features

that characterize the scene, relevant objects, their relationships, and the intended task. Its operation can be abstractly expressed as

$$\mathbf{z}_t^{\text{sem}} = f_{\text{VLM}}(\mathbf{I}_t^{1:K}, \ell), \quad (7)$$

where f_{VLM} denotes the pretrained VLM backbone and $\mathbf{z}_t^{\text{sem}}$ represents the resulting semantic representation.

Initialized from large-scale image-text pretraining, the VLM supplies broad visual and linguistic priors, but it does not inherently represent how a particular manipulator should move its joints or end effector. Robot trajectory training therefore remains necessary to associate semantic representations with physically executable interactions.

Embodiment interface. The *embodiment interface* is the module that *aligns* the VLM backbone’s embodiment-agnostic features with the physical constraints and action regularities of a concrete robot.

Let $\mathbf{e}$ denote information associated with a particular robot embodiment. The embodiment-specific representation can be abstractly formulated as

$$\mathbf{z}_t^{\text{emb}} = P_\phi(\mathbf{q}_t, \mathbf{e}), \quad (8)$$

where P_ϕ denotes an embodiment-aware projection function parameterized by ϕ , and $\mathbf{z}_t^{\text{emb}}$ is the resulting embodiment-aware representation. Depending on the implementation, $\mathbf{e}$ may explicitly or implicitly encode the dimensionality and definition of joints and actions, the robot kinematic structure, control modes, arm configuration, workspace characteristics, and other hardware-dependent properties.

The embodiment interface is therefore not a cosmetic adapter. It is the place where real-robot physical constraints are injected so that later policy training can spend capacity on *what* to do rather than on rediscovering *how this robot is allowed to move*. This paper’s post-training recipe in Section 4 operates primarily on this interface: task-agnostic motion on a new embodiment is used to specialize the interface before target-task adaptation.

Generative action module for continuous control. A prominent design choice in recent VLA systems is to employ a Diffusion Transformer (DiT) or a closely related flow-matching Transformer to model continuous action chunks. Unlike autoregressive action-token prediction, which typically discretizes actions and generates them sequentially, diffusion- and flow-based policies directly model continuous action distributions. This property is particularly desirable for robot manipulation, where valid behaviors can be multimodal and small numerical errors in joint-space or end-effector commands may substantially affect task success.

For clarity, we distinguish the robot time index t from the continuous flow time $\tau \in [0, 1]$. Let $\mathbf{o}_t = (\mathbf{I}_t^{1:K}, \ell, \mathbf{q}_t)$ denote the conditioning information available to the generative action module at robot time step t .

For a ground-truth action chunk $\mathbf{A}$ sampled from the robot demonstration dataset and an initial noise sample $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, a linear probability path between the noise sample and the demonstrated action can be defined as $\mathbf{A}^\tau = (1 - \tau)\epsilon + \tau\mathbf{A}$, $\tau \in [0, 1]$.

Model	VLM backbone	Embodiment interface	Generative action module
$\pi_{0.7}$ [4]	Gemma3 4B VLM + MEM history encoder	language + episode metadata + subgoal images	flow-matching action expert
GR00T N1.7 [5]	Cosmos-Reason2-2B (Qwen3-VL)	per-embodiment state/action MLPs	flow-matching DiT
X-VLA [6]	Florence + shared wrist ViT	soft prompts + EEf/Rotate6D alignment + proprio-action linear map	flow-matching Transformer encoder
Qwen-VLA [28]	Qwen3.5	embodiment-aware text prompts	flow-matching DiT
SmolVLA [29]	SmolVLM2 (SigLIP + SmolLM2)	linear sensorimotor-state token projection	flow-matching transformer w/ interleaved cross/self-attn

Table 11: Representative VLA systems organized by the three-component view of Section A. The embodiment interface aligns joints, cameras, kinematics, and low-level motion with the shared backbone.

