Beyond Data Scaling: Representation-Centric Continued Pre-training for Vision-Language-Action Models

Senqiao Yang[1†] **Chengyao Wang**[1†] **Yuxin Chen**[2] **Zixuan Wang**[2] **Longxiang Tang**[2]
Haokun Gui[2] **Jinhui Ye**[2] **Changsheng Lu**[2] **Xiaoyang Wu**[3] **Mingkang Zhu**[1]
Pengguang Chen[1] **Shu Liu**[1‡] **Zhuotao Tian**[4] **Hengshuang Zhao**[3] **Bei Yu**[1] **Jiaya Jia**[1,2]

[1]The Chinese University of Hong Kong [2]The Hong Kong University of Science and Technology
[3]The University of Hong Kong [4]Shenzhen Loop Area Institute
[†]Project leaders [‡]Corresponding author
yangsenqiao.ai@gmail.com

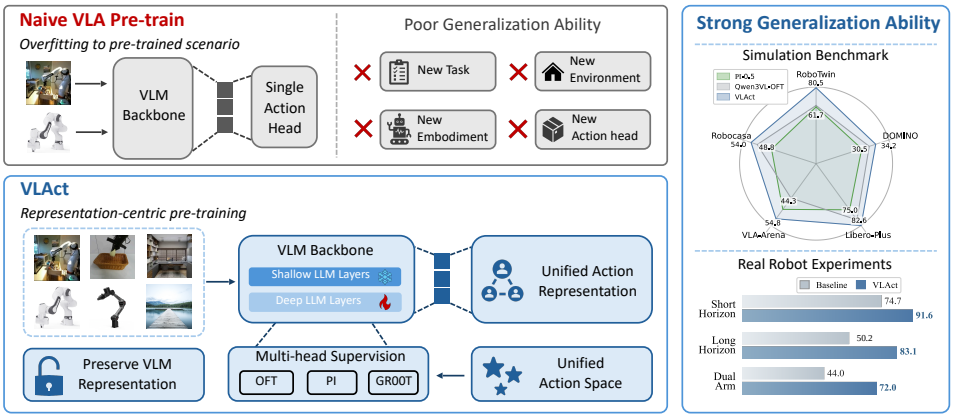


Figure 1: **Overview of VLAct.** VLAct uses representation-centric continued pre-training to learn generalizable action-aware representations across tasks, embodiments, environments, and action heads.

Abstract

Scaling robot data is crucial for building generalist Vision-Language-Action (VLA) models, yet robot trajectories are fundamentally harder to scale than web-scale image-text data because they require embodied collection and sparsely cover the physical world. This makes representation quality a central bottleneck: under a fixed robot-data budget, VLA continued pre-training must convert limited trajectories into transferable visual-action knowledge, rather than merely fit actions. We propose **VLAct**, a VLA-oriented VLM backbone trained with a representation-centric continued pre-training recipe, which starts from a pretrained VLM and trains on broad, heterogeneous, multi-embodiment robot data before downstream task-specific fine-tuning. VLAct preserves the broad VLM prior, avoids over-specializing the backbone to a single action head, and encourages shared action semantics across embodiments through VLM-prior preservation, multi-head continuous action co-supervision, and a partially unified cross-embodiment action layout, while leaving downstream users free to attach task-specific action heads during fine-tuning. Across multi-embodiment simulation benchmarks, real-world robot experiments, and unseen-embodiment transfer, VLAct consistently improves downstream performance under fixed fine-tuning protocols. On LIBERO-Plus and RoboTwin 2.0, VLAct surpasses large-scale industrial VLA systems such as ABot-M0 and LingBot-VLA, achieving 82.6% and 92.5% success, respectively. On

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

the recently published RoboDojo simulation benchmark, VLAct achieves a 10.66 average score and 7.60% success rate, ranking sixth among all policies by success rate. It outperforms all explicitly designated world-action-model (WAM) entries on both metrics, as well as several industry-developed systems. Most notably, on RoboCasa-GR1, a humanoid embodiment never seen during continued pre-training, **VLAct with only 20% of downstream trajectories already outperforms the full-data GR00T-N1.6 baseline.** These results are obtained using fully open-source data and only a **16-GPU** training setup, showing that representation-centric continued pre-training can deliver highly competitive performance under a modest compute budget and is an important independent axis of VLA progress beyond data scaling. All models and training pipelines are open-sourced.

1 Introduction

Scaling data, model size, and training compute has been a central driver behind modern foundation models [1, 82]. In language and vision, this scaling paradigm succeeds not simply because web-scale corpora are large [66, 28, 3], but because they provide broad coverage of visual and semantic variation. Vision-Language-Action (VLA) models naturally aspire to the same paradigm: more robot trajectories should, in principle, support more generalist robot policies [12, 115, 11, 54, 103]. Yet robot learning faces a qualitatively different scaling bottleneck.

Robot trajectories cannot be scraped from the web. They must be produced through embodied execution in the physical world, often under human teleoperation or carefully designed data-collection protocols[75, 52, 94, 55]. More importantly, the space that robot policies must generalize over is combinatorial and continuous, spanning scenes, objects, task goals, embodiments, and contact-rich dynamics. As a result, even very large robot datasets remain sparse samples of the physical interaction space, with partial and uneven coverage across the situations a robot may encounter.

This does not diminish the value of scale, but it changes the role scale should play. If VLA continued pre-training cannot count on exhaustive coverage [75, 115], then its success depends not only on how many trajectories are collected, but on how effectively those trajectories induce transferable visual-action representations. We therefore look *beyond data scaling*: not to replace scale, but to ask what must accompany it. ***Given a fixed robot-data budget, how can VLA continued pre-training learn strong representations that generalize beyond the trajectories on which they were trained?***

This paper takes a representation-centric view of VLA continued pre-training. Rather than treating robot continued pre-training merely as large-scale action fitting, we view it as a process of distilling robot trajectories into reusable visual-action knowledge within the backbone. A useful VLA backbone should not only predict the actions observed in the pre-training data, but also internalize transferable priors about objects, affordances, spatial relations, and action-conditioned physical interactions that downstream policies can reuse across new tasks, scenes, and embodiments [41, 33]. Effective VLA continued pre-training therefore requires designing how trajectory data shapes the backbone representation, rather than simply optimizing action prediction on the pre-training distribution.

To build a transferable VLA backbone, we first examine how robot continued pre-training shapes the learned representation, rather than only its action-prediction accuracy. We identify three failure modes in naive VLA continued pre-training. First, because robot trajectories are much narrower than the web-scale corpora used to train VLMs, full end-to-end updating can erode broadly useful vision-language features. Second, supervision from a single action head can over-specialize the backbone to that head's decoding geometry, limiting transfer to alternative heads. Third, embodiment-specific output spaces can weaken cross-embodiment sharing by representing physically comparable actions, such as gripper opening and closing, through isolated robot-specific heads.

Motivated by these observations, we propose **VLAct**, a VLA-oriented VLM backbone trained with a representation-centric continued pre-training recipe. VLAct preserves the VLM prior through shallow-layer protection and caption-data mixing, prevents single-head over-specialization through multi-head continuous action co-supervision, and encourages shared action semantics through a partially unified cross-embodiment action layout with a wrap-aware loss for periodic joints. These components are used only during continued pre-training to shape the backbone representation, while downstream users remain free to attach the action head best suited to their task and fine-tune under their own data budget.

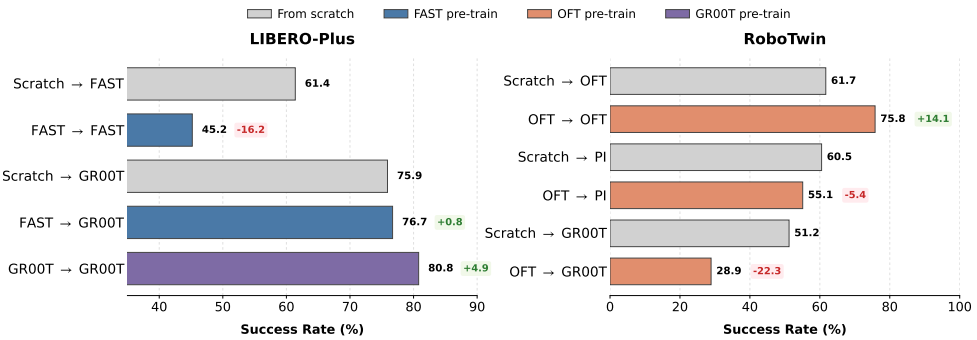


Figure 2: **Action supervision reshapes VLA backbone representations.** Keeping the VLM backbone fixed, we vary the action head used during pre-training and fine-tuning. Discrete FAST supervision can transfer to a continuous GR00T head, but retaining the FAST head suffers from discretization-induced information loss. Continuous OFT supervision improves same-head fine-tuning, but transfers poorly to other continuous heads, suggesting head-specific representation collapse rather than a generally reusable action representation.

What "continued pre-training" means here. We start from an *already pretrained* VLM and train it on broad, multi-embodiment robot trajectories before any downstream task-specific fine-tuning, rather than pre-training a foundation model from scratch. Representative generalist VLA systems, including π_0 [12], $\pi_{0.5}$ [45], and GR00T N1/N1.5 [11], adopt exactly this setting and commonly call it *VLA pre-training*; we adopt the more precise term here, and keep "pre-training" as shorthand where the stage is clear from context.

We evaluate **VLAct** across multi-embodiment simulation benchmarks, real-world robot experiments, and unseen-embodiment transfer. Under a fixed downstream protocol, it consistently improves over generic VLM backbones and naive VLA-pretrained baselines. VLAct outperforms strong public and industrial VLA systems across benchmarks: 82.6% on LIBERO-Plus [36], 54.8% on VLA-Arena [111], and 92.5% on RoboTwin 2.0 [24]. On recent RoboDojo [23], it achieves a 10.66 score and 7.60% success rate, ranking sixth by success and outperforming all explicitly designated WAM models. With only 20% of RoboCasa-GR1 data, it also surpasses the full-data GR00T-N1.6 baseline [11]. All these results are obtained using fully open-source data and only a **16-GPU** training setup.

Furthermore, across every VLAct results comparison, **the VLM backbone weights are the only thing that changes**: the action head and its fresh initialization, the downstream data, the optimizer, and the fine-tuning budget are all identical. The resulting **7.6–21.4** point gains are therefore attributable to the backbone alone. Hence, we take this as evidence that the VLM should be treated not as a fixed component inherited from general vision-language pre-training, but as a first-order design variable for VLA. We hope this encourages the community to study **how foundation VLMs can be made better suited to physical tasks and serve as stronger foundations for VLA models.**

2 Pilot Study: How Action Supervision Shapes VLA Backbones

A central goal of VLA continued pre-training is to produce a reusable backbone, rather than a policy initialization tied to one particular action head. In practice, there is no universally optimal action head: discrete token heads (FAST), regression heads (OFT), flow-matching heads (PI), and diffusion-style continuous heads (GR00T) impose different inductive biases and may be preferred under different tasks, embodiments, horizons, or deployment constraints. A useful pretrained backbone should therefore expose action-relevant information in a form that downstream heads can reuse, instead of encoding it in a representation specialized to the head used during pre-training.

As shown in Fig. 2, we isolate this action-head axis through controlled pilot experiments on LIBERO-Plus and the RoboTwin-Clean setting. Fixing the VLM backbone to Qwen3-VL-4B, we vary the action head used during pre-training and fine-tuning, and test how different heads reshape the resulting backbone. We use representative discrete and continuous heads, and the head definitions are shown in Appendix I.1.

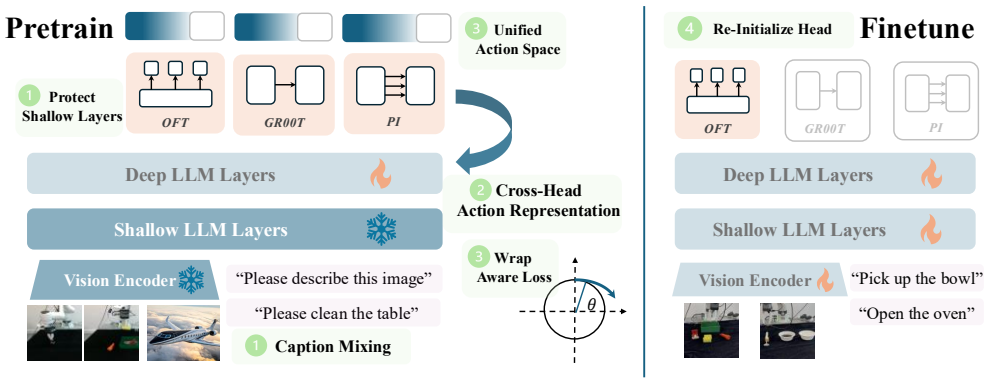


Figure 3: **Overview of VLAct.** During pre-training, ① VLAct preserves the VLM prior through shallow-layer protection and caption mixing (§ 3.2), ② uses multi-head continuous supervision to avoid head-specific representation collapse (§ 3.3), and ③ aligns cross-embodiment action semantics with a unified action layout and wrap-aware loss (§ 3.4). During fine-tuning, ④ we reuse the pretrained backbone and freshly initialize the task-specific action head (§ 3.1).

Discrete supervision can transfer, but suffers from discretization-induced information loss. The LIBERO-Plus probe asks whether discrete FAST-token pre-training provides sufficient action information for downstream control. FAST supervision can inject transferable structure into the backbone: when the FAST-pretrained backbone is paired with a continuous GR00T head, performance slightly improves over from-scratch GR00T fine-tuning. However, retaining the FAST Action Head performs much worse than continuous-head fine-tuning, and FAST pre-training does not close this gap. This suggests that discrete action tokens teach coarse action structure, but lose fine-grained temporal and amplitude information that matters for manipulation.

Single-head continuous supervision can induce head-specific representation collapse. The RoboTwin-Clean probe shows a different failure mode. OFT pre-training substantially improves downstream performance when the fine-tuning head is also OFT, confirming that continuous action supervision can inject fine-grained action information into the backbone. However, this gain does not transfer to other continuous heads: the same OFT-pretrained backbone performs poorly when paired with PI or GR00T. This suggests that the learned representation is not simply more action-aware in a head-agnostic sense. Instead, single-head pre-training can collapse the backbone feature geometry toward the directions that are most useful for the pre-training head. Action-relevant information may remain present, but it is organized in a head-specific form that other downstream heads cannot easily decode. Strong same-head performance therefore overstates backbone reusability.

Analysis. Action supervision is not neutral: the pre-training head directly shapes the backbone representation. Discrete supervision can transfer across heads, but loses fine-grained action information through discretization. Continuous supervision avoids this bottleneck, but single-head pre-training can cause head-specific representation collapse, organizing action features around one head geometry rather than keeping them broadly accessible. Thus, a transferable VLA backbone should preserve fine-grained action information while supporting multiple downstream heads. This motivates VLAct's multi-head continuous co-supervision in Sec. 3.3.

3 VLAct: Representation-Centric VLA Continued Pre-training

3.1 Overall Continued Pre-training and Fine-tuning Recipe

As shown in Fig. 3, VLAct performs continued pre-training of a reusable VLA backbone from a VLM initialization. During continued pre-training, it preserves the VLM prior by freezing the vision encoder and shallow LLM layers, while mixing robot trajectories with caption data. Robot samples supervise the shared backbone through multiple continuous action heads, including OFT, PI, and GR00T, under a partially unified action layout with masked inactive dimensions and wrap-aware losses.

These continued pre-training components are used only to shape the backbone. During fine-tuning, we discard the pre-training heads and caption stream, attach a freshly initialized task-specific action

head, and train under the same downstream setting as each baseline. Thus, improvements come from the learned backbone representation rather than from reusing a pre-adapted action head.

3.2 Preserving the VLM's Vision-Language Representation

Motivation. A VLA backbone starts from a strong vision-language model, such as Qwen3-VL, whose visual and language representations are learned from broad web-scale data. Naive action-driven continued pre-training can overwrite this prior: robotics trajectories are expensive, narrow in visual diversity, and concentrated on a limited set of manipulation scenes. As a result, end-to-end VLA continued pre-training may improve action prediction on the pre-training distribution while degrading the general vision-language features that make the backbone transferable. Our first design goal is therefore to inject action knowledge without unnecessarily drifting away from the original VLM representation. We use two simple mechanisms: shallow-layer protection and caption-mixed pre-training.

Shallow-layer protection. We freeze the entire vision encoder and the lower half of the LLM layers during VLA pre-training, while updating the upper LLM layers and the action heads. The motivation is to protect the parts of the model most responsible for low-level visual processing and early vision-language alignment, while still allowing the upper layers to adapt to action-conditioned reasoning. During downstream fine-tuning, we unfreeze the full model so that all parameters can adapt to the target task. This simple freezing strategy improves downstream performance by 3.7% on LIBERO-Plus and 3.4% on Agilex. Detailed ablations are provided in Appendix C.

