RL-Pretrained Action Experts for Vision-Language-Action Models: Towards Physical Priors in Diffusion Policy Initialization

Matthieu Scharffe

matthieu.scharffe@gmail.com

Abstract

Vision-Language-Action (VLA) models are promising as they allow to interact with a robot through natural language and are very generalist policies. VLAs attach a randomly-initialized Diffusion Transformer (DiT) to a pretrained Vision-Language Model (VLM) and train it via supervised fine-tuning (SFT) on real or synthetic demonstration data. In parallel, the development of modern physics simulator (MuJoCo, Isaac Lab) combined with reinforcement learning (RL) has produced highly capable control policies especially for the locomotion of very complex robots like humanoids or dexterous manipulation. One of the main bottleneck of physical AI is the lack of data. The main advantage of RL training is that it can be run inside a simulator. I propose to use RL to pretrain a VLA in a simulator to let it build a physical knowledge a priori. Unlike recent work that RL-pretrains the action expert on a single downstream manipulation task, the physical prior I target is task-agnostic: it is trained once against a generic physics reward and transfers across tasks and embodiments, including classical whole-body skills such as walking. To do so, it requires to initialize the DiT. I show that this is feasible with two established architectures: a causal transformer trained by standard PPO, or a diffusion policy trained by DPPO or DDiffPG.

1 Introduction

Vision-Language-Action (VLA) models are emerging as the most promising candidate for a generalist robotic policy. By attaching an action decoder to a pretrained Vision-Language Model (VLM), they inherit the semantic generalization of internet-scale pretraining and expose it to a robot through a natural-language interface. The Physical Intelligence π family [1–4], NVIDIA GR00T [5], SmolVLA [6] and X-VLA [7] all share this template and are pushing the next frontier of physical AI.

In parallel, model-free reinforcement learning in modern physics simulators (MuJoCo [8], Isaac Lab [9]) has produced a second generation of controllers with capabilities that no purely supervised pipeline has matched. Humanoids walk over miles of trails [10, 11], and the viral videos of humanoids dancing or performing martial-arts choreography are, almost without exception, the product of large-scale simulator RL. The training signal of an unmodified per-step reward, replayed across billions of randomized simulator steps, is the substrate on which these exotic physical skills are built.

The two fields are largely separated. VLAs are trained by supervised fine-tuning (SFT) on demonstration data; their action decoders are randomly initialized at the start of training. RL controllers are trained from environment reward in simulation; they have no language or vision interface. A recent NVIDIA effort, SONIC [12], partially bridges the two by training a separate whole-body RL motion-tracking policy on 700 hours of motion-capture data and routing commands from a fine-tuned GR00T N1.5 VLA through it via a shared token interface - a System-1 / System-2 split that requires two trained models, virtual reality (VR) teleoperation hardware, and a large mocap corpus. This work proposes a different bridge: rather than maintaining the split, instill a physical sense directly into the VLA's own action-expert weights via simulator RL, so a single model is trained end-to-end and downstream task-specific demonstrations are the only data required. The two approaches are complementary on the data axis - VR-teleoperated trajectories collected through a SONIC-style pipeline can in turn be reused as SFT data for an RL-pretrained VLA - but architecturally

distinct.

Contribution. I propose RL pretraining of the VLA action expert: a transfer-learning recipe that uses a simulator-trained RL control policy as the initialization of the action expert, before SFT on demonstration data. Recent work, ActionX [13], has shown the empirical value of this ordering - RL-pretraining a flow-matching action expert with a frozen VLM beats both supervised pretraining and no pretraining - but confines it to a single downstream manipulation task, pretraining the action expert in that task's own simulator under its task-success reward, on a single embodiment. The contribution here is orthogonal and more general: the RL stage instills a task-agnostic physical prior - trained once against a generic physics reward, covering classical whole-body skills such as locomotion, and transferred across many downstream tasks and different embodiments rather than re-paid per task. I identify two pretraining paradigms grounded in established work: a Paradigm 1 causal-transformer route via standard Proximal Policy Optimization (PPO) [14], and a Paradigm 2 diffusion-policy route via Diffusion Policy Policy Optimization (DPPO) [15] or Deep Diffusion Policy Gradient (DDiffPG) [16]. I then describe how the result transfers into a DiT-style [17] VLA action expert.

