

Execution-Aware VLAs for Robust Manipulation

Abstract—Pretrained vision-language-action (VLA) models typically assume a particular execution dynamics, limiting their robustness when deployed with different robot controllers. In practice, physical robots exhibit tracking error, latency, actuator saturation, compliance effects, calibration inaccuracies, and embodiment-specific dynamics, causing discrepancies between commanded policy actions and the realized robot motion. We refer to this discrepancy as the *execution gap*. To address this execution gap, we propose Execution-Aware Vision-Language-Action (EA-VLA), an approach for adapting pretrained VLA models to varying execution dynamics. EA-VLA augments the robot policy with execution-state feedback, enabling the policy to reason about discrepancies between commanded and executed behavior during action generation.

By conditioning on robot proprioception and execution-state information, the proposed approach learns to compensate for the execution gap and tracking error. We demonstrate EA-VLA performance in both simulation and real robot evaluation. In simulation evaluations, we observe EA-VLA maintains performance across a much larger perturbation magnitudes than those tolerated by baseline VLAs. We showcase robustness to underlying controller on multiple-embodiments, achieving 11% higher success rate on the WidowX robot and up to 50% improvement in throughput on the Franka embodiment. These results demonstrate that incorporating execution-state feedback can substantially improve the robustness of pretrained VLAs to variations in how actions are executed.

I. INTRODUCTION

Recent Vision-Language-Action (VLA) models [1], [2], [3], [4] have demonstrated impressive capabilities for robot manipulation, enabling policies that condition on natural language and generalize across tasks, environments, and robot embodiments. This progress has been driven in large part by scaling imitation learning across increasingly diverse robot datasets [5], [6], establishing a promising paradigm for scalable robot learning. However, as VLA policies are deployed across increasingly diverse physical platforms, an important systems-level assumption remains largely overlooked: the relationship between a commanded action and the motion actually realized by the robot.

Standard VLA policies typically predict actions under the implicit assumption that these commands are faithfully executed by the underlying control system. In practice, however, physical robots exhibit tracking error, latency, actuator saturation, compliance, calibration inaccuracies, and controller-dependent behavior. Consequently, the realized robot motion can deviate substantially from the motion implied by the policy’s commanded actions. We refer to this discrepancy as the *execution gap*. Unlike visual or semantic distribution shift, which changes what the policy observes or what task it is asked to perform, the execution gap changes how the policy’s commands are physically realized, as shown in

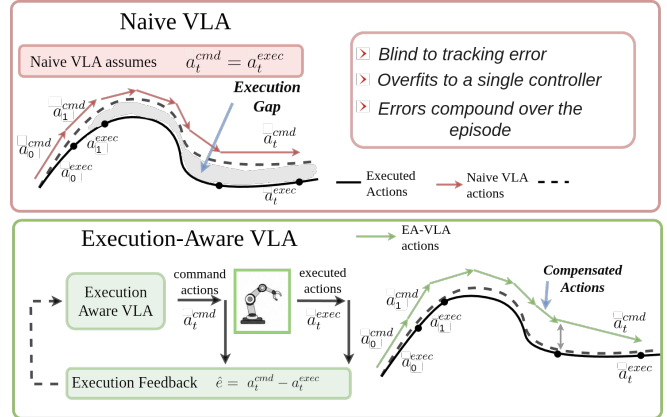


Fig. 1: a) Naive VLA predicts actions assuming commanded actions are tracked accurately. However, this is not the case during real world deployment. b) We propose an Execution-Aware VLA that incorporates execution feedback during action generation, enabling the model to produce compensated actions that account for tracking errors.

Fig. 1.

While small in magnitude, such discrepancies can be detrimental for robot especially during interaction with objects. For example, on WidowX, we observe commanded and realized end-effector trajectories differing by up to 2 cm during an episode, while on Franka, tracking deviations can exceed 10 cm depending on the controller configuration, with larger errors under compliant control and faster motions. Fig. 2 illustrates representative examples. Although these deviations might seem minor compared to the workspace, they can be substantial relative to the precision needed for grasping, manipulating and placing objects. Importantly, imitation policies are especially vulnerable to such perturbations, since small execution errors compound over time and induce state-distribution drift during deployment [7]. Recent work has highlighted the importance of execution effects in robot learning, including action delays, controller dynamics, and other forms of execution mismatch [8], [9], [10], [11], [12].