The corresponding target velocity field is obtained by differentiating the probability path with respect to τ :

$$\mathbf{u}_\tau = \frac{\partial \mathbf{A}^\tau}{\partial \tau} = \mathbf{A} - \boldsymbol{\epsilon}. \quad (9)$$

The conditional flow-matching objective can then be written as

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{\substack{(\mathbf{o}_t, \mathbf{A}) \sim \mathcal{D}, \\ \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \\ \tau \sim \mathcal{U}(0,1)}} \left[\|v_\theta(\mathbf{A}^\tau, \tau \mid \mathbf{o}_t) - (\mathbf{A} - \boldsymbol{\epsilon})\|_2^2 \right], \quad (10)$$

where $\mathcal{D}$ denotes the robot demonstration dataset, $\mathbf{A}^\tau$ is an intermediate noisy action chunk along the probability path, and v_θ denotes the conditional neural vector field. The conditioning variable $\mathbf{o}_t$ contains visual observations, the language instruction, the robot proprioceptive state, and, when applicable, embodiment information.

At inference time, an action chunk is initialized from Gaussian noise, $\mathbf{A}^0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and progressively transformed toward a valid robot action trajectory by integrating the learned ordinary differential equation

$$\frac{d\mathbf{A}^\tau}{d\tau} = v_\theta(\mathbf{A}^\tau, \tau \mid \mathbf{o}_t), \quad \tau \in [0, 1]. \quad (11)$$

The resulting sample $\mathbf{A}^1$ represents the predicted continuous action chunk.

This action-learning stage is where semantic knowledge becomes grounded in physical interactions. During behavior cloning or flow-matching training on robot demonstrations, the policy learns which visual and linguistic features are relevant to actions, how objects should be approached and manipulated, and how high-level instructions correspond to temporally extended motor trajectories. Thus, task semantics and language grounding should not be attributed exclusively to either the VLM backbone or the generative action module: the VLM backbone supplies semantic priors, while robot-policy training establishes the correspondence between these priors and executable object interactions. The generative action module additionally captures the multimodality

and temporal structure of the resulting motor behavior.

B Stage A Instruction Pools

The following is the complete set of low-level language templates used by the SO-101 task-agnostic motion generator. For each segment, one paraphrase is sampled from the pool associated with the commanded primitive. The templates describe only the motion and deliberately omit object identity, color, and downstream task goals.

```
INSTRUCTION_POOLS: dict[str, tuple[str, ...]] = {
    "translate_forward": (
        "move the gripper forward",
        "move the arm forward",
        "move the gripper ahead",
    ),
    "translate_backward": (
        "move the gripper backward",
        "move the arm backward",
        "pull the gripper back",
    ),
    "translate_left": (
        "move the gripper left",
        "move the arm to the left",
        "move the gripper to the left",
    ),
    "translate_right": (
        "move the gripper right",
        "move the arm to the right",
        "move the gripper to the right",
    ),
    "translate_up": (
        "move the gripper up",
        "raise the gripper",
        "move the arm upward",
    ),
    "translate_down": (
        "move the gripper down",
        "lower the gripper",
        "move the arm downward",
    ),
    "tilt_up": (
        "tilt the gripper up",
        "tilt the gripper upward",
        "pitch the gripper up",
    ),
}
```

```

    ),
    "tilt_down": (
        "tilt the gripper down",
        "tilt the gripper downward",
        "pitch the gripper down",
    ),
    "roll_cw": (
        "rotate the wrist clockwise",
        "turn the wrist clockwise",
        "roll the gripper clockwise",
    ),
    "roll_ccw": (
        "rotate the wrist counterclockwise",
        "turn the wrist counterclockwise",
        "roll the gripper counterclockwise",
    ),
    "gripper_open": (
        "open the gripper",
        "open the jaw",
        "release the gripper",
    ),
    "gripper_close": (
        "close the gripper",
        "close the jaw",
        "squeeze the gripper",
    ),
    "lift": (
        "lift the gripper",
        "lift the gripper up",
        "raise the gripper",
    )
}