The walking gap illustrates the opportunity. Current VLAs walk poorly, if at all, while a 1.4M-parameter RL-trained transformer walks zero-shot over rough terrain [11]. The VLA action expert and the RL transformer are close architectural cousins. The community has built every piece of the bridge - the simulators, the architectures, the algorithms - and has not yet crossed it in the direction proposed here.

2 Background

2.1 Vision-Language-Action model architectures

A modern VLA decomposes into three modules:

1. A pretrained VLM backbone that consumes the images (or videos) and the language instruction and emits a sequence of token embeddings.
2. An action expert, architecturally a Diffusion Transformer (DiT) [17], that consumes (noised) action tokens together with the proprioceptive state. The VLM token embeddings are routed into the DiT through its cross-attention layers.
3. An embodiment-specific projection MLPs that map the robot's heterogeneous state and action vectors to and from a shared latent dimension.

Notably, the Physical Intelligence π family [1], [2], [3], [4], the NVIDIA GR00T model family [5] and SmolVLA [6] fused the VLM to the DiT, through the cross-attention mechanisms. X-VLA [7] uses another paradigm where the output of the VLM is directly fed into the input of its DiT.

VLAs are typically trained through behavior cloning. Reusing the framing of [7], they are fine-tuned on expert trajectories $\mathcal{D} = \{\tau_j\}_{j=1}^{M}$, $\tau_j = \{(o_n, a_n)\}_{n=1}^{N_j}$, where o_n denotes the multimodal observation at step n (e.g., visual input, language instruction, proprioceptive state) and a_n is the corresponding expert action. The policy $\pi_\theta(o_n)$ parameterized by θ is optimized to predict the demonstrated action chunk $A_n = [a_n, a_{n+1}, \ldots, a_{n+T-1}]^\top$, where T denotes the chunk size, by minimizing a suitable supervised loss $\ell(\cdot)$:

$$\mathcal{L}_{BC}(\theta) = \mathbb{E}_{(o_n, A_n)\sim\mathcal{D}} \left[\ell \left(\pi_\theta(o_n), A_n \right) \right]. \tag{1}$$

2.2 Reinforcement Learning control policy

Modern physics simulators (MuJoCo [8], Isaac Lab [9]) allow a robot designed in CAD to be brought into a parallelized GPU simulation where a neural-network control policy is trained to maximize a hand-designed reward function. Reward shaping remains a craft, but the resulting policies can transfer zero-shot to real hardware on quadrupeds, bipeds, and dexterous hands.

RL algorithms split into two families. On-policy methods estimate the gradient of expected return from rollouts collected by the current policy and discard them after one update; Proximal Policy Optimization (PPO) [14] is the dominant choice for simulator-scale training because it is stable and parallelizes well. Off-policy methods learn a value function (Q-learning) from a replay buffer of past transitions, which is more sample-efficient but harder to stabilize at scale; DDiffPG [16] relies on off-policy actor-critic learning to drive a diffusion policy.

Recent works from Berkeley [10, 11] use a small causal transformer trained by PPO to build a locomotion policy for Agility's Digit humanoid robot, exhibiting zero-shot walking over complex outdoor environments.

2.3 Diffusion policies and their RL training

A Diffusion Policy [18] parameterizes the action distribution as the reverse of a forward noising process: starting from Gaussian noise, the policy iteratively denoises an action chunk conditioned on the current observation. Flow matching [19] is the continuous-time, deterministic-ODE special case of this framework, and it is the variant used by the modern VLA action experts; any training recipe that applies to a diffusion policy therefore applies to a flow-matching action expert.

Two recent works lift the training of diffusion policies to RL. DPPO [15] treats the chain of denoising steps as an additional inner MDP nested inside the environment MDP, yielding a tractable Gaussian likelihood at each denoising step that PPO can optimize. DDiffPG [16] trains a diffusion policy from scratch with off-policy actor-critic updates and exploration bonuses. The takeaway is that a diffusion policy with the same architecture as a VLA action expert can be trained from environment reward alone.

3 Problem Formulation

3.1 POMDP formulation

We treat VLA control as a Partially Observed Markov Decision Process $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{O}, P, R, \gamma)$ with hidden state $s \in \mathcal{S}$, action $a \in \mathcal{A}$, observation $o \in \mathcal{O}$, transition $P(s' \mid s, a)$ given by the simulator or real robot, and reward $R(s, a)$. The policy $\pi_\theta(a_{t:t+H-1} \mid o_{t-T+1:t}, \ell)$ outputs an action chunk of horizon H given an observation history of length T and a language instruction ℓ, and is parameterized as a transformer-based flow-matching or diffusion model.