Caption-mixed pre-training. Mixing VLM data during VLA pre-training is a common strategy for mitigating representation drift, but not all VLM supervision is equally useful. We ablate several VLM data mixtures and find that captioning data provides the strongest anchor for downstream performance. The reason is intuitive: captions offer dense supervision over objects, attributes, spatial relations, and scene context, helping preserve the backbone's original vision-language representation while it learns from robot trajectories. We therefore use caption-mixed pre-training in VLAct. Detailed ablations are provided in Appendix D.

Takeaway 1. *VLAct preserves the VLM prior while injecting action knowledge during continued pre-training: shallow-layer protection limits representation drift, and caption mixing anchors the trainable layers with dense vision-language supervision.*

3.3 Aligning and Diversifying the Action Representation

Design principle. The pilot study in Sec. 2 suggests that an effective VLA backbone should satisfy two requirements. First, it should support *head transfer*: downstream users should be able to attach action heads suited to their target embodiments and deployment scenarios, without inheriting a representation specialized to one particular head geometry. Second, it should learn high-quality action features, rather than merely fitting the pre-training head. Naive single-head pre-training can violate these requirements. Since the backbone and the attached head are optimized jointly, the backbone may encode action information in a form tailored to that head. This head-specific feature specialization reduces transferability across heads, even if the resulting representation performs well when the same head is reused.

Co-supervised multi-head pre-training. We address this by pre-training the same backbone with multiple action heads in parallel. Concretely, we attach three representative continuous heads, *OFT*, *PI*, and *GR00T*, to the shared VLA backbone. Given the same vision-language input, the backbone produces a shared latent representation z. Each head receives the same z and predicts the same ground-truth action chunk a. The training objective is

$$\mathcal{L}_{\text{action}} = \mathcal{L}_{\text{OFT}} + \mathcal{L}_{\text{PI}} + \mathcal{L}_{\text{GR00T}}.$$

This objective prevents the representation from becoming tied to a single decoder: the backbone must expose action information in a form that can be used by several action parameterizations. Since all heads share the same backbone forward pass, multi-head supervision adds only lightweight head-specific computation rather than repeated backbone computation.

This design is deliberately simple: we do not introduce a new action head or a specialized alignment module. Instead, we use head diversity itself as supervision. Because the three heads impose different objectives and decoder biases on the same action-prediction problem, the backbone cannot rely on

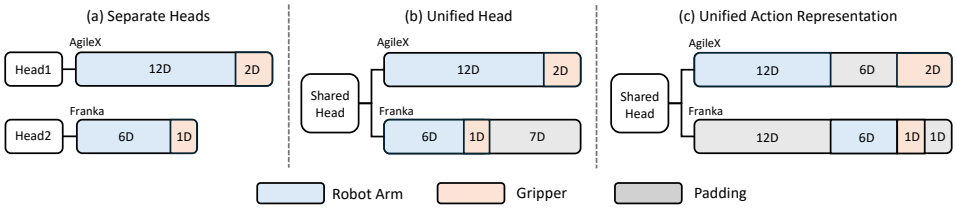


Figure 4: **Unified action space design.** We compare separate embodiment-specific heads, a naively unified action space, and our partially unified action space. VLAct shares physically aligned dimensions, such as gripper open/close, while masking unavailable or embodiment-specific dimensions.

features that are useful only to one head. To reduce all losses simultaneously, it must encode action information in a form that remains accessible across multiple downstream parameterizations.

Co-supervision therefore plays two roles. It makes the backbone more head-agnostic, improving transfer to alternative downstream heads; and it regularizes the representation, yielding stronger action features even when the downstream head matches one of the pre-training heads. As shown in Appendix E, this second effect is empirically important: multi-head pre-training not only improves head transfer, but can also outperform single-head pre-training under the same downstream head.

Takeaway 2. *Multi-head co-supervision turns head diversity into a representation-learning signal. By training OFT, PI, and GR00T on the same backbone latent and action targets, VLAct forces action information to remain accessible across different head geometries rather than collapsing toward one of them. This produces a backbone that transfers better across downstream heads and learns stronger action features than single-head pre-training.*

3.4 Unifying the Action Representation across Embodiments

Motivation. The previous Sec. 3.3 addresses variation across action heads. We now consider variation across robot embodiments. Different embodiments have different action spaces, and a common solution is to attach a separate action head or output projector to each robot type. This is flexible, but it also hides any shared action structure inside embodiment-specific heads. Our motivation is simple: when two embodiments share a physically meaningful action dimension, the supervision for that dimension should also be shared; when they do not, we should not force an artificial alignment.

Partially unified cross-embodiment action space. To share action supervision across embodiments without forcing incompatible degrees of freedom into the same coordinates, VLAct uses a partially unified cross-embodiment action space. Fig. 4 compares three natural designs. The first uses embodiment-specific heads, which isolate supervision across robots. The second uses a fully unified action space, where lower-dimensional robots are padded to a common action dimension; this can naively align coordinates with different physical meanings. VLAct adopts the third design: one shared action head with an action space that is unified only along physically comparable dimensions.

Concretely, gripper dimensions are shared across embodiments because open/close commands have comparable semantics, while arm dimensions remain embodiment-specific when the robots have different kinematics or degrees of freedom. During training, each sample contributes loss only on the active dimensions for its embodiment, and inactive dimensions are masked out. This keeps the architecture minimal: no embodiment adapter, routing module, or embodiment-conditioned decoder is introduced. Instead, cross-embodiment sharing is induced directly through shared coordinates in the action space. As shown in Appendix I.2, this partial unification improves downstream performance by sharing supervision where action semantics are physically aligned while avoiding naive full sharing across incompatible joints.

Wrap-aware loss for periodic joint angles. We also handle a small but important parameterization issue for absolute joint angles. Standard regression treats 179° and -179° as 358° apart, although they are physically only 2° apart. We therefore apply a wrap-aware loss on periodic joint-angle dimensions, measuring angular residuals modulo 360°. This correction is only used for absolute joint dimensions; the full definition and dimension scope are provided in Appendix I.2.

Table 1: **Results on LIBERO-Plus.** Per-dimension success rate (%) on the LIBERO-Plus robustness benchmark. All models are trained on standard LIBERO and evaluated on the perturbed test set. Dark blue highlights VLAct and per-column best results; light blue highlights the strongest prior method by total score and per-column second-best results.

Method	Camera	Robot	Lang.	Light	Bg.	Noise	Layout	Total
OpenVLA [54]	0.8	3.5	23.0	8.1	34.8	15.2	28.5	15.6
OpenVLA-OFT [53]	56.4	31.9	79.5	88.7	93.3	75.8	74.2	69.6
NORA [44]	2.2	37.0	65.1	45.7	58.6	12.8	62.1	39.0
WorldVLA [19]	0.1	27.9	41.6	43.7	17.1	10.9	38.0	25.0
UniVLA [90]	1.8	46.2	69.6	69.0	81.0	21.2	31.9	42.9
π_0 [12]	13.8	6.0	58.8	85.0	81.4	79.0	68.8	53.6
π_0-FAST [76]	65.1	21.6	61.0	73.2	73.2	74.4	68.8	61.6
RIPT-VLA [81]	55.2	31.2	77.5	88.3	91.6	73.5	74.2	68.4
Abot-M0 [103]	60.4	67.9	86.4	96.2	91.6	**86.4**	82.6	80.5
Qwen3VL-OFT	47.0	60.1	**87.0**	96.3	95.3	73.1	79.2	75.0
VLAct	**73.9**	**68.4**	81.5	**96.7**	**96.7**	86.0	**83.3**	**82.6**

Takeaway 3. *Cross-embodiment continued pre-training benefits from sharing supervision where semantics are physically aligned. VLAct uses one shared head with a partially unified action space: gripper dimensions are shared across embodiments, while incompatible arm dimensions remain embodiment-specific and masked when inactive. This avoids isolated embodiment-specific heads and naive full sharing, improving cross-embodiment representations and downstream performance.*

4 Experiments

VLAct is pre-trained on fully open-source robot datasets, including DROID [52], InternA1 [86], RoboCoin [94], and MolmoAct [55], together with captioning data for VLM representation preservation. Data processing and cleaning details are provided in Appendix H. We build on the StarVLA training codebase [27] and use Qwen3-VL-4B as the base vision-language backbone. All experiments are conducted on 16 GPUs. We will release the training scripts and model checkpoints.

4.1 Effectiveness on Single-arm Franka Manipulation

Evaluation benchmarks. We evaluate single-arm Franka manipulation mainly on *LIBERO-Plus* [36], a recent benchmark designed to probe robustness beyond the saturated standard LIBERO setting. LIBERO-Plus introduces systematic test-time perturbations, including changes in camera views, robot states, visual noise, object layouts, and task instructions, making it suitable for evaluating transferable VLA continued pre-training. We compare VLAct with representative VLA systems, including π_0 [12], $\pi_{0.5}$ [45], and Abot-M0 [103]. Baseline numbers are taken from LIBERO-Plus [36]. Additional Franka results, VLA-Arena, are reported in Appendix B.1.

LIBERO-Plus results. Table 1 shows that our VLAct achieves the best overall score, reaching **82.6%**, and improves over our strongest in-house baseline, Qwen3VL-OFT, by **7.6%** (82.6% *vs.* 75.0%). This comparison isolates the effect of the continued pre-training recipe, since both models use the same Qwen3-VL backbone family and an OFT downstream head. VLAct also surpasses Abot-M0, a large-scale VLA system from Alibaba, by 2.1%, using fully open-source data and substantially fewer training resources. The largest gains over Qwen3VL-OFT appear on *Camera*, *Robot*, *Noise*, and *Layout*, suggesting that representation-centric continued pre-training yields more robust visual-spatial representations than standard Qwen3-VL fine-tuning on LIBERO.

4.2 Effectiveness on Bimanual AgileX Manipulation

Evaluation benchmarks. To evaluate bimanual AgileX manipulation, we use two recent benchmarks with complementary generalization axes: *RoboTwin 2.0* [24] and *DOMINO* [34]. RoboTwin 2.0 evaluates robust dual-arm manipulation under clean scenes and randomized visual conditions. We report two settings: the *Base* setting uses only 50 clean trajectories per task, while the *Data Scaling*

Table 2: **RoboTwin 2.0 results.** Task success rate (%) under the Base and Data Scaling settings. Base uses 50 clean trajectories per task and is evaluated on the Clean regime; Data Scaling additionally uses 500 randomized trajectories per task and is evaluated on both the Clean and Random regimes. Each reported regime evaluates 50 tasks with 100 episodes per task. Published numbers are taken from the cited papers or their same-protocol re-evaluations. Dark blue highlights our default VLAct-OFT configuration and light blue highlights alternative VLAct heads. Best results are bold and second-best results are underlined.

Method	**Family**	**Base Setting**	**Data Scaling Setting**	
		Clean	**Clean**	**Random**
Published VLA and world-action-model systems				
π_0 [12]	Flow	46.4	65.9	58.4
$\pi_{0.5}$ [45, 95]	Flow	60.2	82.7	76.8
X-VLA [114, 10]	Flow	70.0	72.8	72.8
Lingbot-VLA [95]	Flow	–	88.6	86.7
Abot-M0 [103]	AML	–	86.1	85.1
InternVLA-A1 [46]	Flow	–	89.4	89.6
Being-H0.7 [4]	Flow	–	90.2	89.6
Motus [10]	WAM	–	88.7	87.0
Fast-WAM [109]	WAM	–	91.9	<u>91.8</u>
HoloBrain-0-QW [64]	Diff.	–	91.9	**92.3**
In-house baseline				
Qwen3VL-OFT	OFT	61.7	88.2	88.3
VLAct (same backbone, varying continuous head)				
VLAct	OFT	**80.5**	<u>92.5</u>	90.8
VLAct	GR00T	76.0	89.6	87.4
VLAct	PI	<u>77.0</u>	**93.0**	88.8

setting further adds 500 domain-randomized expert trajectories per task. The latter are successful expert demonstrations released by the RoboTwin 2.0 team under its official `demo_randomized` setting, in which background, clutter, tabletop height, lighting, and other scene factors are varied; they are *not* random-action rollouts. Unless otherwise noted in the cited paper, Data Scaling comparisons follow this multi-task protocol with 2,500 clean and 25,000 randomized demonstrations. Training compute, optimization schedules, and checkpoint-selection procedures may nevertheless differ across methods. This allows us to evaluate both performance under limited clean trajectories and whether the pre-trained model can continuously benefit from more downstream robot trajectories. Moreover, DOMINO evaluates dynamic manipulation with moving objects and changing environments, targeting the spatiotemporal reasoning abilities often missed by static manipulation benchmarks. Additional results are shown in Appendix B.2.

RoboTwin 2.0 results. Table 2 shows that VLAct establishes the strongest results among the compared methods in the *Base* setting and remains competitive with recent frontier VLA and world-action-model systems in the *Data Scaling* setting. In the Base setting, VLAct-OFT reaches 80.5% success with only 50 clean trajectories per task, substantially improving over the Qwen3VL-OFT baseline (61.7%) and outperforming the compared published methods. This demonstrates that VLAct enables strong sample-efficient downstream adaptation from limited task-specific data. In the *Data Scaling* setting, VLAct-OFT reaches 92.5% on Clean and 90.8% on Random, outperforming large-scale systems including InternVLA-A1, Being-H0.7, Motus, LingBot-VLA, ABot-M0, and $\pi_{0.5}$. It also remains close to HoloBrain-0 and Fast-WAM. These comparisons position VLAct among the strongest reported RoboTwin 2.0 policies while avoiding a claim of absolute state of the art. VLAct remains strong as more downstream data is added, showing that the learned representation is not merely specialized to a low-data regime. Across downstream head choices, performance remains stable, supporting the head-transfer claim in Sec. 3.3: the transferred object is the backbone representation, rather than a particular action head. Indeed, among the VLAct variants, the PI head attains the highest Clean success (93.0%), slightly above the OFT head (92.5%) that we use as the default configuration elsewhere in the paper; all three heads remain within 3.4 points of each other.

4.3 Real-world Experiments

In what follows, we perform the real-world robot experiments using Franka Research 3 7-DoF arms to validate the effectiveness of our VLAct. We evaluate the proposed VLAct and a competitive baseline (Qwen3VL-4B-OFT without pre-training) across four regimes: **(1)** *In-Domain (ID) single-arm short-horizon tasks*, such as 'Carrot-from-pot', 'Button pressing', 'Cube stacking', 'Pen in cup'; **(2)** *ID single-arm long-horizon tasks*, *e.g.*, 'Table cleaning' & 'Scoop beans'; **(3)** *ID dual-arm coordination*, with tasks 'Unplugging', 'Breakfast preparation', 'Banana handover-place', 'Fold

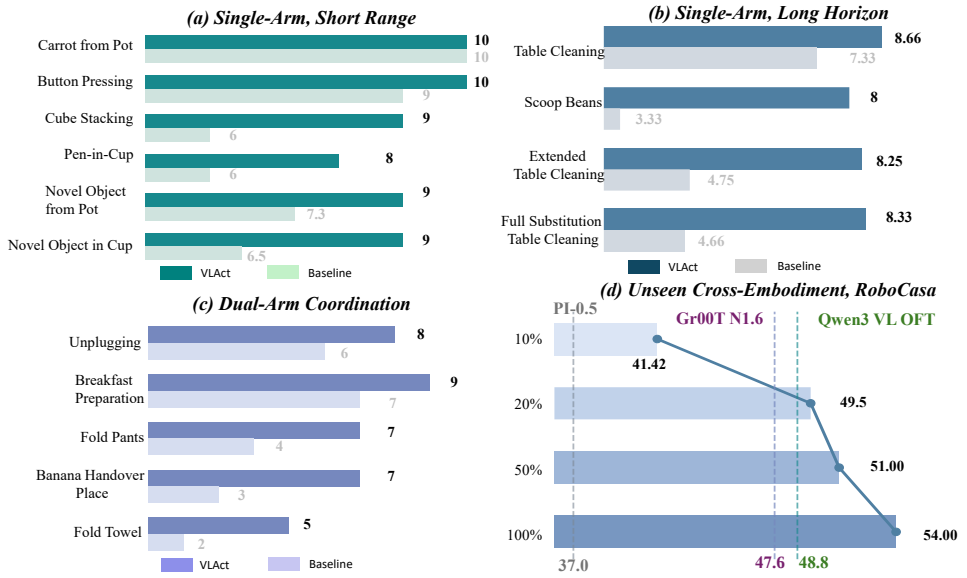


Figure 5: **Real-machine evaluation and cross-embodiment transfer.** (a)–(c) Real-machine evaluation results on single-arm short-range, single-arm long-horizon, and dual-arm coordination tasks, where VLAct consistently improves over the baseline across diverse manipulation settings. (d) Cross-embodiment transfer on RoboCasa-GR1, where VLAct transfers to the held-out GR-1 embodiment and surpasses reported full-data baselines with only 20% downstream data.

pants', 'Fold towel'; and **(4)** *Out-of-Domain (OOD) generalization*, including novel objects from pot, novel objects in cup, and table cleaning with extended or fully substituted object types.

We train two separate models for their respective evaluations: one for all single-arm tasks while another for all dual-arm tasks. Each model is trained with 50k training steps on 8 H800 GPUs, and evaluated over 10 trials to calculate the success rate. Detailed hardware settings, prompts, visualizations of each task, and success rate calculation criterion are given in Appendix J.