```

References

- [1] A. Brohan et al., “RT-2: Vision-language-action models transfer web knowledge to robotic control,” in *Proceedings of the 7th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 229, 2023, pp. 2165–2183.
- [2] Octo Model Team et al., “Octo: An open-source generalist robot policy,” in *Proceedings of Robotics: Science and Systems*, 2024.
- [3] M. J. Kim et al., “OpenVLA: An open-source vision-language-action model,” in *Proceedings of the 8th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 270, 2025, pp. 2679–2713.
- [4] Physical Intelligence et al., “ $\pi_{0.7}$: A steerable generalist robotic foundation model with emergent capabilities,” 2026. arXiv: 2604.15483.
- [5] NVIDIA. “NVIDIA Isaac GR00T N1.7: A foundation model for generalist robots,” Accessed: Aug. 22, 2026. [Online]. Available: <https://github.com/NVIDIA/Isaac-GR00T>.
- [6] J. Zheng et al., “X-VLA: Soft-prompted transformer as scalable cross-embodiment vision-language-action model,” in *The Fourteenth International Conference on Learning Representations*, 2026.
- [7] J. Shi, S. Wang, X. Yu, L. Ji, J. Gong, and X. Qiu, “Learning to move before learning to do: Task-agnostic pretraining for VLAs,” in *Proceedings of the 43rd International Conference on Machine Learning*, 2026.
- [8] D. Jing et al., “Learning action priors for cross-embodiment robot manipulation,” 2026. arXiv: 2606.26095.
- [9] JoyAI-RA Team, “JoyAI-RA 0.5: Scaling robot manipulation learning via dual action alignment,” 2026. arXiv: 2608.05674.
- [10] S. Wang et al., “In-context world modeling for robotic control,” 2026. arXiv: 2606.26025.
- [11] A. Mandlekar et al., “MimicGen: A data generation system for scalable robot learning using human demonstrations,” in *Proceedings of the 7th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 229, 2023, pp. 1820–1864.
- [12] J. Jang et al., “DreamGen: Unlocking generalization in robot learning through video world models,” in *Proceedings of the 9th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 305, 2025, pp. 5170–5194.
- [13] M. Torne et al., “Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation,” in *Proceedings of Robotics: Science and Systems*, 2024.
- [14] A. Maddukuri et al., “Sim-and-real co-training: A simple recipe for vision-based robotic manipulation,” in *Proceedings of Robotics: Science and Systems*, 2025.
- [15] K. Black et al., “ π_0 : A vision-language-action flow model for general robot control,” in *Proceedings of Robotics: Science and Systems*, 2025.
- [16] J. Bjorck et al., “GR00T N1: An open foundation model for generalist humanoid robots,” 2025. arXiv: 2503.14734.

- [17] K. Black et al., “ $\pi_{0.5}$: A vision-language-action model with open-world generalization,” in *Proceedings of the 9th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 305, 2025, pp. 17–40.
- [18] Open X-Embodiment Collaboration, “Open X-Embodiment: Robotic learning datasets and RT-X models,” in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, May 2024, pp. 6892–6903.
- [19] K. Bousmalis et al., “RoboCat: A self-improving generalist agent for robotic manipulation,” *Transactions on Machine Learning Research*, 2024.
- [20] R. Doshi, H. Walke, O. Mees, S. Dasari, and S. Levine, “Scaling cross-embodied learning: One policy for manipulation, navigation, locomotion and aviation,” in *Proceedings of the 8th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 270, 2025, pp. 496–512.
- [21] L. Wang, X. Chen, J. Zhao, and K. He, “Scaling proprioceptive-visual learning with heterogeneous pre-trained transformers,” in *Advances in Neural Information Processing Systems*, vol. 37, 2024.
- [22] D. Qu et al., “SpatialVLA: Exploring spatial representations for visual-language-action model,” in *Proceedings of Robotics: Science and Systems*, 2025.
- [23] M. J. Kim, C. Finn, and P. Liang, “Fine-tuning vision-language-action models: Optimizing speed and success,” in *Proceedings of Robotics: Science and Systems*, 2025.
- [24] NVIDIA. “NVIDIA Isaac Sim documentation,” Accessed: Sep. 1, 2026. [Online]. Available: <https://docs.isaacsim.omniverse.nvidia.com/latest/index.html>.
- [25] B. Liu et al., “LIBERO: Benchmarking knowledge transfer for lifelong robot learning,” in *Advances in Neural Information Processing Systems*, vol. 36, 2023, pp. 44 776–44 791.
- [26] TheRobotStudio. “SO-101 robot: URDF and MuJoCo description,” Accessed: Aug. 23, 2026. [Online]. Available: <https://github.com/TheRobotStudio/SO-ARM100/tree/main/Simulation/S0101>.
- [27] R. Cadene et al., “LeRobot: An open-source library for end-to-end robot learning,” in *The Fourteenth International Conference on Learning Representations*, 2026.
- [28] Qwen Team, “Qwen-VLA: Unifying vision-language-action modeling across tasks, environments, and robot embodiments,” 2026. arXiv: 2605.30280.
- [29] M. Shukor et al., “SmolVLA: A vision-language-action model for affordable and efficient robotics,” 2025. arXiv: 2506.01844.