3.2 The initialization problem

Let $\theta = (\theta_{\text{VLM}}, \theta_{\text{AE}}, \theta_{\text{embod}})$ partition the VLA parameters into the VLM backbone, the action-expert trunk, and the embodiment-specific projection MLPs. Standard practice initializes $\theta^{(0)}_{\text{VLM}} = \theta^{\text{web}}_{\text{VLM}}$ from a web-scale pretrained checkpoint, and draws $\theta^{(0)}_{\text{AE}}, \theta^{(0)}_{\text{embod}}$ from a random initializer. SFT then minimizes a flow-matching loss $\mathcal{L}_{\text{fm}}(\theta; \mathcal{D}_{\text{demo}})$ on a demonstration dataset.

The bottleneck is that $\theta^{(0)}_{\text{AE}}$ encodes no inductive bias about actuation. The flow-matching loss must therefore use demonstrations to teach the action expert, simultaneously, two things:

1. how to denoise toward a clean action distribution;
2. the basic dynamics of the embodiment.

Demonstrations are scarce, expensive, and biased toward expert behavior; recovery from external disturbances and the resulting policy robustness are weakly represented in them. RL in simulation has, for nearly a decade, been the substrate that solves the second point. This paper proposes to leverage that substrate during the pretraining of the VLA.

Why this is the bottleneck. The VLM brings semantic generalization; demonstrations bring task-specific knowledge; nothing upstream brings general physics understanding. The walking gap illustrates this: the VLAs surveyed in this paper either cannot walk, walk only slowly in controlled environments, or walk via a separately-trained low-level controller, while a 1.4M-parameter RL-trained transformer walks Digit zero-shot over rough terrain. The disparity is not in the architecture - the architectures are in the same family - but in the training signal seen by the action-expert weights.

4 Proposed Framework

4.1 Paradigm 1 - Causal transformer via standard PPO

Architecture. A small causal transformer in the same family as the target VLA's action-expert trunk. Following the Berkeley humanoid controllers HT and HT-2 [10, 11], the policy $\pi_\theta(a_t \mid o_{t-T+1:t}, a_{t-T+1:t-1})$ takes a context window of the past T proprioceptive observations and actions. Each observation and each past action is embedded by a MLP into the transformer's hidden dimension; the resulting token sequence is processed by a stack of causal self-attention blocks; an output MLP head decodes the final token into an action token. HT-2 is a concrete instance - 4 transformer

blocks, hidden dim 192, 4 heads, MLP ratio 2, context window 16, 1.4M parameters total - but the recipe is agnostic to the exact depth, width, and head count: the practitioner should match these to the shape of the downstream action expert.

Training signal. PPO [14] optimizes the clipped surrogate objective

$$\mathcal{L}^{\text{PPO}}(\theta) = \mathbb{E}_t\Big[\min\Big(r_t(\theta)\,\hat{A}_t,\ \text{clip}(r_t(\theta), 1{-}\epsilon, 1{+}\epsilon)\,\hat{A}_t\Big)\Big], \quad r_t(\theta) = \frac{\pi_\theta(a_t \mid o_{t-T+1:t})}{\pi_{\theta_{\text{old}}}(a_t \mid o_{t-T+1:t})}, \tag{2}$$

where $\hat{A}_t$ is an advantage estimate computed from the unmodified per-step environment reward (typically via GAE) and the clip range ϵ bounds the per-update policy change. This is the regime in which simulator-scale RL has been extensively explored on quadrupeds, bipeds, and humanoids, and in which HT-2 walked Digit zero-shot over rough terrain. The clean reward signal and the small parameter count combine to produce a stable, sample-efficient training loop.

Conversion to a VLA action expert. The Paradigm-1 model is a causal transformer; the VLA action expert is a Diffusion Transformer [17]. The conversion has three steps:

1. **Insert adaLN.** Each transformer block is wrapped with adaptive layer norm whose scale and shift are regressed from the flow-matching timestep embedding, following the DiT recipe [17]. The residual scale is initialized to zero (adaLN-Zero) so that, at the start of SFT, each block acts as the identity over the timestep input and preserves the RL-pretrained forward function.

2. **Replace input/output projections.** The Paradigm-1 input embedder is replaced by the VLA's embodiment-specific state-and-action embedder MLP, and the output head by the embodiment-specific action decoder MLP, sized to the new state and action dimensions. This is the cross-embodiment pattern established by GR00T N1 [5]: "embodiment-aware state and action encoder to embed the robot's state and action inputs."