Results. As shown in Fig. 5, VLAct consistently improves real-world performance over the non-pre-trained Qwen3VL-4B-OFT baseline. In single-arm short-horizon ID tasks (Fig. 5 (a)), VLAct achieves 92.5% average success, higher than the baseline's 77.5%, with clear gains on spatially sensitive tasks such as cube stacking and pen-in-cup. VLAct also generalizes better to novel objects: on the short-horizon OOD tasks, it obtains 90.0% success on both 'novel object from pot' and 'novel object in cup', compared with 73.3% and 65.0% for the baseline. In single-arm long-horizon tasks (Fig. 5 (b)), VLAct achieves weighted scores of 86.6% on table cleaning and 80.0% on scooping beans, outperforming the baseline scores of 73.3% and 33.3%. Under harder OOD long-horizon settings, VLAct keeps high scores of 82.5% for extended-sequence table cleaning and 83.3% for full object substitution, while the baseline drops to 47.5% and 46.6%. Finally, despite being pre-trained only on single-arm data, VLAct transfers effectively to dual-arm coordination, reaching 72.0% average success across dual-arm tasks versus 44.0% for the baseline, with the largest gains on deformable-object and synchronization-heavy tasks such as pants folding and towel folding (Fig. 5 (c)). These results suggest that VLAct improves not only in-domain manipulation accuracy, but also novel-object generalization, long-horizon execution, and dual-arm coordination.

4.4 Does VLAct Transfer to Unseen Embodiments?

Setup. We test whether VLAct learns action representations that transfer to embodiments unseen during continued pre-training. VLAct is continually pre-trained only on Franka single-arm and AgileX bimanual data; both the GR-1 humanoid and the ARX X5 bimanual platform are held out during continued pre-training. We fine-tune VLAct-OFT on RoboCasa-GR1 under a fixed downstream protocol and additionally evaluate VLAct through the official RoboDojo simulation benchmark on ARX X5. These settings directly probe cross-embodiment transfer because the target robots differ from the pre-training embodiments in morphology, action space, and manipulation dynamics.

RoboCasa-GR1 results. Fig. 5(d) reports RoboCasa-GR1 performance under different fractions of the downstream training data. With full fine-tuning data, VLAct reaches **54.0%**, outperforming the strongest reported full-data baselines, including Qwen3VL-OFT at 48.8%, GR00T-N1.6 at 47.6%, and $\pi_{0.5}$ at 37.0%. Since GR-1 trajectories are never used in VLAct pre-training, this shows that the backbone learned from Franka and AgileX provides a stronger initialization for a new humanoid.

The data-efficiency curve further supports this conclusion. With only 20% of the RoboCasa-GR1 trajectories, VLAct already reaches 49.5%, matching or exceeding the full-data Qwen3VL-OFT baseline. With 50% of the data, performance rises to 51.0%, and full-data fine-tuning further improves it to 54.0%. Thus, VLAct is not merely benefiting from downstream data scale; the pretrained backbone transfers useful action structure to the unseen embodiment before full task-specific adaptation. These results support the central claim that VLAct learns reusable action representations rather than overfitting to the robot embodiments observed during pre-training.

RoboDojo results. RoboDojo [23] evaluates policies on 42 ARX X5 simulation tasks spanning Generalization, Precision, Long-Horizon, Memory, and Open capabilities. Each task is evaluated over 50 episodes, and the benchmark reports both a partial-progress score and binary task success. Table 5 in Appendix B.3 reports the official leaderboard snapshot from August 24, 2026. Among 35 policies, VLAct ranks **eighth by average score** and **sixth by average success rate**, placing it in the top quartile under both metrics. Among the four explicitly designated world-action-model (WAM) entries, X-WAM is the strongest, with a 7.69 score and 3.83% success rate. VLAct exceeds X-WAM by **2.97 score points** and **3.77 percentage points** in success, thereby outperforming every designated WAM entry on both aggregate metrics. VLAct also outperforms several industry-developed systems, including Xiaomi-Robotics-0, GalaxeaVLA (G0), LingBot-VLA [95], and ABot-M0 [103]. Moreover, six of the seven higher-scoring entries are submitted by industry teams or robotics companies: Dexmal, Galaxea AI, Xiaomi Robotics, Tencent Robotics X, OpenHelix Robotics, and Physical Intelligence. Although the leaderboard does not normalize for training compute, this result is notable given that VLAct uses fully open-source data and a 16-GPU continued pre-training setup.

Compared with the Qwen3-VL-based StarVLA-α [105] entry, VLAct improves the average score by 4.26 points and success by 4.36 percentage points. The largest gains occur on Precision and Long-Horizon tasks, while Memory remains a clear limitation. Since ARX X5 is also unseen in pre-training, this provides an additional cross-embodiment test on an external benchmark.

5 Related Work

Generalist Robot Policies. Recent research has shifted toward generalist policies via Vision-Language-Action models [14, 54, 115]. By fine-tuning web-scale vision-language models [5] on diverse robotic datasets [75, 52], VLAs exhibit remarkable generalization across novel instructions and environments. However, frontier models (e.g., the π series [12]) often rely on closed-source data, hindering reproducibility. Our work provides a systematic recipe study using only public data, focusing on the representation learning nuances that determine VLA performance. The additional related work is shown in Appendix A.

6 Conclusion

We presented **VLAct**, a representation-centric VLA continued pre-training recipe for building more transferable VLM backbones from limited robot data. Instead of treating robot continued pre-training as large-scale action fitting, VLAct is designed to preserve the broad VLM prior, avoid over-specialization to a single action head, and encourage shared action semantics across embodiments. Across multi-embodiment simulation benchmarks, real-world robot experiments, and unseen-embodiment transfer, VLAct consistently improves downstream performance under fixed fine-tuning protocols. It surpasses strong VLA baselines on LIBERO-Plus, achieves state-of-the-art performance on RoboTwin 2.0, ranks in the top quartile of the all policy RoboDojo leaderboard while outperforming several industry-developed systems, and rapidly adapts to a humanoid embodiment unseen during continued pre-training on RoboCasa-GR1. These results are obtained using fully open-source data and only a 16-GPU training setup. Their competitiveness against predominantly industry-backed systems shows that VLA progress is not only a matter of scaling robot trajectories or compute: better continued pre-training design can substantially improve what a fixed resource budget delivers.

Acknowledgments and Disclosure of Funding

This work was supported in part by the Research Grants Council under the Areas of Excellence scheme grant AoE/E-601/22-R, the National Natural Science Foundation of China (Nos. 62422606 and 62441615), the Hong Kong Research Grants Council General Research Fund (No. 17213925), the Shenzhen Science and Technology Program (Nos. ZDCY20250901113000001 and KJZD20240903102901003), and the Shenzhen Loop Area Institute Fund (No. FPF10120250006).

The authors declare no competing interests.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Jorge Aldaco, Travis Armstrong, Robert Baruch, Jeff Bingham, Sanky Chan, Kenneth Draper, Debidatta Dwibedi, Chelsea Finn, Pete Florence, Spencer Goodrich, et al. Aloha 2: An enhanced low-cost hardware for bimanual teleoperation. *arXiv preprint arXiv:2405.02292*, 2024.

[3] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[4] BeingBeyond Team. Being-H0.7: A latent world-action model from egocentric videos. *arXiv preprint arXiv:2605.00078*, 2026.

[5] Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, et al. Paligemma: A versatile 3b vlm for transfer. *arXiv preprint arXiv:2407.07726*, 2024.

[6] Homanga Bharadhwaj, Abhinav Gupta, Shubham Tulsiani, and Vikash Kumar. Zero-shot robot manipulation from passive human videos. *arXiv preprint arXiv:2302.02011*, 2023.

[7] Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation. In *European Conference on Computer Vision*, pages 306–324. Springer, 2024.

[8] Homanga Bharadhwaj, Jay Vakil, Mohit Sharma, Abhinav Gupta, Shubham Tulsiani, and Vikash Kumar. Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4788–4795. IEEE, 2024.

[9] Chethan Bhateja, Derek Guo, Dibya Ghosh, Anikait Singh, Manan Tomar, Quan Vuong, Yevgen Chebotar, Sergey Levine, and Aviral Kumar. Robotic offline rl from internet videos via value-function pre-training. *arXiv preprint arXiv:2309.13041*, 2023.

[10] Hongzhe Bi, Hengkai Tan, Shenghao Xie, Zeyuan Wang, Shuhe Huang, Haitian Liu, Ruowen Zhao, Yao Feng, Chendong Xiang, Yinze Rong, et al. Motus: A unified latent action world model. *arXiv preprint arXiv:2512.13030*, 2025.

[11] Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

[12] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[13] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[14] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[15] Qingwen Bu, Jisong Cai, Li Chen, Xiuqi Cui, Yan Ding, Siyuan Feng, Shenyuan Gao, Xindong He, Xuan Hu, Xu Huang, et al. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. *arXiv preprint arXiv:2503.06669*, 2025.

[16] Jisong Cai, Long Ling, Shiwei Chu, Zhongshan Liu, Jiayue Kang, Zhixuan Liang, Wenjie Xu, Yinan Mao, Weinan Zhang, Xiaokang Yang, et al. AHA-WAM: Asynchronous horizon-adaptive world-action modeling with observation-guided context routing. *arXiv preprint arXiv:2606.09811*, 2026.

[17] Rui Cai, Jun Guo, Xinze He, Piaopiao Jin, Jie Li, Bingxuan Lin, Futeng Liu, Wei Liu, Fei Ma, Kun Ma, et al. Xiaomi-Robotics-0: An open-sourced vision-language-action model with real-time execution. *arXiv preprint arXiv:2602.12684*, 2026.

[18] Zhongang Cai, Ruisi Wang, Chenyang Gu, Fanyi Pu, Junxiang Xu, Yubo Wang, Wanqi Yin, Zhitao Yang, Chen Wei, Qingping Sun, et al. Scaling spatial intelligence with multimodal foundation models. *arXiv preprint arXiv:2511.13719*, 2025.

[19] Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, et al. Worldvla: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*, 2025.

[20] Gongwei Chen, Leyang Shen, Rui Shao, Xiang Deng, and Liqiang Nie. Lion: Empowering multimodal large language model with dual-level visual knowledge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26540–26550, 2024.

[21] Jierun Chen, Fangyun Wei, Jinjing Zhao, Sizhe Song, Bohuai Wu, Zhuoxuan Peng, S-H Gary Chan, and Hongyang Zhang. Revisiting referring expression comprehension evaluation in the era of large multimodal models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 513–524, 2025.

[22] Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Conghui He, Jiaqi Wang, Feng Zhao, and Dahua Lin. Sharegpt4v: Improving large multi-modal models with better captions. In *European Conference on Computer Vision*, pages 370–387. Springer, 2024.

[23] Tianxing Chen, Yue Chen, Zixuan Li, Junyuan Tang, Kailun Su, Haoran Lu, Weijie Wan, Baijun Chen, Songling Liu, Haowen Yan, Honghao Su, Zhiyang Dou, Kaixuan Wang, Dandan Zhang, Yunze Liu, Yan Qin, Qiwei Liang, Qiwei Wu, Zijian Lin, Wenwei Lin, Yuran Wang, Minghua He, Tianshu Wu, Ruihai Wu, Jingquan Zhou, Kai-Chong Lei, Haibao Yu, Yuanfeng Ji, Weiyang Jin, Guanyu Lin, Xiaofan Li, Qi Xiong, Renjing Xu, Zhongyu Li, Wenhao Chai, Enze Xie, Ziwei Wang, Yao Mu, Hao Dong, Wojciech Matusik, Mingyu Ding, Wenbo Ding, Ping Luo, and Masayoshi Tomizuka. RoboDojo: A unified sim-and-real benchmark for comprehensive evaluation of generalist robot manipulation policies. *arXiv preprint arXiv:2607.04434*, 2026.

[24] Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025.

[25] Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, et al. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271*, 2024.

[26] Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. *arXiv preprint arXiv:2402.10329*, 2024.

[27] StarVLA Community. Starvla: A lego-like codebase for vision-language-action model developing. *arXiv preprint arXiv:2604.05014*, 2026.

[28] Wenliang Dai, Junnan Li, Dongxu Li, Anthony Tiong, Junqi Zhao, Weisheng Wang, Boyang Li, Pascale N Fung, and Steven Hoi. Instructblip: Towards general-purpose vision-language models with instruction tuning. *Advances in neural information processing systems*, 36:49250–49267, 2023.

[29] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. The epic-kitchens dataset: Collection, challenges and baselines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(11):4125–4141, 2020.

[30] Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, et al. Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 91–104, 2025.

[31] Dexmal Team. DM0.5: An open-world foundation model for general-purpose embodied intelligence. Technical report and model release, 2026.

[32] Ria Doshi, Homer Walke, Oier Mees, Sudeep Dasari, and Sergey Levine. Scaling cross-embodied learning: One policy for manipulation, navigation, locomotion and aviation. *arXiv preprint arXiv:2408.11812*, 2024.

[33] Danny Driess, Fei Xia, Mehdi SM Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

[34] Heng Fang, Shangru Li, Shuhan Wang, Xuanyang Xi, Dingkang Liang, and Xiang Bai. Towards generalizable robotic manipulation in dynamic environments. *arXiv preprint arXiv:2603.15620*, 2026.

[35] Hongjie Fang, Hao-Shu Fang, Yiming Wang, Jieji Ren, Jingjing Chen, Ruo Zhang, Weiming Wang, and Cewu Lu. Airexo: Low-cost exoskeletons for learning whole-arm manipulation in the wild. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15031–15038. IEEE, 2024.

[36] Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, et al. Libero-plus: In-depth robustness analysis of vision-language-action models. *arXiv preprint arXiv:2510.13626*, 2025.

[37] GenRobot. 10kh-realomin-opendata. Hugging Face Dataset, 2026.

[38] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18995–19012, 2022.

[39] Jun Guo, Qiwei Li, Peiyan Li, Zilong Chen, Nan Sun, Yifei Su, Heyun Wang, Yuan Zhang, Xinghang Li, and Huaping Liu. Unified 4d world action modeling from video priors with asynchronous denoising, 2026.

[40] Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, et al. Cogagent: A visual language model for gui agents. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14281–14290, 2024.

[41] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023.

[42] Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022.

[43] Wenxuan Huang, Bohan Jia, Zijie Zhai, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Xu Tang, Yao Hu, and Shaohui Lin. Vision-r1: Incentivizing reasoning capability in multimodal large language models. *arXiv preprint arXiv:2503.06749*, 2025.

[44] Chia-Yu Hung, Qi Sun, Pengfei Hong, Amir Zadeh, Chuan Li, U Tan, Navonil Majumder, Soujanya Poria, et al. Nora: A small open-sourced generalist vision language action model for embodied tasks. *arXiv preprint arXiv:2504.19854*, 2025.

[45] 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.

[46] InternVLA-A1 Team. InternVLA-A1: Unifying understanding, generation and action for robotic manipulation. *arXiv preprint arXiv:2601.02456*, 2026.

[47] Tao Jiang, Tianyuan Yuan, Yicheng Liu, Chenhao Lu, Jianning Cui, Xiao Liu, Shuiqi Cheng, Jiyang Gao, Huazhe Xu, and Hang Zhao. Galaxea open-world dataset and G0 dual-system vla model. *arXiv preprint arXiv:2509.00576*, 2025.

[48] Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: Robot manipulation with multimodal prompts. 2023.

[49] Ivan Kapelyukh, Vitalis Vosylius, and Edward Johns. Dall-e-bot: Introducing web-scale diffusion models to robotics. *IEEE Robotics and Automation Letters*, 8(7):3956–3963, 2023.

[50] Siddharth Karamcheti, Suraj Nair, Annie S Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. Language-driven representation learning for robotics. *arXiv preprint arXiv:2302.12766*, 2023.

[51] Simar Kareer, Dhruv Patel, Ryan Punamiya, Pranay Mathur, Shuo Cheng, Chen Wang, Judy Hoffman, and Danfei Xu. Egomimic: Scaling imitation learning via egocentric video. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13226–13233. IEEE, 2025.

[52] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

[53] Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

[54] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[55] Jason Lee, Jiafei Duan, Haoquan Fang, Yuquan Deng, Shuo Liu, Boyang Li, Bohan Fang, Jieyu Zhang, Yi Ru Wang, Sangho Lee, et al. Molmoact: Action reasoning models that can reason in space. *arXiv preprint arXiv:2508.07917*, 2025.

[56] Bo Li, Hao Zhang, Kaichen Zhang, Dong Guo, Yuanhan Zhang, Renrui Zhang, Feng Li, Ziwei Liu, and Chunyuan Li. Llava-next: What else influences visual instruction tuning beyond data?, May 2024.

[57] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024.

[58] Fuhao Li, Wenxuan Song, Han Zhao, Jingbo Wang, Pengxiang Ding, Donglin Wang, Long Zeng, and Haoang Li. Spatial forcing: Implicit spatial representation alignment for vision-language-action model. *arXiv preprint arXiv:2510.12276*, 2025.