The execution gap poses a fundamental challenge for pretrained VLAs because the data used to train these models are collected under heterogeneous robots, controllers, and control interfaces [13], [14]. A policy can therefore encounter an execution regime at deployment that differs substantially from those represented during training. In practice, adapting a robot to a new deployment setting often requires controller-specific tuning or policy retraining to compensate for these

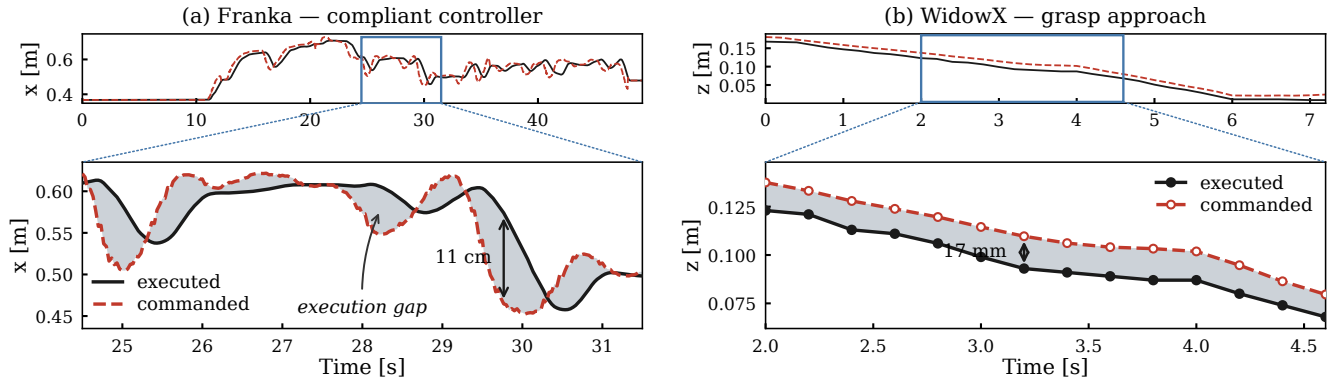


Fig. 2: **The execution gap on physical robots.** Commanded policy actions (dashed) versus realized end-effector motion (solid); shaded regions mark the discrepancy. (a) Franka under a compliant controller, where tracking deviations reach 11 cm. (b) WidowX during a grasp approach, with an average 13 mm offset.

differences. Such controller optimization is tedious and does not necessarily yield an optimal solution: the desired execution behavior is inherently task-dependent, requiring trade-offs among tracking accuracy, compliance, safety, and throughput. A controller that provides precise tracking may be overly stiff for contact-rich manipulation, while a more compliant configuration may improve safety and interaction but introduce larger tracking errors. Consequently, controller selection and tuning can become a costly per-robot and per-task optimization problem, particularly when deploying a pretrained VLA across heterogeneous platforms. Rather than relying solely on controller-level optimization to eliminate execution errors, we ask whether the policy itself can adapt its actions to the execution dynamics it encounters. We evaluate EA-VLA in both controlled simulation and real-world robot experiments. In simulation, we systematically vary the magnitude of execution perturbations and compare EA-VLA against pretrained and supervised-finetuned VLA baselines. In real-world experiments, we evaluate robustness to different controller configurations on both Franka and WidowX robots. Across these settings, EA-VLA substantially improves robustness to execution mismatch, maintaining reliable task performance under significantly larger execution perturbations than those tolerated by baseline VLAs.

Overall, this work makes the following contributions:

- We identify and characterize the *execution gap* as an important failure mode for pretrained VLA policies, and systematically evaluate their sensitivity to heterogeneous control and execution dynamics while keeping the task, embodiment, and environment fixed.
- We introduce **EA-VLA**, an execution-aware adaptation framework that incorporates robot proprioception and execution-state feedback into VLA action generation, enabling the policy to compensate for discrepancies between commanded and realized robot motion.
- We evaluate the proposed approach on both real-world experiments and controlled simulation ablations, demonstrating improved robustness and transfer across controllers, embodiments, and physical robot platforms.

Our experimental evaluations suggest that scalable robot learning requires moving beyond the assumption of a fixed execution interface. As VLA systems are deployed across increasingly diverse robots and controllers, robustness to execution variability may become as important as robustness to visual or semantic distribution shift itself.