3. **Convert some self-attentions to cross-attentions.** A subset of the trunk's self-attention sublayers is repurposed as cross-attention to the frozen VLM's token embeddings, mirroring the encoder-to-decoder cross-attention of the original Transformer [20] and the alternation of cross- and self-attention blocks used in GR00T N1 [5]. The VLM itself is untouched: only its output tokens are forwarded into the action expert.

4.2 Paradigm 2 - Diffusion policy via DPPO or DDiffPG

Architecture. A diffusion or flow-matching policy whose backbone is the same transformer-based stack used for the target VLA's action-expert trunk. The output is an action chunk $A_t = [a_t, \ldots, a_{t+H-1}]$. By construction, the architecture is the same as the VLA action expert minus the cross-attention to the VLM tokens, so weight transfer at the end of pretraining is direct.

4.2.1 DPPO - on-policy fine-tuning of a diffusion policy

DPPO [15] unrolls the denoising process into an inner MDP nested inside the environment MDP. The composite "Diffusion Policy MDP" $\mathcal{M}_{\text{DP}}$ has states $\bar{s}_{\bar{t}(t,k)} = (s_t, a_t^{k+1})$, actions $\bar{a}_{\bar{t}(t,k)} = a_t^k$, and reward

$$\bar{R}_{\bar{t}(t,k)}(\bar{s}, \bar{a}) = \begin{cases} 0 & k > 0, \\ R(s_t, a_t^0) & k = 0, \end{cases} \tag{3}$$

i.e. reward flows only at the fully-denoised step $k{=}0$. The Gaussian likelihood of each denoising step, $\pi_\theta(a_t^k \mid a_t^{k+1}, s_t) = \mathcal{N}(a_t^k; \mu(a_t^{k+1}, \varepsilon_\theta(a_t^{k+1}, k+1, s_t)), \sigma_k^2 I)$, admits exact policy-gradient updates. The original paper reports the following best practices when warm-starting from a demonstration-pretrained Diffusion Policy:

- fine-tune only the last K' denoising steps;
- run inference with Denoising Diffusion Implicit Models (DDIM);
- clamp the minimum exploration noise σ_k^{exp} to preserve exploration.

DPPO can also be trained from scratch, at the cost of roughly $6\times$ the wall-clock of a standard MLP policy trained by PPO. The architecture transfers directly to the VLA action expert.

4.2.2 DDiffPG - off-policy from-scratch training

DDiffPG [16] trains a diffusion policy from scratch, without demonstrations, using off-policy actor-critic learning. Its main contribution is to decouple exploration from policy learning: a Random Network Distillation bonus drives the agent toward novel states, the discovered trajectories are clustered into behavioral modes, and a separate Q-function is fit per mode. The diffusion policy is then updated against a per-mode target action obtained by gradient ascent on the corresponding Q-function. The recipe is the natural choice when demonstrations are unavailable for the pretraining task.

4.3 Transfer to the VLA action expert

Both paradigms produce a pretrained trunk θ_{AE} that initializes the action expert of a target VLA. The full assembly loads the VLM backbone θ_{VLM} from its public pretrained checkpoint, initializes θ_{AE} from the simulator-trained checkpoint - with the architectural conversion of §4.1 for Paradigm 1, or direct weight transfer for Paradigm 2 - and randomly initializes the embodiment MLPs θ_{embod} to fit the target embodiment's state and action dimensions. The VLM is typically left frozen during SFT, as in π_0 [1] and GR00T N1 [5].

For the action-expert SFT itself, two regimes are worth comparing empirically. Adapter-first freezes θ_{AE} and trains only θ_{embod} and the newly inserted layers (adaLN, cross-attentions) for a short warm-up before unfreezing the trunk; the goal is to protect the RL-pretrained trunk weights while the new layers settle. Joint unfreezes θ_{AE} together with the new layers from the first SFT step, which may converge faster on large demonstration sets at the risk of erasing some of the pretrained physics prior. Neither regime is a priori preferable - the choice is left to ablation.