[59] Zaijing Li, Yuquan Xie, Rui Shao, Gongwei Chen, Dongmei Jiang, and Liqiang Nie. Optimus-1: Hybrid multimodal memory empowered agents excel in long-horizon tasks. *Advances in neural information processing systems*, 37:49881–49913, 2024.

[60] Zhang Li, Biao Yang, Qiang Liu, Zhiyin Ma, Shuo Zhang, Jingxu Yang, Yabo Sun, Yuliang Liu, and Xiang Bai. Monkey: Image resolution and text label are important things for large multi-modal models. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 26763–26773, 2024.

[61] Zhiqi Li, Guo Chen, Shilong Liu, Shihao Wang, Vibashan VS, Yishen Ji, Shiyi Lan, Hao Zhang, Yilin Zhao, Subhashree Radhakrishnan, et al. Eagle 2: Building post-training data strategies from scratch for frontier vision-language models. *arXiv preprint arXiv:2501.14818*, 2025.

[62] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International conference on robotics and automation (ICRA)*, pages 9493–9500. IEEE, 2023.

[63] Kevin Lin, Christopher Agia, Toki Migimatsu, Marco Pavone, and Jeannette Bohg. Text2motion: From natural language instructions to feasible plans. *Autonomous Robots*, 47(8):1345–1365, 2023.

[64] Xuewu Lin, Tianwei Lin, Yun Du, Hongyu Xie, Yiwei Jin, Jiawei Li, Shijie Wu, Qingze Wang, Mengdi Li, Mengao Zhao, et al. HoloBrain-0 technical report. *arXiv preprint arXiv:2602.12062*, 2026.

[65] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[66] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 26296–26306, 2024.

[67] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

[68] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

[69] Yicheng Liu, Zibin Dong, Baijun Ye, Tianyuan Yuan, Tao Jiang, Anqi Yang, Shicheng Cao, Haonan Liu, Yue Sun, Zihan Guo, et al. G0.5: One autoregressive stream for robot reasoning and action, 2026.

[70] Corey Lynch, Ayzaan Wahid, Jonathan Tompson, Tianli Ding, James Betker, Robert Baruch, Travis Armstrong, and Pete Florence. Interactive language: Talking to robots in real time. *IEEE Robotics and Automation Letters*, 2023.

[71] Haoxiang Ma, Junhao Cai, Xiaoxu Xu, Hao Li, Yuyin Yang, Yang Tian, Jiafei Cao, Hongrui Zhu, Zherui Qiu, et al. InternVLA-A1.5: Unifying understanding, latent foresight, and action for compositional generalization. *arXiv preprint arXiv:2607.04988*, 2026.

[72] Ajay Mandlekar, Jonathan Booher, Max Spero, Albert Tung, Anchit Gupta, Yuke Zhu, Animesh Garg, Silvio Savarese, and Li Fei-Fei. Scaling robot supervision to hundreds of hours with roboturk: Robotic manipulation dataset through human reasoning and dexterity. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1048–1055. IEEE, 2019.

[73] Ajay Mandlekar, Yuke Zhu, Animesh Garg, Jonathan Booher, Max Spero, Albert Tung, Julian Gao, John Emmons, Anchit Gupta, Emre Orbay, et al. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. In *Conference on Robot Learning*, pages 879–893. PMLR, 2018.

[74] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

[75] Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6892–6903. IEEE, 2024.

[76] Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.

[77] Kenneth Shaw, Shikhar Bahl, and Deepak Pathak. Videodex: Learning dexterity from internet videos. In *Conference on Robot Learning*, pages 654–665. PMLR, 2023.

[78] Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, et al. Smolvla: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.

[79] Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. Progprompt: Generating situated robot task plans using large language models. *arXiv preprint arXiv:2209.11302*, 2022.

[80] Shweta Singh, Aayan Yadav, Jitesh Jain, Humphrey Shi, Justin Johnson, and Karan Desai. Benchmarking object detectors with coco: A new path forward. In *European Conference on Computer Vision*, pages 279–295. Springer, 2024.

[81] Shuhan Tan, Kairan Dou, Yue Zhao, and Philipp Krähenbühl. Interactive post-training for vision-language-action models. *arXiv preprint arXiv:2505.17016*, 2025.

[82] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[83] Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, et al. Gemini robotics: Bringing ai into the physical world. *arXiv preprint arXiv:2503.20020*, 2025.

[84] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

[85] Xiaoyu Tian, Junru Gu, Bailin Li, Yicheng Liu, Yang Wang, Zhiyong Zhao, Kun Zhan, Peng Jia, Xianpeng Lang, and Hang Zhao. Drivevlm: The convergence of autonomous driving and large vision-language models. *arXiv preprint arXiv:2402.12289*, 2024.

[86] Yang Tian, Yuyin Yang, Yiman Xie, Zetao Cai, Xu Shi, Ning Gao, Hangxu Liu, Xuekun Jiang, Zherui Qiu, Feng Yuan, et al. Interndata-a1: Pioneering high-fidelity synthetic data for pre-training generalist policy. *arXiv preprint arXiv:2511.16651*, 2025.

[87] Mel Vecerik, Carl Doersch, Yi Yang, Todor Davchev, Yusuf Aytar, Guangyao Zhou, Raia Hadsell, Lourdes Agapito, and Jon Scholz. Robotap: Tracking arbitrary points for few-shot visual imitation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5397–5403. IEEE, 2024.

[88] Homer Rich Walke, Kevin Black, Tony Z Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, pages 1723–1736. PMLR, 2023.

[89] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

[90] Yuqi Wang, Xinghang Li, Wenxuan Wang, Junbo Zhang, Yingyan Li, Yuntao Chen, Xinlong Wang, and Zhaoxiang Zhang. Unified vision-language-action model. *arXiv preprint arXiv:2506.19850*, 2025.

[91] Chuan Wen, Xingyu Lin, John So, Kai Chen, Qi Dou, Yang Gao, and Pieter Abbeel. Any-point trajectory modeling for policy learning. *arXiv preprint arXiv:2401.00025*, 2023.

[92] Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. *arXiv preprint arXiv:2312.13139*, 2023.

[93] Philipp Wu, Yide Shentu, Zhongke Yi, Xingyu Lin, and Pieter Abbeel. Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 12156–12163. IEEE, 2024.

[94] Shihan Wu, Xuecheng Liu, Shaoxuan Xie, Pengwei Wang, Xinghang Li, Bowen Yang, Zhe Li, Kai Zhu, Hongyu Wu, Yiheng Liu, et al. Robocoin: An open-sourced bimanual robotic data collection for integrated manipulation. *arXiv preprint arXiv:2511.17441*, 2025.

[95] Wei Wu, Fan Lu, Yunnan Wang, Shuai Yang, Shi Liu, Fangjing Wang, Qian Zhu, He Sun, Yong Wang, Shuailei Ma, et al. A pragmatic vla foundation model. *arXiv preprint arXiv:2601.18692*, 2026.

[96] Xiaomi Robotics Team, Jun Guo, Piaopiao Jin, Jason Li, Peiyan Li, Yingyan Li, Futeng Liu, Wanli Peng, Optimus Qin, Yifei Su, et al. Xiaomi-Robotics-1: Scaling vision-language-action models with over 100k hours of real-world trajectories. *arXiv preprint arXiv:2607.15330*, 2026.

[97] Ganlin Yang, Zhangzheng Tu, Yuqiang Yang, Sitong Mao, Junyi Dong, Tianxing Chen, Jiaqi Peng, Jing Xiong, Jiafei Cao, Jifeng Dai, et al. EventVLA: Event-driven visual evidence memory for long-horizon vision-language-action policies. *arXiv preprint arXiv:2606.20092*, 2026.

[98] Jonathan Yang, Catherine Glossop, Arjun Bhorkar, Dhruv Shah, Quan Vuong, Chelsea Finn, Dorsa Sadigh, and Sergey Levine. Pushing the limits of cross-embodiment learning for manipulation and navigation. *arXiv preprint arXiv:2402.19432*, 2024.

[99] Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. Visionzip: Longer is better but not necessary in vision language models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19792–19802. IEEE, 2025.

[100] Senqiao Yang, Junyi Li, Xin Lai, Jinming Wu, Wei Li, Zejun MA, Bei Yu, Hengshuang Zhao, and Jiaya Jia. Visionthink: Smart and efficient vision language model via reinforcement learning. *Advances in Neural Information Processing Systems*, 38:95187–95227, 2026.

[101] Senqiao Yang, Tianyuan Qu, Xin Lai, Zhuotao Tian, Bohao Peng, Shu Liu, and Jiaya Jia. Lisa++: An improved baseline for reasoning segmentation with large language model. *arXiv preprint arXiv:2312.17240*, 2023.

[102] Senqiao Yang, Kaichen Zhang, Zhaoyang Jia, Jinghao Guo, Yifei Shen, Xinjie Zhang, Xiaoyi Zhang, Haoqing Wang, Xiao Li, Peng Zhang, et al. Mage-vl: An efficient codec-native streaming multimodal foundation model. *arXiv preprint arXiv:2607.24904*, 2026.

[103] Yandan Yang, Shuang Zeng, Tong Lin, Xinyuan Chang, Dekang Qi, Junjin Xiao, Haoyun Liu, Ronghan Chen, Yuzhi Chen, Dongjie Huo, et al. Abot-m0: Vla foundation model for robotic manipulation with action manifold learning. *arXiv preprint arXiv:2602.11236*, 2026.

[104] Angen Ye, Boyuan Wang, Chaojun Ni, Guan Huang, Guosheng Zhao, Hao Li, Hengtao Li, Jie Li, Jindi Lv, Jingyu Liu, et al. GigaWorld-Policy: An efficient action-centered world–action model. *arXiv preprint arXiv:2603.17240*, 2026.

[105] Jinhui Ye, Ning Gao, Senqiao Yang, Jinliang Zheng, Zixuan Wang, Yuxin Chen, Pengguang Chen, Yilun Chen, Shu Liu, and Jiaya Jia. StarVLA-α: Reducing complexity in vision-language-action systems. *arXiv preprint arXiv:2604.11757*, 2026.

[106] Qilang Ye, Zitong Yu, Rui Shao, Xinyu Xie, Philip Torr, and Xiaochun Cao. Cat: Enhancing multimodal large language model to answer questions in dynamic audio-visual scenarios. In *European Conference on Computer Vision*, pages 146–164. Springer, 2024.

[107] Seonghyeon Ye, Joel Jang, Byeongguk Jeon, Sejune Joo, Jianwei Yang, Baolin Peng, Ajay Mandlekar, Reuben Tan, Yu-Wei Chao, Bill Yuchen Lin, et al. Latent action pretraining from videos. *arXiv preprint arXiv:2410.11758*, 2024.

[108] Tianhe Yu, Ted Xiao, Austin Stone, Jonathan Tompson, Anthony Brohan, Su Wang, Jaspiar Singh, Clayton Tan, Jodilyn Peralta, Brian Ichter, et al. Scaling robot learning with semantically imagined experience. *arXiv preprint arXiv:2302.11550*, 2023.

[109] Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-WAM: Do world action models need test-time future imagination? *arXiv preprint arXiv:2603.16666*, 2026.

[110] Wentao Yuan, Jiafei Duan, Valts Blukis, Wilbert Pumacay, Ranjay Krishna, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. Robopoint: A vision-language model for spatial affordance prediction for robotics. *arXiv preprint arXiv:2406.10721*, 2024.

[111] Borong Zhang, Jiahao Li, Jiachen Shen, Yishuai Cai, Yuhao Zhang, Yuanpei Chen, Juntao Dai, Jiaming Ji, and Yaodong Yang. Vla-arena: An open-source framework for benchmarking vision-language-action models. *arXiv preprint arXiv:2512.22539*, 2025.

[112] He Zhang, Lingzhu Xiang, Haitao Lin, Zeyu Huang, Minghui Wang, Dingyan Zhong, Yubo Dong, Yihao Wu, Yongming Rao, Dongsheng Zhang, et al. Hy-Embodied-0.5-VLA: From vision-language-action models to a real-world robot learning stack. *arXiv preprint arXiv:2606.14409*, 2026.

[113] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[114] Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, et al. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model. *arXiv preprint arXiv:2510.10274*, 2025.

[115] Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR, 2023.

Contents

A Additional Related Work **21**

B Additional Main Results **21**

B.1 Additional Results on Franka 21

B.2 Additional Results on Agilex 22

B.3 Full RoboDojo Leaderboard 22

C Effectiveness of Shallow-Layer Protection During Pre-training **22**

C.1 Motivation and Design 22

C.2 Ablation Results 23

D VLM Co-training for Representation Preservation **24**

D.1 Why Mix Auxiliary Data During VLA Pre-training? 24

D.2 Auxiliary Data Mixture 24

D.3 Effect of Different Auxiliary Co-training Data 25

E Representation Effects of Head-Diverse Pre-training **27**

E.1 Head-Diverse Pre-training Prevents Decoder Lock-in 27

E.2 Does Head Diversity Also Improve Same-head Adaptation? 28

F Unified Joint Space and Wrap-Aware Loss **28**

F.1 Unified joint space by wrapping. 28

F.2 Experiments. 29

G Adding Heterogeneous UMI Data to Continued Pre-training **29**

H Details of Data Cleaning **30**

I Details of Action Heads **31**

I.1 Representative Action-Head Families 31

I.2 Unified Action Representation 32

J Details of Real-world Experiments **34**

J.1 Hardware and observation. 34

J.2 Expert Data Collection and Fine-tuning Protocol 34

J.3 Detailed Real-World Results and Failure Analysis 34

J.4 Detailed Prompts for Each Real-World Robotic Arm Task 35

J.5 Visualization 36

K Future Work and Broader Impact. **36**

A Additional Related Work

Generalist Robot Policies and Data Pretraining

The emergence of foundation models [13] has catalyzed a shift from task-specific controllers to generalist robotic policies. Early approaches often deploy pre-trained models as high-level semantic planners that interface with low-level, robot-specific skills [115, 33, 42, 62, 63, 79]. To overcome the hierarchy's rigidity, Vision-Language-Action (VLA) models have gained prominence by fine-tuning backbones such as Eagle-2 [61] or PaliGemma [5]directly on embodied data for end-to-end optimization [12, 115, 11, 54, 103, 83, 105, 27]. Modern VLA architectures frequently incorporate advanced generation techniques like flow-matching [65, 68] and action chunking [113]. While some frameworks use Mixture-of-Experts to bridge vision and action [12], others utilize cross-attention or embodiment-specific projectors to facilitate multi-robot transfer across latent [84].

The performance of these models is fundamentally driven by data scaling. While high-fidelity trajectories from teleoperation [2, 73, 72, 93] have been scaled into massive datasets with thousands of hours of experience [15, 12, 14, 88, 70, 75], the high cost of collection remains a bottleneck. Consequently, alternative sources such as instrumented human trajectories [26, 35] and internet-scale human video datasets [29, 38] are increasingly leveraged. These non-robotic sources serve either as a substrate for representation pretraining [50, 92] or as a means of joint learning through motion-based intermediate representations [8, 107]. In contrast to prior efforts that often rely on proprietary training stacks, our work systematically explores the nuances of learning from a diverse mixture of public robot data, human videos, and synthetic samples.

Cross-task and cross-embodiment generalization Achieving robust generalization across diverse environments, tasks, and embodiments is a central objective in robot learning. A prevalent strategy involves leveraging human video data to bootstrap policies, either via general representation learning [74, 9], direct supervision through human motion cloning [77, 6, 51] or the extraction of 2D point tracks as intermediate representations [7, 87, 91]. Parallel to these data-driven efforts, another line of work integrates Internet-scale foundation models into training or inference pipelines to inherit broad semantic knowledge [49, 108, 48]. With the emergence of large-scale, cross-embodiment datasets [75], recent research has increasingly focused on enhancing transfer learning between heterogeneous robot platforms [32, 98].

Vision Language Models Modern Vision-Language Models (VLMs) [20, 66, 28, 3, 60, 89, 99, 102] have transitioned from task-specific designs toward unified frameworks capable of general-purpose multimodal reasoning [59, 106, 101]. Typically comprising a vision encoder, a projector, and a Large Language Model (LLM) backbone [66, 89], these models unify diverse tasks as textual outputs. This shift is driven by visual instruction tuning [67, 25], which grants VLMs the flexibility to follow open-ended commands and generalize to out-of-distribution scenarios. Recent advancements further push these boundaries through dynamic resolution [3], unified image-video processing [57], and reinforcement-learning-enhanced reasoning [43, 100], significantly bolstering real-world perception. While VLMs have demonstrated high-level planning in domains like autonomous driving [85] and GUI interaction [40], they often remain passive observers. The emerging frontier lies in bridging this "embodiment gap" by extending VLM reasoning into direct physical interaction. By grounding these models in action spaces, Vision-Language-Action (VLA) models leverage the rich semantic and spatial knowledge of VLMs to enable robust manipulation in complex environments.

B Additional Main Results

B.1 Additional Results on Franka

Results on VLA-Arena. Table 3 reports success on VLA-Arena across four behavioral generalization axes. VLAct achieves the best score on every axis and reaches **54.8%** overall. Compared to Qwen3-VL-OFT, our VLAct improves the average score by **21.4%** despite using the same backbone family and downstream OFT head. This large gap again isolates the effect of the representation-centric pre-training recipe rather than the base VLM or the fine-tuning head.