II. RELATED WORK

A. Vision-Language-Action Models

Vision-Language-Action (VLA) models have emerged as a promising paradigm for scalable robot learning, demonstrating strong zero-shot generalization across diverse manipulation tasks, embodiments, and environments [2], [15], [16], [6]. A common design choice is to leverage pretrained Vision-Language Models (VLMs) as perceptual backbones and adapt their representations for embodied decision-making and control.

Existing VLA architectures can be broadly categorized into two families. The first treats control as autoregressive sequence modeling, finetuning VLMs to directly predict discretized action tokens [15], [17]. The second introduces a dedicated action module or “action expert” on top of a frozen or partially frozen VLM, which predicts continuous control signals [18], [16], [1], [19]. This design partially decouples high-level perception-language reasoning from low-level motor control and has been shown to improve performance on a range of manipulation benchmarks [20].

Beyond architectural choices, prior work has explored temporal modeling to improve long-horizon consistency and closed-loop performance [21], [22], [23], typically through conditioning on histories of observations and states. However, most approaches primarily focus on visual and proprioceptive history, without explicitly modeling the relationship between commanded actions and their executed outcomes under heterogeneous control dynamics.

B. RL finetuning of VLA

A rapidly growing line of work explores how to improve pretrained Vision-Language-Action (VLA) models via rein-

forcement learning (RL). These approaches vary primarily in their update mechanisms and how the RL signal is incorporated. One prevalent approach leverages offline RL to update the full VLA model. For instance, RECAP [19] trains the entire model end-to-end via advantage-conditioned policy extraction. By utilizing a distributional value function to estimate per-timestep advantages, the VLA is trained on a comprehensive dataset of demonstrations, autonomous rollouts, and human interventions. Through iterative cycles of on-robot data collection and offline RL updates, RECAP demonstrates significantly improved throughput on complex, long-horizon tasks such as espresso making, laundry folding, and box assembly. Conversely, a distinct body of prior work focuses on online RL finetuning [24], [25], [26], often leveraging Proximal Policy Optimization (PPO) [27] or variants thereof [28], [29], [30] through on-policy methods. Within this paradigm, DPPO [29] proposes a two-stage Markov Decision Process (MDP) to couple denoising steps with environment steps, thereby unifying the denoising and simulator MDPs into a single formulation. Furthermore, methods such as Flow-GRPO [31] and π_{rl} [30] enhance stochastic exploration by converting the denoising process from an Ordinary Differential Equation (ODE) to a Stochastic Differential Equation (SDE) formulation, facilitating efficient online RL.

C. Simulation for Manipulation

Robotics simulation enables the generation and supervision of data in ways that are not feasible with purely offline datasets. It has been explored as a source of synthetic and structured data [32], [33], as well as a provider of privileged information within teacher-student frameworks [34], [35], [36].

Domain randomization has been extensively studied for learning robust robot controllers in locomotion [37], [38] and whole-body control [39], [40]. Similar approaches have also been proposed for manipulation and grasping [41], [35], [42]. More recently, simulation has additionally been adopted in vision-language-action (VLA) systems for generated training data [6], evaluation, distillation, and demonstrating online RL.

III. METHOD

In this section, we present our methodology. We first provide the necessary background in Sec. III-A, introducing the key components of prior work that our approach builds upon. In Sec. III-B, we introduce EA-VLA and describe the architectural modifications to the underlying VLA backbone. Finally, in Sec. III-C, we detail our reinforcement learning training procedure and simulator setup used for policy optimization and evaluation.

A. Preliminaries

a) VLA Policies: During inference, the model receives an observation tuple $o_t = (c_t, \ell, s_t)$, where c_t denotes one or more camera views, ℓ is the language instruction, and s_t is

the robot proprioceptive state. Following the common action-chunking formulation used by modern denoising VLAs, the model predicts a short horizon of actions, action chunks $A_t = (a_t, a_{t+1}, \dots, a_{t+N-1})$, where N is the number of actions in the chunk. Action chunks typically span 1-5 s and are executed at 5-15 Hz, as dictated by the data-collection frequency

b) Flow Matching for VLA: Flow matching action expert is tasked with generating actions by learning a conditional vector field v_θ that maps noise to target actions A_t via the Conditional Flow Matching objective [43]:

$$\mathcal{L}_{\text{CFM}} = \mathbb{E}_{\tau, p(A_t, o_t), q(A_t^\tau | A_t)} \left[\|v_\theta(A_t^\tau, o_t) - u(A_t^\tau | A_t)\|_2^2 \right], \quad (1)$$

where the A_t^τ is the noisy action and u is the target vector and τ as the denoising timestamp.