5 Discussion

5.1 The walking gap

The clearest empirical case for the framework is the gap in physical capability between VLAs and dedicated RL controllers. The most explicit instance is the Unitree G1 demonstration of Luo et al. [12], in which a fine-tuned GR00T N1.5 emits upper-body teleoperation commands while a separate PPO-trained motion tracker handles the legs. More recent VLAs do exhibit some walking in their public demonstrations, but the wider gap between what these systems can do in controlled scenes and what RL-trained controllers do in the wild remains, and the capabilities of the action expert itself are bounded by what is present in the demonstration set, which heavily over-represents quasi-static manipulation. Simulator RL, in contrast, routinely produces behaviors that no demonstration set contains: Berkeley HT-2 walks Digit over 4 miles of trails and 31% city grades with a 1.4M-parameter RL-trained transformer that has no vision and no language input [11], and the viral footage of humanoids dancing or performing martial-arts choreography is the product of similar pipelines. The architectures are in the same family as the VLA action expert. The training signals differ. Pretraining the action expert via RL is a route to importing the more exotic physical capabilities of simulator RL into a VLA.

5.2 Distinction from RL fine-tuning of VLAs

A growing body of work applies RL to a VLA after SFT, in order to refine task success, suppress error accumulation under distribution shift, or improve reasoning. iRe-VLA [21] alternates online RL stages (with the VLM frozen and only lightweight action heads trained) with supervised stages on successful trajectories. ConRFT [22] fine-tunes a consistency-policy VLA via combined offline behavior cloning + Q-learning followed by online RL with human interventions. SimpleVLA-RL [23] scales VLA-tailored RL on top of OpenVLA-OFT and reports the emergence of unseen "pushcut" behaviors. π_{RL} [24] addresses the intractable action log-likelihood of flow-matching VLAs to enable online RL fine-tuning. VLA-RFT [25] replaces real environment rollouts with a learned world simulator. The shared template across all of these is SFT first, RL second.

The framework proposed here is the opposite ordering: RL before SFT, on a different domain (simulated physical control), so that SFT need only specialize a competent controller to the target task and embodiment rather than have to learn the underlying physics from scratch. The two approaches are complementary; an RL-pretrained VLA can in principle be post-trained with iRe-VLA-style or ConRFT-style RL after the SFT phase.

5.3 Distinction from task-specific RL pretraining

The most similar prior work is ActionX [13], which - like the present proposal - RL-pretrains a flow-matching action expert with a frozen VLM before SFT, and reports that this ordering beats both supervised pretraining and no pretraining;

on the LIBERO-Spatial subset, for instance, task success rises from 0% when a frozen-VLM backbone is adapted by SFT alone to 95% with RL pretraining. ActionX is thus the first empirical validation of the core ordering advocated here, and its wrist-orientation analysis - the RL-pretrained expert explores a markedly wider distribution of poses than the teleoperation demonstrations contain - directly supports the claim that RL instills physical behaviour the demonstration set never held.

Three properties nonetheless separate ActionX from the framework proposed here, and together they delimit its scope. First, the prior is task-specific: ActionX pretrains the action expert "with task-specific action distributions" inside the downstream task's own RL environment and under its sparse task-success reward, so the pretraining - and a bespoke task simulator and reward - must be re-paid for every new task. The present proposal instead pretrains against a generic physical reward once and amortizes that single task-agnostic physical prior across many downstream tasks. Second, the domain is manipulation only: every ActionX task is tabletop manipulation on LIBERO, Meta-World, or single/dual-arm robots, so no locomotion, balance, or whole-body skill is pretrained and the walking gap of §5.1 is left untouched. Third, the prior is single-embodiment: each ActionX action expert is pretrained for one embodiment, whereas the embodiment-specific projection MLPs of §4.3 are exactly what let one pretrained trunk transfer across different embodiments. ActionX names this generalization as open, calling for action-expert pretraining "similar to how infants learn to control their limbs before performing meaningful tasks, to develop general purpose action expert models for single or multiple embodied agents." The present framework is a concrete route to that goal.

5.4 Distinction from offline RL pretraining

Another closely related line is $\pi^*_{0.6}$ / RECAP [3]. RECAP uses offline RL to train a value-function VLM that emits a binary advantage label, which is then fed to the policy as an extra text input. The RL signal in RECAP therefore trains the value function; the action-expert weights themselves are still updated by a flow-matching (behavior cloning) loss on advantage-conditioned data, and at the start of training those weights are random. There are two distinctions with the proposed framework. First, the present proposal directs the RL signal at the action-expert weights themselves, before any demonstration data is presented. Second, RECAP relies on autonomous real-robot rollouts and human interventions, while the present proposal trains its RL signal inexpensively inside a physics simulator. The two methods are complementary: one could RL-pretrain the action expert via Paradigm 1 or 2 in simulation, then apply RECAP-style advantage conditioning on real-robot data during downstream SFT.