VLAct also outperforms the strongest public baseline, $\pi_{0.5}$, by 10.5 points overall, with especially large gains on *Long-Horizon* and *Safety*, 21.0 and 11.7 points, respectively. Since all methods are fine-tuned on the same downstream task data, the improvement directly demonstrates the effectiveness

Table 3: **VLA-Arena results.** Success rate (%) averaged over difficulty levels L0/L1/L2 within each task category. The final average follows the official suite weighting over 11 task suites. Dark blue highlights VLAct and per-column best results; light blue highlights the strongest public baseline by average and per-column second-best results.

Method	Safety	Distractor	Extrap.	Long-H.	Avg.
SmolVLA [78]	16.5	21.0	10.0	24.7	16.3
π_0-FAST [76]	34.1	39.0	4.2	20.7	25.6
GR00T-N1.6 [11]	32.8	40.7	16.7	10.3	27.8
Qwen3-VL-π [3]	38.3	45.3	22.4	25.3	34.1
UniVLA [90]	45.5	41.3	31.1	22.0	38.7
OpenVLA [54]	41.9	43.0	36.7	26.7	39.3
OpenVLA-OFT [53]	43.2	52.3	30.4	26.7	39.9
π_0 [12]	49.7	43.7	32.7	31.3	42.3
$\pi_{0.5}$ [45]	51.5	54.0	31.1	29.0	44.3
Qwen3-VL-OFT	36.1	40.0	24.7	32.7	33.4
VLAct	**63.2**	**64.0**	**36.9**	**50.0**	**54.8**

Table 4: **DOMINO dynamic manipulation results.** We evaluate one policy on all 35 DOMINO tasks under the clean dynamic setting. DOMINO reports Success Rate (SR) and Manipulation Score (MS). Dark blue highlights VLAct-OFT; light blue highlights the strongest baseline.

Model	Backbone	SR ↑	MS ↑
OpenVLA	Llama-2	1.54	6.10
RDT-1B	DiT-1B	5.34	17.71
π_0	PaliGemma	8.17	23.96
π_0-FAST	PaliGemma	3.54	20.87
$\pi_{0.5}$	PaliGemma	9.63	26.17
InternVLA-M1	InternVL	5.40	27.57
OpenVLA-OFT	Llama-2	9.06	24.06
Qwen3VL-OFT	Qwen3-VL	10.86	30.49
VLAct-OFT	Qwen3-VL	**18.50**	**34.20**

of VLAct as a pretrained VLA backbone: it provides a substantially stronger starting point for downstream policy learning and yields better generalization across challenging behavioral settings.

B.2 Additional Results on Agilex

Results on DOMINO. Table 4 reports results on DOMINO, a dynamic manipulation benchmark with moving objects and changing scene states. VLAct-OFT achieves the best performance on both metrics, reaching **18.50** SR and **34.20** MS. Compared with Qwen3VL-OFT, the strongest baseline with the same Qwen3-VL backbone family and OFT downstream head, VLAct improves SR by **7.64** points and MS by 3.71 points. Since both models are fine-tuned under the same downstream setting, the improvement highlights the benefit of VLAct as a pretrained backbone for dynamic bimanual manipulation. Together, the RoboTwin 2.0 and DOMINO results show that VLAct is effective across both visually randomized static manipulation and temporally evolving dynamic manipulation.

B.3 Full RoboDojo Leaderboard

Table 5 reports the per-capability RoboDojo results discussed in Sec. 4.4.

C Effectiveness of Shallow-Layer Protection During Pre-training

C.1 Motivation and Design

Our VLA-oriented backbone is initialized from a VLM trained on broad vision-language data. This initialization already contains useful low-level visual features, spatial cues, and early vision-language

Table 5: **RoboDojo simulation results.** Top 20 policies from the official August 24, 2026 leaderboard, ordered by average score. Each cell reports partial-progress score / success rate (%). Gen.-Std. and Gen.-Rand. denote the standard and randomized Generalization settings. The full leaderboard contains 35 policies; VLAct ranks eighth by score and sixth by success rate. The dark-blue row highlights VLAct; light-blue rows and † identify explicitly designated WAM entries.

Method	Gen.-Std.	Gen.-Rand.	Precision	Long-Horizon	Memory	Open	Average
DM0.5 [31]	23.49 / 18.00	8.06 / 4.00	24.82 / 16.75	33.70 / 19.50	47.74 / 47.44	2.43 / 2.08	24.90 / 19.34
GalaxeaVLA (G0.5) [69]	26.74 / 20.00	11.16 / 6.00	28.25 / 20.42	44.12 / 32.25	8.61 / 7.33	1.73 / 1.58	20.23 / 14.88
Xiaomi-Robotics-1 [96]	35.65 / 28.00	11.44 / 6.00	26.69 / 18.83	38.39 / 23.67	7.81 / 6.56	3.94 / 3.58	20.07 / 13.93
Hy-Embodied-0.5-VLA [112]	21.98 / 17.00	1.57 / 0.00	13.81 / 8.00	25.74 / 14.92	13.37 / 12.11	0.65 / 0.58	13.07 / 8.80
Spatial Forcing [58]	21.25 / 15.00	6.98 / 4.00	17.33 / 10.58	23.26 / 14.58	5.43 / 4.11	1.78 / 1.58	12.38 / 8.04
$\pi_{0.5}$ [45]	20.93 / 15.00	5.82 / 1.00	12.40 / 5.50	23.54 / 14.67	5.78 / 4.56	1.98 / 1.67	11.41 / 6.91
InternVLA-A1.5 [71]	16.81 / 12.00	3.90 / 2.00	15.23 / 10.17	23.80 / 13.75	4.93 / 3.56	1.43 / 1.42	11.15 / 7.14
VLAct	16.33 / 12.00	2.74 / 1.00	20.62 / 15.25	20.12 / 13.67	0.66 / 0.56	2.37 / 2.25	**10.66 / 7.60**
X-VLA [114]	17.90 / 12.00	3.04 / 1.00	18.32 / 12.00	16.53 / 9.75	4.76 / 3.56	0.55 / 0.50	10.13 / 6.52
X-WAM† [39]	11.24 / 5.00	3.54 / 1.00	6.72 / 1.83	17.47 / 9.08	6.32 / 4.67	0.57 / 0.25	7.69 / 3.83
Xiaomi-Robotics-0 [17]	13.81 / 11.00	1.05 / 0.00	8.42 / 4.58	13.51 / 6.92	5.07 / 3.67	0.22 / 0.17	6.93 / 4.18
StarVLA-α [105]	7.54 / 5.00	0.33 / 0.00	9.90 / 4.33	14.15 / 6.50	3.34 / 2.44	0.68 / 0.58	6.40 / 3.24
GigaWorld-Policy-0† [104]	10.28 / 6.00	0.41 / 0.00	6.15 / 1.83	15.51 / 8.92	3.46 / 2.22	0.54 / 0.50	6.20 / 3.27
GalaxeaVLA (G0) [47]	8.71 / 6.00	0.36 / 0.00	8.10 / 3.83	12.60 / 5.58	3.17 / 1.89	0.70 / 0.67	5.82 / 2.96
LingBot-VLA [95]	10.88 / 8.00	2.55 / 1.00	5.33 / 1.83	10.89 / 5.25	3.82 / 2.78	0.72 / 0.67	5.50 / 2.96
EventVLA [97]	6.68 / 3.00	1.22 / 0.00	10.13 / 5.75	5.05 / 0.83	4.92 / 4.78	0.80 / 0.75	4.97 / 2.81
AHA-WAM† [16]	10.32 / 6.00	1.26 / 0.00	5.86 / 2.42	8.61 / 2.67	2.97 / 2.78	0.88 / 0.83	4.82 / 2.39
ABot-M0 [103]	9.20 / 5.00	2.26 / 2.00	5.50 / 1.75	3.96 / 0.50	2.44 / 2.22	0.72 / 0.67	3.67 / 1.73
Fast-WAM† [109]	4.33 / 2.00	0.34 / 0.00	1.96 / 0.00	9.14 / 5.17	3.55 / 3.44	0.42 / 0.42	3.48 / 2.03
π_0 [12]	7.18 / 5.00	0.71 / 0.00	3.56 / 0.75	6.19 / 2.00	3.47 / 2.11	0.25 / 0.25	3.48 / 1.53

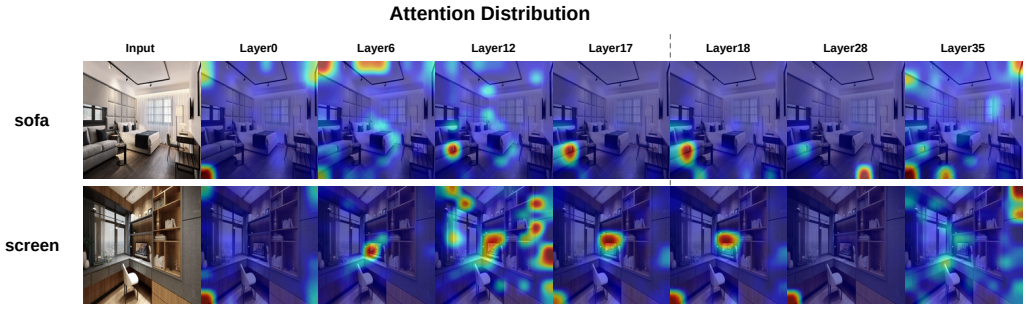


Figure 6: **Layer-wise attention visualization.** Lower layers attend more broadly to visual and spatial information, while deeper layers focus more on semantic and task-relevant regions. This motivates freezing the vision encoder and lower LLM layers during VLA pre-training to reduce representation drift under limited robot data.

alignment. In contrast, VLA pre-training data is much narrower: robot trajectories often come from fixed camera viewpoints, repeated manipulation scenes, and embodiment-specific action correlations. If all layers are updated end-to-end during VLA pre-training, gradients from this narrow distribution can shift the backbone away from its general VLM representation before the model has learned robust action-conditioned features.

Shallow-layer protection is designed to reduce this representation drift. As illustrated in Fig. 6, lower layers tend to preserve broad visual and spatial information, while deeper layers focus more on semantic and task-relevant regions. We therefore freeze the entire vision encoder and the lower half of the LLM layers during VLA pre-training. This protects the general perceptual and early alignment features inherited from the VLM, while leaving the upper LLM layers and action heads trainable for instruction following, task semantics, and action-conditioned reasoning. During downstream fine-tuning, we unfreeze the full model so that all parameters can adapt to the target task and embodiment.

C.2 Ablation Results

Table 6 reports the ablation. Updating the full backbone during VLA pre-training gives weaker downstream performance, with 78.9% on LIBERO-Plus and 77.1% on RoboTwin2.0. Freezing

Table 6: **Ablation on shallow-layer protection.** Freezing the vision encoder and lower LLM layers during VLA pre-training reduces representation drift and improves downstream fine-tuning. Dark blue highlights the full design; light blue highlights the strongest partial variant.

Pre-training update strategy	LIBERO-Plus	RoboTwin2.0
Update full backbone	78.9	77.1
Freeze vision encoder only	81.3	79.3
Freeze vision encoder + lower 1/2 LLM	**82.6**	**80.5**

only the vision encoder improves performance to 81.3 and 79.3%, showing that protecting visual features is already helpful. Freezing both the vision encoder and the lower half of the LLM performs best, reaching 82.6% on LIBERO-Plus and 80.5% on RoboTwin2.0, improving over full-backbone updating by 3.7% and 3.4%, respectively. These results support the role of shallow-layer protection in our recipe: it preserves the general VLM representation during pre-training, while leaving enough upper-layer capacity for action-aware adaptation.

D VLM Co-training for Representation Preservation

D.1 Why Mix Auxiliary Data During VLA Pre-training?

Our first design goal is to preserve the general foundation-model representation inherited from the VLM backbone while adapting it to robot control. During action-only VLA pre-training, the trainable layers are optimized through a narrow supervision channel: repeated robot scenes, embodiment-specific correlations, and sparse action targets. If used alone, this signal can make the backbone over-specialize to the robot trajectory distribution and weaken the semantic, visual, and language coverage that made the original VLM useful.

We therefore mix auxiliary non-action data into VLA pre-training. The goal is not simply to add more samples, but to provide representation-preserving supervision outside the action channel. Robot trajectories teach action-conditioned control, while auxiliary foundation-model data keeps the trainable layers closer to their original operating regime and exposes them to more diverse feature updates. Thus, auxiliary co-training acts as both a representation anchor and a source of feature diversity.

Among the auxiliary sources, image captions play the central role. Detailed captions are close to the dominant supervision used to train the original VLM and provide dense semantic gradients over objects, attributes, relations, and scene context. Grounding and spatial-QA data further constrain localized and spatial representations. We additionally include pure language instruction data as an off-domain diagnostic source: because it is not directly tied to robot perception or control, any gain from this source would suggest that the benefit of auxiliary co-training is not only task-specific transfer, but also representation preservation and diversification.

At every pre-training step, each minibatch contains both robot-trajectory samples and auxiliary samples. We optimize the action loss together with an auxiliary VLM cross-entropy loss:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{action}} + 0.5\,\mathcal{L}_{\text{VLM-CE}}.$$

D.2 Auxiliary Data Mixture

We study five types of auxiliary supervision: image captioning, bounding-box question answering (BBox-QA), point question answering (Point-QA), spatial question answering (Spatial-QA), and pure language instruction data. These sources provide different constraints on the backbone representation. Caption data preserves broad semantic grounding; BBox-QA and Point-QA strengthen localized visual grounding; Spatial-QA targets spatial relations and viewpoint reasoning; pure language instruction data supplies off-domain, non-action supervision on the language side of the backbone. Table 7 summarizes the processed auxiliary datasets used in our study.

Image caption data. We use image caption data as the primary semantic anchor for the VLM representation. LLaVA-ReCap-CC3M [56] provides detailed recaptions of web images, preserving broad visual coverage while replacing short or noisy alt-text with richer descriptions. ShareGPT4V [22] provides high-quality descriptive captions that emphasize object identity, attributes, spatial relations,

Table 7: **Auxiliary data used during VLA pre-training.** The mixture covers captioning, grounding, spatial reasoning, and pure language instruction supervision.

Type	Dataset
Image Caption	LLaVA-ReCap-CC3M [56] LLaVA OneVision [57]
BBox-QA	RefCOCO [21] COCO-ReM [80]
Point-QA	PixMo-Points [30] RoboPoint [110]
Spatial-QA	SenseNova-SI-800K [18]
Pure Language Instruction	Nemotron-SFT-Instruction-Following-Chat-v2

world knowledge, and visual details. These properties make caption data especially suitable for preserving the backbone's original vision-language representation during robot pre-training.

Grounding data. BBox-QA and Point-QA data provide localized visual-language supervision. RefCOCO [21] and COCO-ReM [80] are converted into bounding-box question-answering conversations, with coordinates normalized to the 0–1000 convention used by the Qwen3-VL family. PixMo-Points [30] and RoboPoint [110] provide point-based grounding signals. RoboPoint is particularly relevant to robotics because its supervision targets spatial affordance prediction from image-instruction pairs. For efficiency, we filter overly large images and unusually long samples, and restrict each grounding sample to at most 10 boxes or points.

Spatial-QA data. SenseNova-SI-800K [18] provides image-conditioned conversations designed to strengthen spatial reasoning. The data covers relative positions, directions, viewpoints, and multi-image spatial context. We keep only single-image cases to control memory cost and remove unusually long conversations for training stability.

We inspect samples from each processed data source to verify the converted format and supervision signal. Representative examples are shown in Fig. 7.

Pure language instruction data. We also include an off-domain auxiliary source from the NVIDIA Nemotron instruction-following dataset, *nvidia/Nemotron-SFT-Instruction-Following-Chat-v2*. This dataset is text-only and targets open-ended chat and instruction-following supervision, rather than visual grounding or robot control. We include it as a control-like auxiliary signal: if pure language supervision improves downstream VLA performance, then the benefit of auxiliary co-training cannot be explained solely by direct transfer from vision-language or robotics-related tasks. Instead, such a gain supports our representation-level interpretation: non-action supervision can help preserve the backbone's foundation-model operating regime and diversify feature updates during VLA pre-training.

D.3 Effect of Different Auxiliary Co-training Data

We compare different auxiliary supervision types while keeping the robot data, backbone, training recipe, and total pre-training budget fixed. Fig. 8 reports the LIBERO-Plus results. Relative to the robot-only baseline, all auxiliary sources improve downstream performance, indicating that non-action supervision helps produce a stronger VLA-oriented backbone.

Image-caption data gives the strongest single-source gain, improving the baseline from 75.0 to 82.6. This is consistent with its role as the main representation-preservation signal: detailed captions are close to the original VLM pre-training distribution and provide dense semantic supervision over objects, attributes, relations, and scene context. Spatial-QA, Point-QA, and BBox-QA also improve performance, suggesting that spatial reasoning and localized grounding provide useful complementary constraints on the learned representation.

Pure language instruction data also improves over the robot-only baseline. Since this source is text-only and not directly related to robot perception, visual grounding, or control, its gain is especially informative. It suggests that auxiliary co-training does not help only through direct transfer from

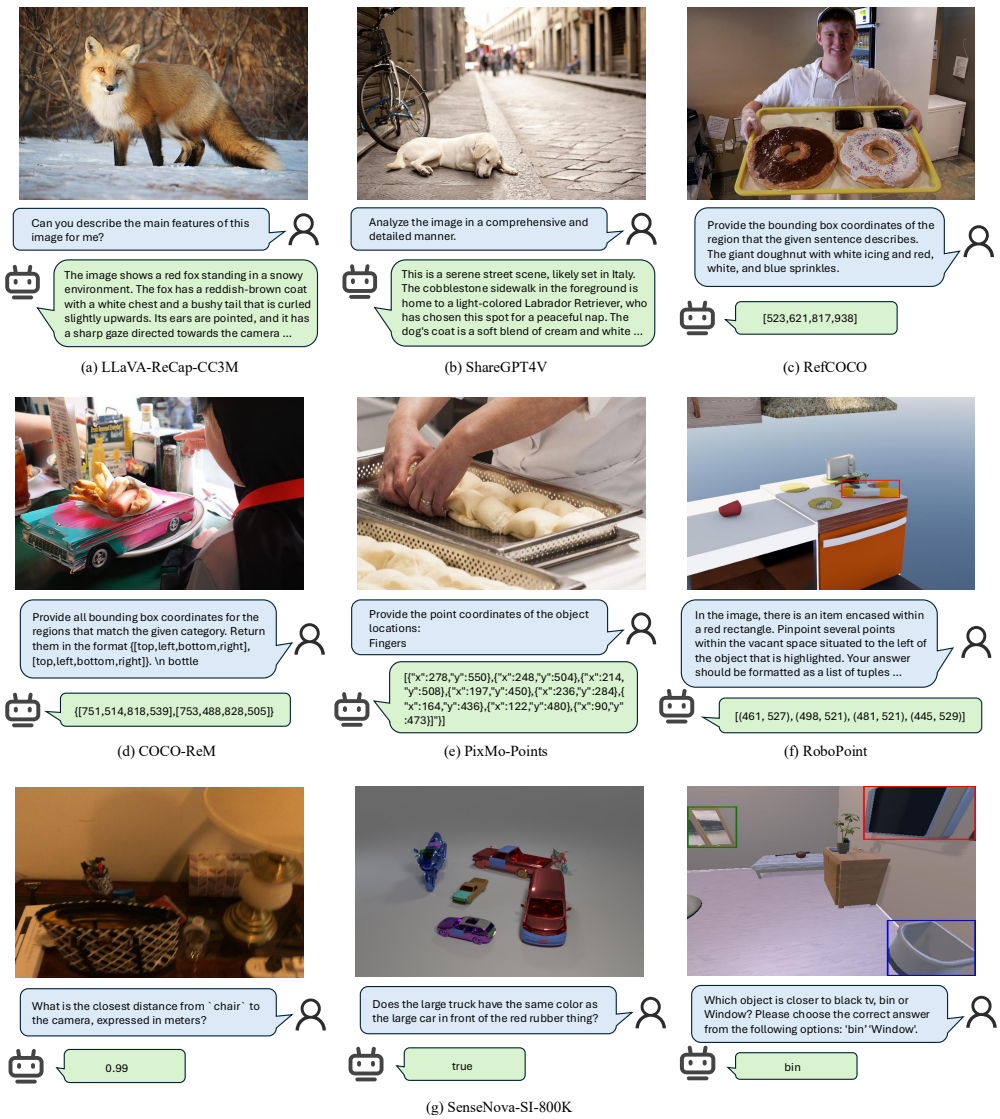


Figure 7: **Examples from the processed VLM data mixture.** We show representative samples from captioning, grounding, and spatial-reasoning datasets used during VLA pre-training.

VLM or robotics-related tasks. Instead, non-action supervision can help preserve the backbone's foundation-model representation and make feature updates more diverse than action-only training.

The mixed-data setting reaches 82.5, slightly below caption-only co-training. Since the total number of pre-training steps is fixed, mixing multiple auxiliary sources reduces the sampling frequency of image-caption data, which is the strongest single source in this ablation. This likely pulls the mixed result slightly below caption-only co-training, rather than indicating that data mixing itself is harmful.

Overall, this ablation supports the representation-preservation role of auxiliary co-training. Caption data provides the strongest single anchor, while grounding, spatial, and pure language instruction data provide additional gains. This justifies using auxiliary data not as generic extra supervision, but as a targeted mechanism for preserving and enriching the representation during VLA pre-training.

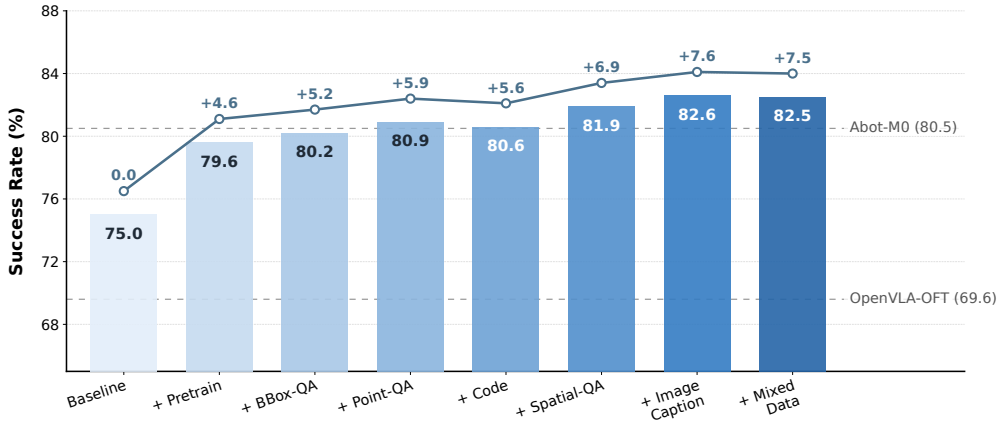


Figure 8: **Effect of auxiliary co-training data.** We compare different non-action supervision sources under the same robot data, model, and pre-training budget. Caption data gives the strongest single-source gain, while grounding, spatial-QA, and pure language instruction data also improve over robot-only training.

Table 8: **Head diversity reduces decoder lock-in.** PI fine-tuning degrades after OFT-only pre-training, but improves after OFT+GR00T pre-training even though PI is excluded during pre-training. Light blue highlights the key unseen-head comparison; dark blue marks the full configuration with direct PI exposure.

Pre-training heads	Seen PI in pre-training	PI FT	Δ vs. scratch
None	–	60.5	–
OFT	No	55.1	−5.4
OFT + GR00T	No	<u>63.1</u>	+2.6
OFT + PI + GR00T	Yes	**77.0**	+16.5

E Representation Effects of Head-Diverse Pre-training

E.1 Head-Diverse Pre-training Prevents Decoder Lock-in

The pilot study suggests that single-head continuous pre-training can reduce the reusability of the backbone with a different downstream action head. We refer to this tendency as *decoder lock-in*: the backbone may organize action-relevant features in a form that is effective for the pre-training decoder, but less accessible to alternative decoders introduced during fine-tuning. This is a concern for building a reusable VLA-oriented backbone, since downstream users may adopt different action heads depending on the embodiment, control frequency, action horizon, or deployment constraints.

Table 8 examines this issue using PI as the downstream fine-tuning head. Without additional continuous pre-training, PI fine-tuning reaches 60.5. After OFT-only pre-training, PI fine-tuning drops to 55.1, suggesting that the OFT pre-trained representation is not necessarily more compatible with the PI action head. In contrast, when the backbone is pre-trained with two continuous heads, OFT and GR00T, PI fine-tuning improves to 63.1, even though PI is not used during pre-training. The improvement is modest, but the comparison between OFT-only and OFT+GR00T is informative: adding a second pre-training head changes the unseen PI fine-tuning result from below the from scratch baseline to slightly above it.

When PI is also included during pre-training, PI fine-tuning reaches 77.0. This result should be interpreted separately, since the downstream head is now directly supervised during pre-training. It confirms that direct head exposure provides a strong compatibility benefit, but the more relevant comparison for decoder lock-in is between OFT-only and OFT+GR00T. Overall, this pilot study suggests that head diversity can reduce over-specialization to a single decoder and improve the accessibility of the learned representation to an unseen downstream head.

Table 9: **Same-head adaptation under head-diverse pre-training.** Head-diverse pre-training improves downstream performance even when the fine-tuning head matches one of the pre-training heads. Within each row, blue and lighter-blue cells mark the best and second-best results.

Fine-tune head	From scratch	Single-head pre-train	Head-diverse pre-train	Δ vs. single
OFT	61.7	<u>78.8</u>	**80.5**	+1.7
PI	60.5	<u>75.4</u>	**77.0**	+1.6
GR00T	51.2	<u>71.7</u>	**76.0**	+4.3

E.2 Does Head Diversity Also Improve Same-head Adaptation?

The main motivation for head-diverse pre-training is to improve compatibility across action heads. We further examine whether this comes at the cost of same-head adaptation, where the downstream fine-tuning head matches one of the heads used during pre-training. Ideally, a more head-agnostic representation should remain competitive when the same head is reused.

Table 9 compares three settings: fine-tuning without additional continuous pre-training, fine-tuning from a matched single-head pre-trained backbone, and fine-tuning from the full head-diverse pre-trained backbone. Head-diverse pre-training improves over the corresponding single-head baselines for OFT, PI, and GR00T, with gains of 1.7, 1.6, and 4.3 points, respectively. These gains are relatively moderate compared with the direct benefit of pre-training itself, but they indicate that multi-head supervision does not harm same-head adaptation in our setting.

This result is consistent with the representation-level view in Sec. 3.3. When the backbone is trained with multiple decoders, it must expose action information in a form that can be read by several action parameterizations. This reduces over-specialization to any single decoder and can produce action features that are more useful even when the same head is reused.

Summary. These diagnostics support the role of head diversity as a representation-compatibility mechanism. Single-head continuous pre-training can lock the backbone into one decoder geometry, making alternative downstream heads difficult to optimize. Adding multiple heads during pre-training reduces this lock-in by requiring the shared latent representation to support more than one action head. The same representation also improves same-head adaptation in our experiments, but the central objective is broader: to obtain a reusable VLA backbone that can support diverse downstream action heads.

F Unified Joint Space and Wrap-Aware Loss

F.1 Unified joint space by wrapping.

Absolute joint angles are naturally defined on a periodic space rather than on the Euclidean line. In particular, each revolute joint angle should be represented within the canonical range $[-\pi, \pi]$. Angles outside this interval do not correspond to new physical joint states; they are merely alternative parameterizations of equivalent poses. For example, $\pi + \epsilon$ and $-\pi + \epsilon$ describe nearly identical joint configurations, but their raw numerical values differ by almost 2π. If such equivalent angles are kept as different supervision targets, the model is forced to produce different outputs for the same underlying physical state, making it harder to learn a consistent action representation. Therefore, before training, we wrap all absolute joint-angle dimensions into $[-\pi, \pi]$:

$$a_{\text{wrap}} = \big(a + \pi\big) \bmod 2\pi - \pi. \tag{1}$$

This data-side wrapping constructs a unified joint space, where equivalent joint states are mapped to a consistent numerical representation.

Wrap-aware loss. Although data-side wrapping ensures that target actions lie in the canonical range, a standard regression loss can still produce an incorrect distance measure near the wrap boundary. For instance, when the target angle is close to $-\pi$ and the prediction is close to $+\pi$, the two angles are physically close, but the naive residual $\hat{a} - a$ is close to 2π. To correct this artifact, we further wrap the residual between the predicted and target actions before computing the regression loss. Given a predicted action $\hat{a}$ and a target action a on a revolute dimension, we define

$$\delta_{\text{wrap}} = \big((\hat{a} - a) + \pi\big) \bmod 2\pi - \pi, \tag{2}$$

Table 10: **Ablation of unified joint space and wrap-aware loss on RoboTwin.** All models are trained under the base setting using 50 tasks with 50 trajectories per task, and evaluated on the Clean split. The pre-training and fine-tuning protocols are kept identical across all variants. Dark blue highlights the full design; light blue highlights the strongest partial variant.

Setting	Unified Joint Space	Wrap Loss	Robotwin
Baseline			75.5
Unified joint space	✓		<u>78.6</u>
Unified joint space + Wrap-aware loss	✓	✓	**80.5**

and compute the wrap-aware loss with a standard L1 penalty:

$$\mathcal{L}_{\text{wrap}} = |\delta_{\text{wrap}}| \,. \tag{3}$$

We add this term to the original training objective of each action head. Specifically, for a head h, we denote its original action-learning objective as $\mathcal{L}^{(h)}_{\text{action}}$. This objective is the standard loss used by the corresponding head architecture, such as direct action regression for OFT, diffusion training loss for GR00T, or flow-matching loss for PI. The final objective for head h is therefore

$$\mathcal{L}^{(h)}_{\text{total}} = \mathcal{L}^{(h)}_{\text{action}} + \mathcal{L}^{(h)}_{\text{wrap}}, \tag{4}$$

where $\mathcal{L}^{(h)}_{\text{wrap}}$ is computed on the final action prediction of head h. For OFT, this corresponds to its direct regression output. For GR00T and PI, this corresponds to the final generated action sample after the denoising or flow-based generation process, rather than the intermediate noise prediction or velocity target. The wrap-aware term is applied only to absolute joint-angle dimensions. Non-periodic dimensions, such as gripper commands and delta end-effector translations, are excluded.

F.2 Experiments.

We ablate the effect of data-side joint-angle wrapping and loss-side residual wrapping on RoboTwin under the base setting. Specifically, we train each model using 50 tasks with 50 trajectories per task, and evaluate the trained policy on the Clean evaluation split. For a fair comparison, all variants use the same pre-training and fine-tuning protocol, and only differ in whether unified joint-space wrapping and wrap-aware loss are enabled. As shown in Table 10, the baseline that directly regresses raw joint angles achieves 75.5% success rate. Introducing the unified joint space improves the performance to 78.6%, suggesting that data-side wrapping reduces ambiguity in the action targets. Adding the wrap-aware loss further improves the success rate to 80.5%, showing that residual wrapping provides a more appropriate distance measure near the angular boundary.

G Adding Heterogeneous UMI Data to Continued Pre-training

Our main results use a fixed open-source pre-training mixture. Here we check whether the recipe can also make use of trajectories collected under a different embodiment and a different collection paradigm. We add 20K trajectories from 10Kh-RealOmin-OpenData [37], a corpus of real-world manipulation trajectories collected through a UMI-style handheld interface.

Setup. We select 20K trajectories, filtering primarily on trajectory continuity and replayability, and convert their actions into delta end-effector representations so that they are compatible with our partially unified action layout. All other settings are unchanged, including the model architecture, the rest of the pre-training mixture, the optimization budget, and the downstream fine-tuning protocol.

Result. As shown in Table 11, adding the 20K UMI trajectories improves LIBERO-Plus from 82.6 to 83.7, a gain of 1.1 points. Since these trajectories differ from the original mixture in both embodiment and collection paradigm, and received only lightweight filtering without task-specific curation, we read this as evidence that the recipe can absorb heterogeneous data rather than being disrupted by it.

Table 11: **Adding heterogeneous UMI trajectories to continued pre-training.** Adding 20K trajectories from 10Kh-RealOmin-OpenData [37], collected with a different embodiment and a UMI-style interface, improves downstream LIBERO-Plus performance. All other training and downstream fine-tuning settings are unchanged.

Continued pre-training data	LIBERO-Plus
Original VLAct mixture	82.6
Original VLAct mixture + 20K RealOmin trajectories	**83.7**

H Details of Data Cleaning

Robot Platforms and Action Spaces. We train our models on two robot platforms: a Franka single-arm platform and an AgileX dual-arm platform. For the AgileX dual-arm setting, we train on AgileX data from InternData-A1 and RoboCoin, and evaluate the trained models on RoboTwin. The AgileX action space is 14-dimensional, consisting of 12 absolute joint-angle dimensions for the two arms and 2 gripper dimensions. For the Franka single-arm setting, we train on DROID and MolmoAct, where DROID includes both v1.0.0 and v1.0.1, and evaluate on LIBERO-Plus and VLA-Arena. The Franka action space is 7-dimensional, consisting of 6-dimensional delta end-effector actions and 1 gripper dimension.

Unified Cleaning Pipeline. The training data are aggregated from multiple robot datasets collected with different platforms, annotation conventions, control interfaces, action spaces, and preprocessing pipelines. As a result, the raw data contain several sources of heterogeneity and noise. First, some trajectories have invalid or uninformative task names, which cannot provide meaningful language supervision for instruction-conditioned policy learning. Second, some action sequences contain abnormal values with unrealistically large magnitudes, which are likely caused by data collection failures, logging errors, or preprocessing artifacts. Third, even when the trajectories are valid, different datasets may use different action scales, control frequencies, and gripper conventions. Without proper alignment, the model may learn dataset-specific action patterns rather than a unified action representation that generalizes across datasets.