5.5 Limitations

The main practical limitation is action-space mismatch: the upstream RL controller may produce joint setpoints with PD gains at a fixed control frequency, while a downstream VLA may produce end-effector deltas, base velocities, or a heterogeneous mix at chunk frequencies. The embodiment-specific projection MLPs absorb this in principle, but the upstream RL run should be designed with the downstream action representation in mind. The choice of upstream pretraining task and reward function - locomotion, dexterous manipulation, contact-rich assembly - is itself an open design space that this paper deliberately leaves to follow-up empirical work.

A second open question is scale. Modern VLA action experts are close to a billion parameters (300M in π_0, 860M in $\pi_{0.6}$ and $\pi_{0.7}$), while the established RL-trained upstream policies discussed here are several orders of magnitude smaller (HT-2 at 1.4M, SONIC at up to 42M). Either the transferred weights occupy a small fraction of the action expert, or the upstream RL run has to be scaled up to match. Large-scale online PPO has been reported unstable on networks of that size [21], though recent simulator-RL results suggest the boundary is moving.

5.6 Future work

The natural validation is empirical. The framework spans a two-dimensional design space: (simulator, RL algorithm) on the upstream side and (VLM, embodiment, demonstration set) on the downstream side. Locomotion, dexterous manipulation, contact-rich assembly, and bimanual manipulation are all valid upstream pretraining tasks within either paradigm, and either MuJoCo or Isaac Lab can host them. The downstream side admits any combination of public VLM checkpoint and robot-data set.

6 Conclusion

VLA action experts are randomly initialized and trained by SFT on demonstration data. Reinforcement learning, in parallel, has produced controllers - some of which are causal transformers or diffusion policies in the same architectural

family as the VLA action expert itself - trained in simulators that the RL community already uses.

This paper proposed a transfer-learning framework that uses RL pretraining as the upstream initialization of the VLA action expert, instantiated by a causal-transformer route via standard PPO or a diffusion-policy route via DPPO or DDiffPG.

For the causal-transformer route, conversion into a DiT-style action expert reduces to three operations - inserting adaLN, repurposing some self-attentions as cross-attentions to a frozen VLM, and fitting embodiment-specific I/O projections. For the diffusion-policy route, the architecture already matches the VLA action expert and the pretrained trunk transfers directly; only the cross-attention to the VLM and the embodiment-specific I/O projections need to be added. Standard SFT on demonstration data then takes over.

The framework is orthogonal to the existing line of RL post-training of VLAs and to advantage-conditioned offline RL. Every architectural and algorithmic component already exists. What remains is the empirical study of the ordering - RL before SFT - and of how broadly the resulting physics priors transfer across embodiments and tasks.

References

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

[2] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025. URL https://arxiv.org/abs/2504.16054.

[3] Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, Danny Driess, Michael Equi, Adnan Esmail, Yunhao Fang, Chelsea Finn, Catherine Glossop, Thomas Godden, Ivan Goryachev, Lachy Groom, Hunter Hancock, Karol Hausman, Gashon Hussein, Brian Ichter, Szymon Jakubczak, Rowan Jen, Tim Jones, Ben Katz, Liyiming Ke, Chandra Kuchi, Marinda Lamb, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Yao Lu, Vishnu Mano, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Charvi Sharma, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, Will Stoeckle, Alex Swerdlow, James Tanner, Marcel Torne, Quan Vuong, Anna Walling, Haohuan Wang, Blake Williams, Sukwon Yoo, Lili Yu, Ury Zhilinsky, and Zhiyuan Zhou. $\pi^*_{0.6}$: a vla that learns from experience, 2025. URL https://arxiv.org/abs/2511.14759.