To address these issues, we apply a unified data cleaning pipeline before training. The pipeline aims to remove unreliable supervision signals while preserving useful trajectories as much as possible. It mainly consists of two stages. First, we remove trajectories or samples with invalid task names, including `unknown_task`, `n/a`, `none`, `no action`, whitespace-only task names, and empty task names. Such annotations do not provide meaningful language supervision and may introduce ambiguity during instruction-conditioned policy learning. Second, we filter action chunks that contain an excessive number of abnormal actions with unusually large magnitudes. This step removes severe action artifacts while avoiding overly aggressive trajectory-level filtering.

Delta End-Effector Action Alignment. For delta end-effector actions, different datasets may be collected at different frame rates, while the physical robot motion speed is generally comparable across datasets. To reduce discrepancies caused by different FPS values, we rescale each delta action by the corresponding FPS and convert it to a unified per-second representation. An action step is marked as invalid if its absolute translational velocity exceeds 0.5 or its absolute rotational velocity exceeds 1.0.

Instead of discarding an entire trajectory whenever abnormal actions appear, we adopt a step-level masking strategy. Specifically, invalid action steps are masked out during training, so that a small number of noisy actions does not remove an otherwise useful trajectory. For each action chunk $\mathcal{C}$, we compute the invalid-step ratio:

$$r_{\mathcal{C}} = \frac{1}{|\mathcal{C}|} \sum_{t \in \mathcal{C}} \mathbb{I}\left[\text{step } t \text{ is invalid}\right].$$

We discard the whole chunk only when $r_{\mathcal{C}} > 0.5$, since such chunks are dominated by unreliable supervision signals. After filtering, we normalize the translational and rotational components by 0.5 and 1.0, respectively. This maps delta end-effector actions from different datasets into a shared and numerically stable action space.

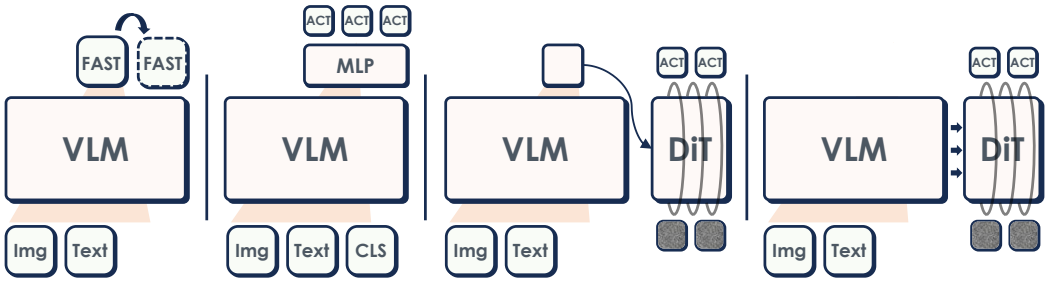


Figure 9: Comparison of representative VLA action heads studied in this work. FAST preserves the VLM's autoregressive interface by discretizing action chunks into action tokens. OFT attaches a lightweight MLP head to VLM action features and predicts continuous actions in parallel. PI and GR00T use a separate DiT-style motor module conditioned on VLM features to generate continuous action chunks through flow matching, with GR00T using a more decoupled dual-system design.

Absolute Joint-Angle Action Alignment. For absolute joint-angle actions, we first remove values outside the range $[-2\pi, 2\pi]$, as such values are likely caused by data collection errors or preprocessing artifacts. For the remaining valid actions, we wrap each joint angle into the range $[-\pi, \pi]$:

$$\theta_{\text{wrap}} = ((\theta + \pi) \bmod 2\pi) - \pi.$$

This wrapping operation resolves equivalent angular representations and ensures that all joint-angle actions follow a consistent convention. After this step, joint-angle actions from different datasets naturally lie in the same angular space, and therefore we do not apply additional normalization to them.

Gripper Action Normalization. For gripper actions on both robot platforms, we first remove extreme outliers and then apply min-max normalization separately for each dataset. This maps gripper values to continuous values in $[0, 1]$, providing a consistent representation across datasets despite differences in gripper hardware, control interfaces, and recording conventions.

I Details of Action Heads

I.1 Representative Action-Head Families

We briefly summarize the four representative action-head families used in our study. They share the same high-level role: mapping VLM representations to robot action chunks. Their main difference lies in the action interface: FAST uses discrete autoregressive action tokens, OFT uses direct continuous regression, and PI/GR00T use flow-matching-based continuous action generation.

FAST: autoregressive discrete action tokens. FAST converts a continuous action chunk into a sequence of discrete action tokens. Let

$$z_{1:M} = \text{Tok}_{\text{FAST}}(A_t)$$

denote the token sequence obtained from the continuous action chunk. The VLA model is trained with the standard next-token prediction objective

$$\mathcal{L}_{\text{FAST}} = -\sum_{m=1}^{M} \log p_\theta(z_m \mid H_t, z_{<m}).$$

At inference time, the model autoregressively generates action tokens $\widehat{z}_{1:M}$ and maps them back to continuous actions through the inverse tokenizer,

$$\widehat{A}_t = \text{Tok}_{\text{FAST}}^{-1}(\widehat{z}_{1:M}).$$

FAST therefore preserves the native language-modeling form of the VLM: action prediction is reduced to token prediction. Compared with direct per-dimension action binning, FAST-style tokenization compresses the action trajectory before discretization, often by representing action chunks in a frequency-domain or otherwise compact basis. Its advantage is architectural compatibility

with autoregressive VLMs; its cost is that continuous control is mediated through a discrete tokenizer and decoded sequentially.

OFT: parallel continuous regression. OFT removes the discrete tokenizer and attaches a lightweight continuous regression head to the VLM. The model appends a fixed set of action query tokens to the backbone sequence. If $h_{t,k}^{\text{act}}$ denotes the hidden state corresponding to the k-th action query, then a small MLP predicts the corresponding action vector:

$$\widehat{a}_{t+k} = g_\phi(h_{t,k}^{\text{act}}), \qquad k = 0, \ldots, K-1.$$

The head is trained with a pointwise regression loss, typically

$$\mathcal{L}_{\text{OFT}} = \frac{1}{K d_a} \sum_{k=0}^{K-1} \left\| g_\phi(h_{t,k}^{\text{act}}) - a_{t+k} \right\|_1 .$$

Unlike FAST, OFT predicts the entire action chunk in one forward pass. This makes OFT a simple and efficient probe of whether the backbone representation linearly exposes the information needed for continuous control. The tradeoff is that the output is a point estimate rather than an explicit multimodal action distribution.

PI: flow-matching action expert. PI-style heads model continuous action chunks through conditional flow matching. Instead of directly predicting A_t, the head learns a vector field that transports Gaussian noise into the demonstrated action chunk conditional on the backbone representation. Let $\epsilon \sim \mathcal{N}(0, I)$ and sample a flow time $\tau \in [0, 1]$. With a linear probability path,

$$A_t^\tau = (1-\tau)\epsilon + \tau A_t,$$

the target velocity is

$$u_t = A_t - \epsilon.$$

The action expert v_ϕ receives the noised action chunk, the flow time, and the backbone representation, and is trained by

$$\mathcal{L}_{\text{PI}} = \mathbb{E}_{A_t, \epsilon, \tau} \left[\left\| v_\phi(A_t^\tau, \tau; H_t) - (A_t - \epsilon) \right\|_2^2 \right].$$

At inference time, the head starts from Gaussian noise $A_t^0 \sim \mathcal{N}(0, I)$ and integrates the learned vector field:

$$A_t^{n+1} = A_t^n + \Delta\, v_\phi(A_t^n, \tau_n; H_t), \qquad n = 0, \ldots, N-1,$$

returning $\widehat{A}_t = A_t^N$. Compared with OFT, PI introduces an iterative generative decoder. This is more expensive than one-shot regression, but it can represent richer continuous action distributions and separates semantic conditioning from motor generation more explicitly.

GR00T: dual-system flow-matching motor module. GR00T also uses flow matching, but places it inside a more explicitly separated dual-system architecture. The vision-language backbone acts as a slow semantic reasoning module, while a DiT-style action module acts as a fast motor-generation module. The backbone encodes images and language into vision-language tokens. These tokens, together with proprioceptive state encodings and noised action encodings, condition a diffusion-transformer action module.

Formally, the training objective has the same flow-matching form:

$$\mathcal{L}_{\text{GR00T}} = \mathbb{E}_{A_t, \epsilon, \tau} \left[\left\| v_\phi(A_t^\tau, \tau; H_t, s_t, e) - (A_t - \epsilon) \right\|_2^2 \right],$$

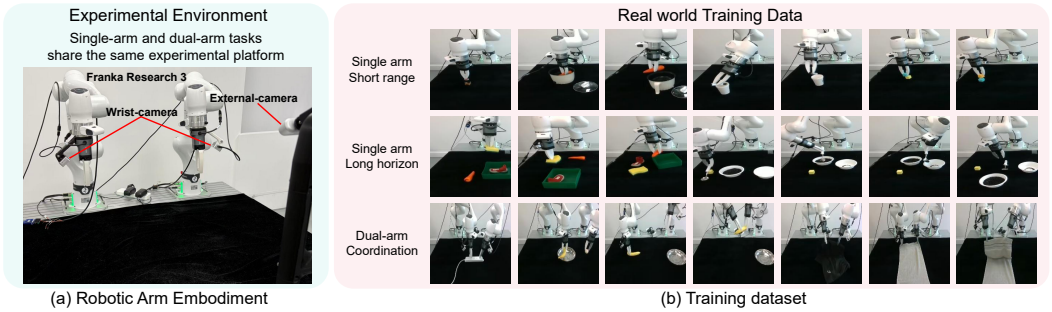
where s_t denotes the robot state and e denotes an embodiment identifier or embodiment-specific adapter. The key distinction from PI is architectural: state and action vectors are embedded into a separate motor module, processed by a DiT-style transformer, and decoded back into the robot's native action space. Cross-attention to the VLM tokens keeps the motor module conditioned on the scene and instruction while preserving a specialized action-generation pathway.

I.2 Unified Action Representation

Although FAST, OFT, PI, and GR00T use different decoding mechanisms, they all supervise a future action chunk. In our multi-embodiment pre-training, we therefore map each dataset's native action format into a shared canonical output space before applying each head's objective.

Table 12: **Effect of unified action representation design.** We compare separate heads, a unified head, and the full unified action representation on RoboTwin and LIBERO-Plus. Dark blue highlights the full design; light blue highlights the unified-head intermediate.

Setting	RoboTwin	LIBERO-Plus
Separate heads	78.5	81.1
Unified head	79.5	81.4
Unified action representation	**80.5**	**82.6**



(a) Robotic Arm Embodiment (b) Training dataset

Figure 10: Overviews of our robot setting and real-world training dataset. (a) Robot setting for the experiments. (b) We collect real-world robot trajectories for eleven task suites.

Design For the continuous heads used in our main pre-training recipe, the action head emits a fixed 20-dimensional vector at each action step,

$$\widehat{a}_{t+k} \in \mathbb{R}^{20}, \qquad k = 0, \ldots, K-1.$$

The dimensions are assigned by embodiment role:

- Dimensions 1–6: the left 6-DoF arm of bimanual embodiments such as AgileX, represented as absolute joint angles.
- Dimensions 7–12: the right 6-DoF arm of bimanual embodiments such as AgileX, represented as absolute joint angles.
- Dimensions 13–18: the 6-DoF arm of single-arm embodiments such as Franka, represented as delta end-effector pose.
- Dimension 19: the shared gripper coordinate, used by Franka's single gripper and AgileX's left gripper.
- Dimension 20: the right gripper coordinate of bimanual embodiments.

This layout preserves each corpus's native low-level action convention: Franka actions use delta end-effector pose in DROID and MolmoAct, while AgileX actions use absolute joint angles in InternData-A1 and RoboCoin. Thus, the unified head design does not force all embodiments into the same controller parameterization. It only aligns output coordinates at the level of embodiment role, allowing the same FAST/OFT/PI/GR00T head families to be instantiated on a common cross-embodiment action representation.

Experiments We evaluate the effect of the unified action representation design under three settings: separate embodiment-specific heads, a unified head without action space alignment, and a unified head with unified action representation. Experiments are conducted on RoboTwin and LIBERO-Plus. As shown in Table 12, simply sharing the action head already improves performance over separate heads, while the fully unified action representation achieves the best results on both benchmarks. This suggests that aligning outputs by embodiment role provide a more reusable and transferable action representation across robots.

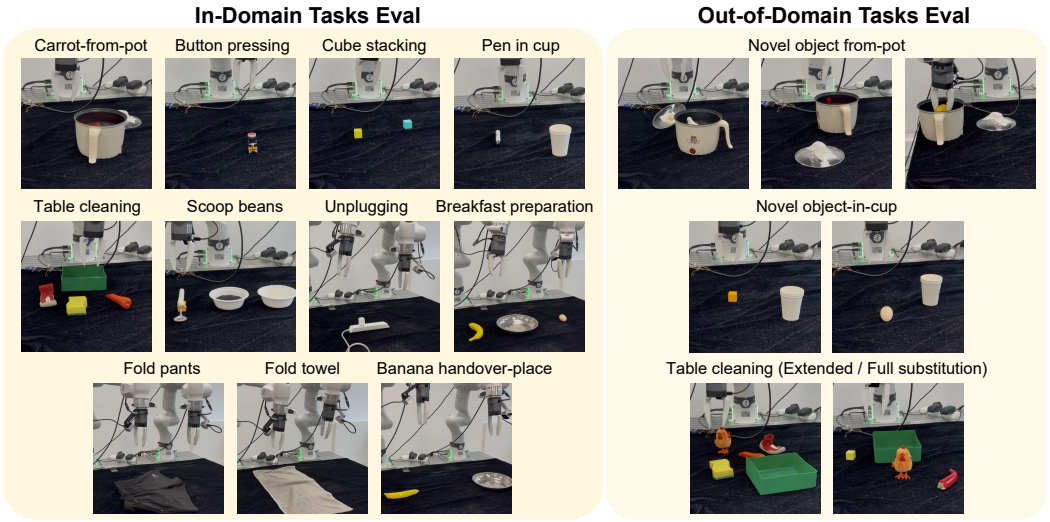


Figure 11: Schematic illustrations of real-robot in-domain and out-of-domain evaluation tasks.

J Details of Real-world Experiments

J.1 Hardware and observation.

As shown in Fig. 10, our physical evaluation platform uses stationary, table-mounted Franka Research 3 7-DoF arms (one for single-arm setups, two for dual-arm tasks). The visual system is a multi-view configuration: a single Intel RealSense D435 is fixed as an external camera providing a third-person perspective, and each robot arm carries a wrist-mounted Intel RealSense D405 for precise first-person observations. In line with the input specification of our representation-centric pre-training recipe, all images from external and wrist cameras are resized to 224×224 before being processed by the model.

J.2 Expert Data Collection and Fine-tuning Protocol

Expert data collection. All real-world demonstrations are collected by human teleoperation using a GELLO [93] interface. During collection, we randomize the initial placements of the manipulated objects and the initial configuration of the robot arms across episodes, so that the demonstrations cover a range of starting states rather than repeating a single scripted trajectory. Fig. 10(b) shows the eleven task suites for which we collect data.

Fine-tuning data. Each single-arm task uses 50 demonstrations, and each dual-arm task uses 100 demonstrations, reflecting the higher coordination complexity of bimanual manipulation. VLAct and the Qwen3VL-4B-OFT baseline are fine-tuned on *exactly the same* demonstrations, under the same action head, optimizer, and training budget, so that the comparison isolates the pretrained backbone rather than the downstream data or recipe.

Evaluation protocol. For each task, both models are evaluated from the same 10 fixed robot and object initial configurations, held constant across models and across runs. These are the 10 rollouts per task used in the success-rate computation described below. Because the initial states are fixed and shared, differences in success rate reflect policy behavior rather than variation in starting conditions.

J.3 Detailed Real-World Results and Failure Analysis

Success Rate Calculation Criterion. For real-machine experiments, each task is rolled out for 10 trajectories to calculate the task success rate. The scoring rules vary for different task types. Specifically, for single-arm short-range tasks and dual-arm collaborative tasks, a score of 1 is awarded for a successful task completion and 0 for a failure. For single-arm long-range tasks, the scoring mechanism is based on the number of completed task steps. Taking the table cleaning task as an example, the entire task consists of three independent steps, and each successfully completed step contributes 0.33 points to the total task score.

Single-arm short-horizon tasks. VLAct achieves an average success rate of 92.5% on in-domain single-arm short-horizon tasks, compared with 77.5% for Qwen3VL-4B-OFT without pre-training. Both models perform well on simpler tasks such as *carrot-from-pot*, where both reach 100% SR, and *button pressing*, where VLAct reaches 100% SR and the baseline reaches 90% SR. However, the gap becomes larger when the task requires more precise spatial localization. On *cube stacking*, VLAct reaches 90% SR while the baseline drops to 60%. We observe that the baseline often freezes during inference when the cube starts from positions not well covered by the training distribution. On *pen-in-cup*, VLAct reaches 80% SR compared with 60% for the baseline. VLAct can localize the cup even when it is placed near the rear of the table, while the baseline often accumulates errors after failed grasping attempts and enters unstable recovery behaviors.