[4] Physical Intelligence, Bo Ai, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Greg Balke, Kevin Black, George Bokinsky, Shihao Cao, Thomas Charbonnier, Vedant Choudhary, Foster Collins, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Maitrayee Dhaka, Jared DiCarlo, Danny Driess, Michael Equi, Adnan Esmail, Yunhao Fang, Chelsea Finn, Catherine Glossop, Thomas Godden, Ivan Goryachev, Lachlan Groom, Haroun Habeeb, Hunter Hancock, Karol Hausman, Gashon Hussein, Victor Hwang, Brian Ichter, Connor Jacobsen, Szymon Jakubczak, Rowan Jen, Tim Jones, Gregg Kammerer, Ben Katz, Liyiming Ke, Mairbek Khadikov, Chandra Kuchi, Marinda Lamb, Devin LeBlanc, Brendon LeCount, Sergey Levine, Xinyu Li, Adrian Li-Bell, Vladislav Lialin, Zhonglin Liang, Wallace Lim, Yao Lu, Enyu Luo, Vishnu Mano, Nandan Marwaha, Aikys Mongush, Liam Murphy, Suraj Nair, Tyler Patterson, Karl Pertsch, Allen Z. Ren, Gavin Schelske, Charvi Sharma, Baifeng Shi, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, Will Stoeckle, Jiaming Tang, Jimmy Tanner, Shalom Tekeste, Marcel Torne, Kyle Vedder, Quan Vuong, Anna Walling, Haohuan Wang, Jason Wang, XuDong Wang, Chris Whalen, Samuel Whitmore, Blake Williams, Charles Xu, Sukwon Yoo, Lili Yu, Wuming Zhang, Zhuoyang Zhang, and Ury Zhilinsky. $\pi_{0.7}$: a steerable generalist robotic foundation model with emergent capabilities, 2026. URL https://arxiv.org/abs/2604.15483.

[5] NVIDIA, :, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. Gr00t n1: An open foundation model for generalist humanoid robots, 2025. URL https://arxiv.org/abs/2503.14734.

[6] Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Remi Cadene. Smolvla: A vision-language-action model for affordable and efficient robotics, 2025. URL https://arxiv.org/abs/2506.01844.

[7] Jinliang Zheng, Jianxiong Li, Zhihao Wang, Dongxiu Liu, Xirui Kang, Yuchun Feng, Yinan Zheng, Jiayin Zou, Yilun Chen, Jia Zeng, Ya-Qin Zhang, Jiangmiao Pang, Jingjing Liu, Tai Wang, and Xianyuan Zhan. X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model, 2025. URL https://arxiv.org/abs/2510.10274.

[8] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 5026–5033, 2012. doi: 10.1109/IROS.2012.6386109.

[9] Mayank Mittal, Pascal Roth, James Tigue, Antoine Richard, Octi Zhang, Peter Du, Antonio Serrano-Muñoz, Xinjie Yao, René Zurbrügg, Nikita Rudin, Lukasz Wawrzyniak, Milad Rakhsha, Alain Denzler, Eric Heiden, Ales Borovicka, Ossama Ahmed, Iretiayo Akinola, Abrar Anwar, Mark T. Carlson, Ji Yuan Feng, Animesh Garg, Renato Gasoto, Lionel Gulich, Yijie Guo, M. Gussert, Alex Hansen, Mihir Kulkarni, Chenran Li, Wei Liu, Viktor Makoviychuk, Grzegorz Malczyk, Hammad Mazhar, Masoud Moghani, Adithyavairavan Murali, Michael Noseworthy, Alexander Poddubny, Nathan Ratliff, Welf Rehberg, Clemens Schwarke, Ritvik Singh, James Latham Smith, Bingjie Tang, Ruchik Thaker, Matthew Trepte, Karl Van Wyk, Fangzhou Yu, Alex Millane, Vikram Ramasamy, Remo Steiner, Sangeeta Subramanian, Clemens Volk, CY Chen, Neel Jawale, Ashwin Varghese Kuruttukulam, Michael A. Lin, Ajay Mandlekar, Karsten Patzwaldt, John Welsh, Jean-Francois Lafleche, Nicolas Moënne-Loccoz, Soowan Park, Rob Stepinski, Dirk Van Gelder, Chris Amevor, Jan Carius, Jumyung Chang, Anka

He Chen, Pablo de Heras Ciechomski, Gilles Daviet, Mohammad Mohajerani, Julia von Muralt, Viktor Reutskyy, Michael Sauter, Simon Schirm, Eric L. Shi, Pierre Terdiman, Kenny Vilella, Tobias Widmer, Gordon Yeoman, Tiffany Chen, Sergey Grizan, Cathy Li, Lotus Li, Connor Smith, Rafael Wiltz, Kostas Alexis, Yan Chang, Linxi "Jim" Fan, Farbod Farshidian, Ankur Handa, Spencer Huang, Marco Hutter, Yashraj Narang, Soha Pouya, Shiwei Sheng, Yuke Zhu, Miles Macklin, Adam Moravanszky, Philipp Reist, Yunrong Guo, David Hoeller, and Gavriel State. Isaac Lab - A GPU-Accelerated Simulation Framework for Multi-Modal Robot Learning. arXiv preprint arXiv:2511.04831, 2025. doi: 10.48550/arXiv.2511.04831. URL https://arxiv.org/abs/2511.04831.