Short-horizon OOD generalization. VLAct also generalizes better to unseen objects. For *novel object from-pot*, where the carrot is replaced by egg, pepper, or garlic, VLAct obtains 90.0% average SR, while the baseline reaches 73.3%. For *novel object in-cup*, where the target object is replaced by cube or egg, VLAct again reaches 90.0% average SR, compared with 65.0% for the baseline. These results indicate that the pre-trained backbone provides stronger visual-semantic grounding and more stable localization under object-level distribution shift.

Single-arm long-horizon tasks. For long-horizon tasks, VLAct shows a clearer advantage in preserving task logic over multiple manipulation stages. On *table cleaning*, VLAct obtains a weighted success rate 86.6%, compared with 73.3% for the baseline. On *scooping beans*, VLAct reaches 80.0%, whereas the baseline only reaches 33.3%. In this task, VLAct usually completes the intended sequence: grasping the spoon, scooping a bean, and pouring it into the target bowl. By contrast, the baseline frequently skips the scooping phase and directly performs a pouring motion with an empty spoon. In cluttered table-cleaning scenes, the baseline also tends to overlook small objects, while VLAct completes the sequence more fluently.

Long-horizon OOD generalization. We further evaluate long-horizon robustness under two harder OOD settings. In the extended-sequence setting, a novel toy chicken is added to the original table cleaning task. VLAct achieves a weighted success rate of 82.5%, nearly doubling the baseline score of 47.5%. In the full-substitution setting, all objects are replaced by unseen categories, including cube, toy chicken, and pepper. VLAct maintains a score of 83.3%, while the baseline drops to 46.6%. The baseline often misses objects or fails to recover from grasping errors, whereas VLAct maintains the task sequence under both object replacement and sequence extension.

Dual-arm coordination. Across dual-arm coordination tasks, VLAct achieves 72.0% average SR, outperforming the baseline's 44.0% SR. On *breakfast preparation*, VLAct reaches 90.0% SR compared with 70.0% for the baseline. On *unplugging*, VLAct reaches 80.0% SR compared with 60.0% for the baseline, showing better coordination in the hold-and-pull interaction. The largest gaps appear in deformable-object manipulation. On *fold pants*, VLAct achieves 70.0% SR, outperforming the baseline by 30.0%. The baseline often selects inaccurate grasping points on the fabric, leading to gripper-table collisions or failed lifting. On *fold towel*, VLAct reaches 50.0% SR, while the baseline reaches only 20.0%. The baseline frequently suffers from velocity mismatch between the two arms, causing the towel to slip from one gripper. VLAct produces more synchronized dual-arm motion and more stable contact with deformable objects.

J.4 Detailed Prompts for Each Real-World Robotic Arm Task

Single-arm (Short-horizon)

- **Carrot-from-pot:** Pick the carrot out of the pot and place it on the table.
- **Button pressing:** Press the red button.
- **Cube stacking:** Pick up one cube and stack it on top of another cube.
- **Pen in cup:** Pick up the pen and place it into the paper cup.

Single-arm (Long-horizon)

- **Table cleaning:** Pick up the small objects on the table one by one and place them into the green box.
- **Scoop beans:** First pick up the spoon, then use it to scoop a bean from one bowl, and finally pour the bean into another bowl.

Dual-arm Coordination

- **Unplugging:** One robotic arm holds the socket in place, while the other pulls the plug out and places it on the table.
- **Breakfast preparation:** One robotic arm picks up the banana, while the other picks up the egg; both are then placed on the plate.
- **Banana handover-place:** One robotic arm picks up the banana and hands it to the other arm; the second arm receives it and places it on the plate.
- **Fold pants:** Two robotic arms fold a pair of pants together.
- **Fold towel:** Two robotic arms fold a towel together.

Out-of-Domain (OOD)

- **Novel object from-pot:** Pick the garlic / pepper / egg out of the pot and place it on the table.
- **Novel object in-cup:** Pick up the cube / egg and place it into the paper cup.
- **Table cleaning (extended / full substitution):** Pick up the small objects on the table one by one and place them into the green box.

J.5 Visualization

Detailed Visualization of Training Data for Each Real-World Robotic Arm Task. Fig. 12 shows the visualization of training data for each real-world robotic arm task.

Detailed Visualization of Inference Demos for Each Real-World Robotic Arm Task. Fig. 13 shows inference demos for the single-arm short-horizon tasks, both in-domain and out-of-domain.

Detailed Visualization of Inference Demos for Each Real-World Robotic Arm Task. Fig. 14 shows inference demos for the single-arm long-horizon and dual-arm coordination tasks, both in-domain and out-of-domain.

K Future Work and Broader Impact.

Limitations. Due to resource constraints, our study focuses on a 4B-scale VLM backbone and does not explore substantially larger models. Larger VLMs may contain stronger visual, spatial, and language priors, and the optimal pre-training recipe may change with model scale.

Broader Impact. We hope this work encourages the community to pay more attention to how VLM backbones should be trained for VLA tasks, rather than treating the backbone as a fixed component inherited from general vision-language pre-training. Future work can study how to build VLA-oriented VLMs that could better generalize across tasks, environments, and embodiments. More broadly, improving the training of reusable VLA backbones may make robot learning less dependent on massive proprietary robot datasets, helping more researchers build generalizable VLA models with accessible data and compute.

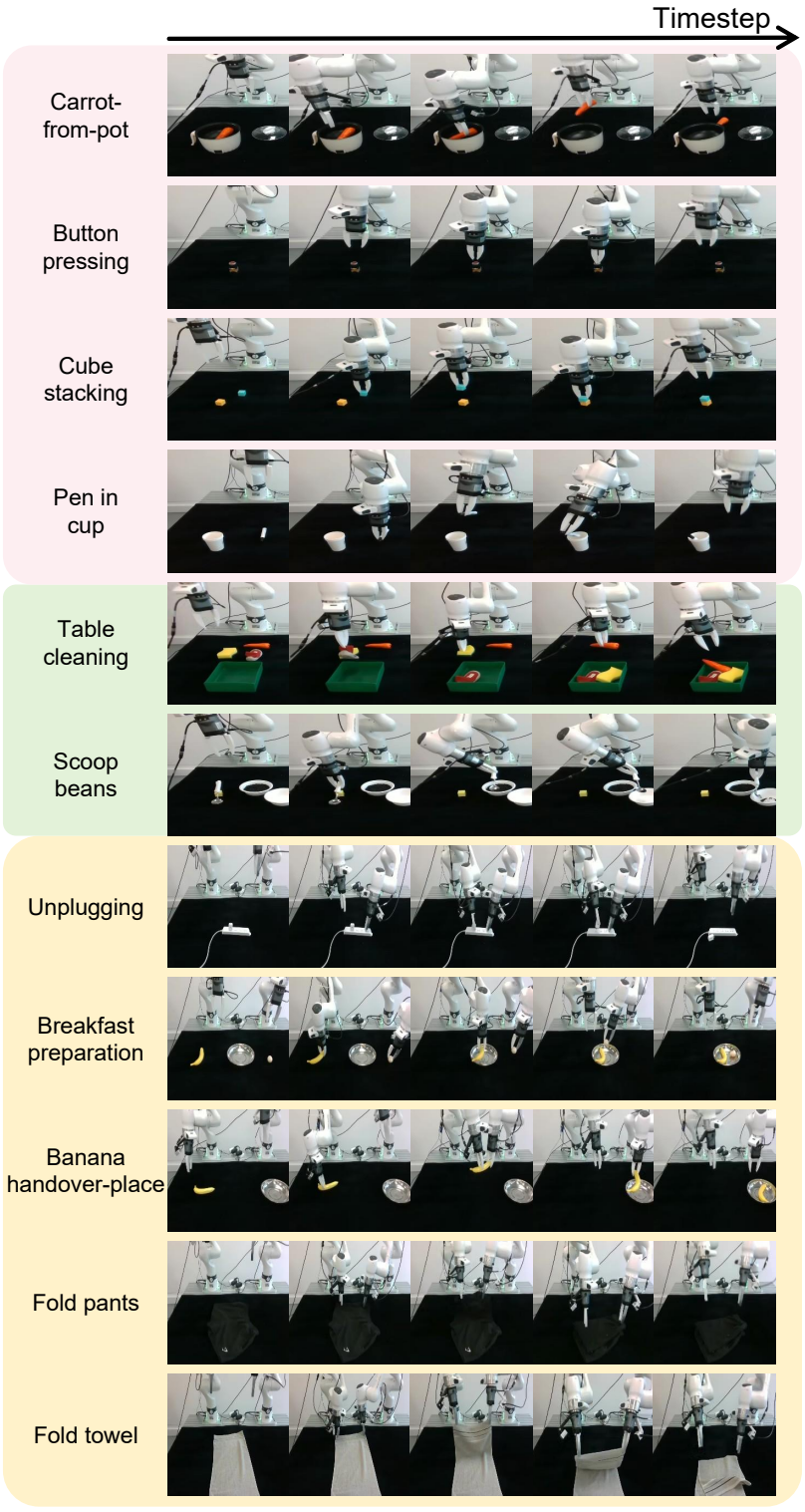


Figure 12: **Detailed Visualization of Training Data for Each Real-World Robotic Arm Task.**

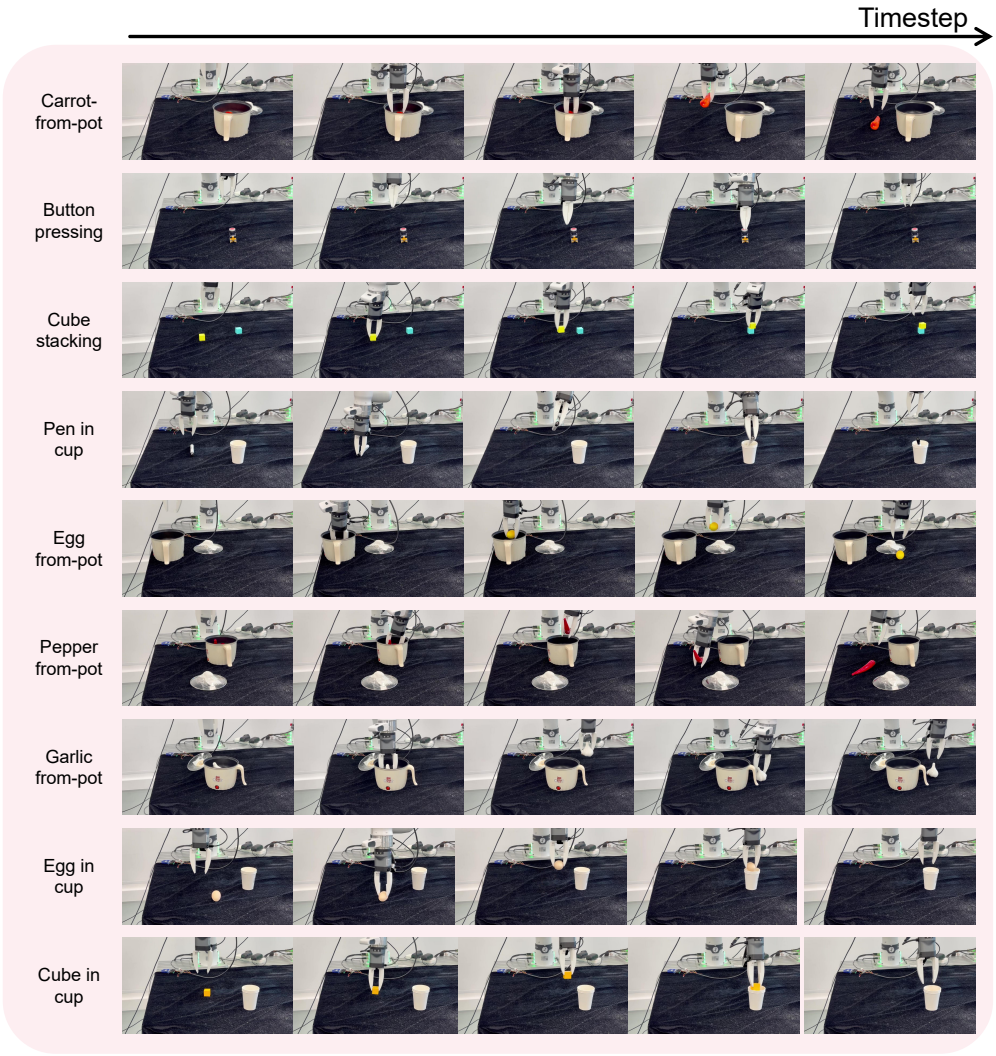


Figure 13: **Detailed visualization for Single arm short range real-world robotic arm task inference demo** We show the inference visualization for Single arm short range real-world task, including In-Domain and Out-of-Domain tasks.

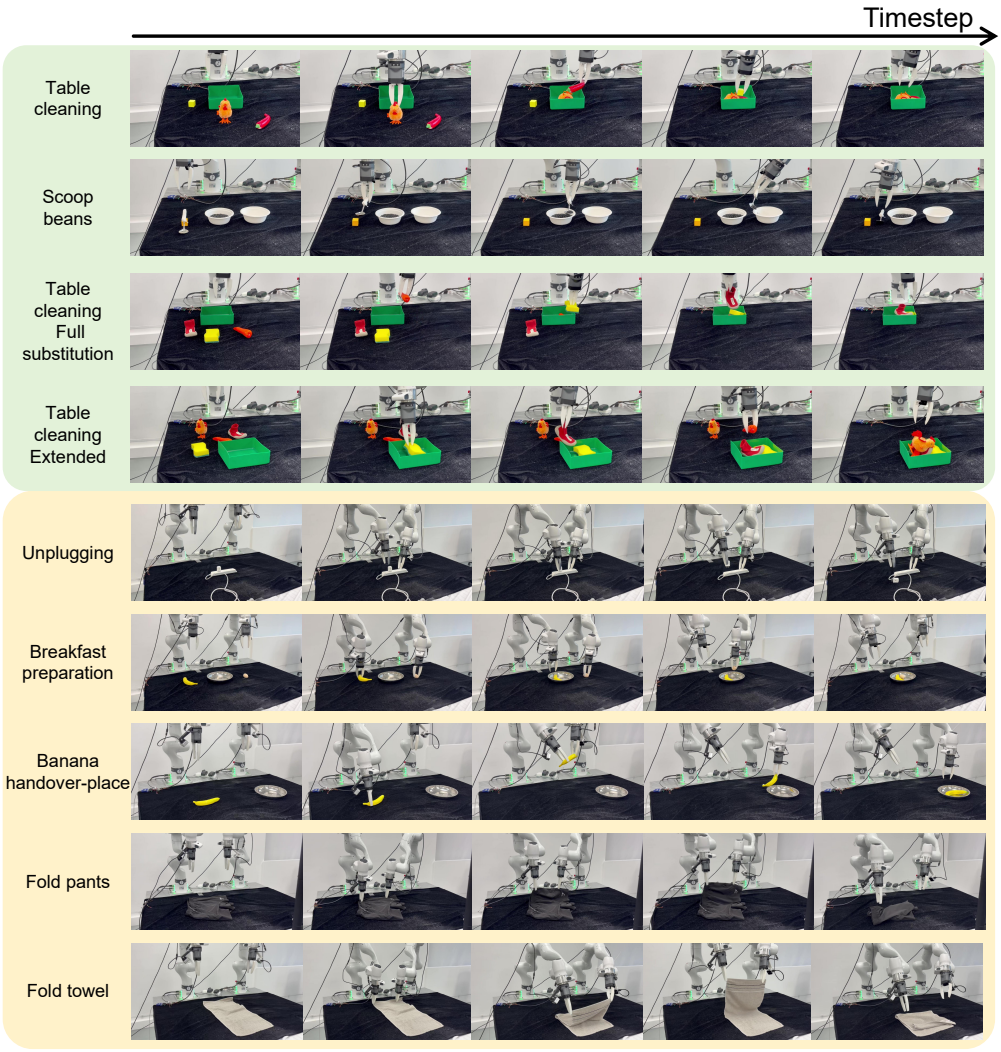


Figure 14: **Detailed visualization for Single arm long horizon and Dual arm coordination real-world robotic arm task inference demo** We show the inference visualization for Single arm long horizon and Dual arm coordination real-world task, including In-Domain and Out-of-Domain tasks.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The claims have been validated with empirical results.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss limitations in this paper.

   Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: We do not have theoretical results.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide experiment settings and details to reproduce our results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.

- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We use publicly available data and the code will be publicly available.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: We provide experiment settings and details.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: The pre-training is expensive, hence we do not report the error bars.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: The compute resources are reported.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: The research follows the NeurIPS Code of Ethics.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: The paper discuss broader impacts

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A].

Justification: Our work does not involve releasing data or models that have a high risk for misuse.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.

- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The datasets and models used are all properly cited.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: There is no new assets in this paper.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This paper does not include study participants.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: N/A

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.