[10] Ilija Radosavovic, Tete Xiao, Bike Zhang, Trevor Darrell, Jitendra Malik, and Koushil Sreenath. Real-world humanoid locomotion with reinforcement learning, 2023. URL https://arxiv.org/abs/2303.03381.

[11] Ilija Radosavovic, Sarthak Kamat, Trevor Darrell, and Jitendra Malik. Learning humanoid locomotion over challenging terrain, 2024. URL https://arxiv.org/abs/2410.03654.

[12] Zhengyi Luo, Ye Yuan, Tingwu Wang, Chenran Li, Sirui Chen, Fernando Castañeda, Zi-Ang Cao, Jiefeng Li, David Minor, Qingwei Ben, Xingye Da, Runyu Ding, Cyrus Hogg, Lina Song, Edy Lim, Eugene Jeong, Tairan He, Haoru Xue, Wenli Xiao, Zi Wang, Simon Yuen, Jan Kautz, Yan Chang, Umar Iqbal, Linxi "Jim" Fan, and Yuke Zhu. Sonic: Supersizing motion tracking for natural humanoid whole-body control, 2025. URL https://arxiv.org/abs/2511.07820.

[13] Tinghao Yi, Quantao Yang, and Enhong Chen. Actionx: pre-training action experts with reinforcement learning for vision-language action models. Frontiers in Neurorobotics, 20:1806605, 2026. doi: 10.3389/fnbot.2026.1806605. URL https://doi.org/10.3389/fnbot.2026.1806605.

[14] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

[15] Allen Z. Ren, Justin Lidard, Lars L. Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization, 2024. URL https://arxiv.org/abs/2409.00588.

[16] Zechu Li, Rickmer Krohn, Tao Chen, Anurag Ajay, Pulkit Agrawal, and Georgia Chalvatzaki. Learning multimodal behaviors from scratch with diffusion policy gradient, 2024. URL https://arxiv.org/abs/2406.00681.

[17] William Peebles and Saining Xie. Scalable diffusion models with transformers, 2023. URL https://arxiv.org/abs/2212.09748.

[18] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion, 2024. URL https://arxiv.org/abs/2303.04137.

[19] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling, 2023. URL https://arxiv.org/abs/2210.02747.

[20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. URL https://arxiv.org/abs/1706.03762.

[21] Yanjiang Guo, Jianke Zhang, Xiaoyu Chen, Xiang Ji, Yen-Jen Wang, Yucheng Hu, and Jianyu Chen. Improving vision-language-action model with online reinforcement learning, 2025. URL https://arxiv.org/abs/2501.16664.

[22] Yuhui Chen, Shuai Tian, Shugao Liu, Yingting Zhou, Haoran Li, and Dongbin Zhao. Conrft: A reinforced fine-tuning method for vla models via consistency policy, 2025. URL https://arxiv.org/abs/2502.05450.

[23] Haozhan Li, Yuxin Zuo, Jiale Yu, Yuhao Zhang, Zhaohui Yang, Kaiyan Zhang, Xuekai Zhu, Yuchen Zhang, Tianxing Chen, Ganqu Cui, Dehui Wang, Dingxiang Luo, Yuchen Fan, Youbang Sun, Jia Zeng, Jiangmiao Pang, Shanghang Zhang, Yu Wang, Yao Mu, Bowen Zhou, and Ning Ding. Simplevla-rl: Scaling vla training via reinforcement learning, 2025. URL https://arxiv.org/abs/2509.09674.

[24] Kang Chen, Zhihao Liu, Tonghe Zhang, Zhen Liu, Sicheng Wang, Xue Bin Peng, Yuke Zhu, Saining Xie, Chong Zhang, et al. π_{rl}: Online rl fine-tuning for flow-based vision-language-action models, 2025. URL https://arxiv.org/abs/2510.25889.

[25] Hengtao Li, Pengxiang Ding, Runze Suo, Yihao Wang, Zirui Ge, Dongyuan Zang, Kexian Yu, Mingyang Sun, Hongyin Zhang, Donglin Wang, and Weihua Su. Vla-rft: Vision-language-action reinforcement fine-tuning with verified rewards in world simulators, 2025. URL https://arxiv.org/abs/2510.00406.