$$A_t^\tau = \tau A_t + (1 - \tau)\epsilon, \quad \epsilon \sim \mathcal{N}(0, I), \quad \tau \in [0, 1], \quad (2)$$

$$u(A_t^\tau | A_t) = A_t - \epsilon$$

During the inference, the action sequence is generated by first sampling a noise vector $\mathbf{A}_t^0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, which is further iteratively refined by integrating the learned vector field v_θ , the action-expert, over a fixed number of integration steps using the forward Euler method [43]:

$$A_t^{\tau+\delta} = A_t^\tau + v_\theta(A_t^\tau, o_t) \cdot \delta \quad (3)$$

B. Execution-Aware VLA

We propose **EA-VLA**, a framework that augments a pretrained VLA with short-horizon execution feedback to account for discrepancies between commanded actions and realized robot motion, as illustrated in Fig. 3. Our central hypothesis is that recent execution history provides sufficient information for the policy to identify systematic execution discrepancies and adapt subsequent actions accordingly. Rather than explicitly modeling or correcting the execution error at the controller level, EA-VLA allows the policy to learn this compensation directly from the relationship between commanded actions and observed robot responses.

a) Execution-State Representation.: To expose execution behavior to the policy, we construct an execution-state representation from the previously commanded action and the subsequent observed robot state:

$$e_t = \phi(a_{t-1}^{\text{cmd}}, s_t, s_{t-1}), \quad (4)$$

where a_{t-1}^{cmd} denotes the action commanded at the previous timestep, s_t denotes the current robot state, and $\phi(\cdot)$ is a lightweight two-layer MLP that encodes these quantities into a learned execution-state representation. The raw commanded action and observed robot state are provided jointly to ϕ , allowing the model to learn the correspondence between intended and realized behavior.

In our implementation, s_t is represented using robot proprioception, including the joint state, while a_{t-1}^{cmd} represents the action previously issued by the VLA. The resulting representation therefore contains information about both what the policy intended the robot to do and the state in which the robot actually arrived after execution.

We maintain a short history of these execution-state representations over a horizon of H timesteps and augment the standard VLA observation as

$$\hat{o}_t = (o_t, e_{t-H:t}), \quad (5)$$

where o_t denotes the original VLA observation. The commanded actions and robot states are normalized using the corresponding statistics from the original training dataset. We find empirically that providing the raw commanded actions and observed states to the learned encoder leads to more stable convergence than explicitly computing execution residuals.

b) Execution-Aware Action Generation.: The resulting execution-state tokens are provided exclusively to the action expert, which is responsible for action generation. This design allows the action-generation module to jointly condition on the task observation, the previously commanded actions, and the corresponding realized robot states. The model can consequently learn systematic relationships between commanded behavior and physical execution, and use these relationships to adjust subsequent actions.

This creates a short-horizon execution feedback loop: the VLA generates a command, the robot executes the command through its underlying controller, the resulting robot state is observed, and the command-state pair is fed back to the policy for subsequent action generation. Because this feedback directly exposes how previous commands were realized, EA-VLA can learn to compensate for controller- and embodiment-dependent execution behavior without requiring explicit residual computation or modification of the underlying low-level controller.

C. RL Adaptation

Building on the execution-aware observation, we adapt the pretrained VLA using reward-based learning. The execution feedback serves as the conditioning signal, while task reward provides the learning signal, allowing the policy to learn actions that compensate for execution mismatch through closed-loop interaction with the simulator.

a) Policy Optimization.: We optimize the flow-matching action expert using PPO [27], following the policy-gradient formulation for flow-based policies introduced by DPPO [29]. We use the RLInf framework [26] to implement action-chunked policy optimization, where the policy generates a chunk of N consecutive actions that is executed in the environment. Policy updates are performed at the chunk level using advantages computed from the resulting rollout rewards.

b) Critic and Reward.: We obtain value estimates from the pretrained VLM representation through a lightweight MLP value head, which is trained from rollout returns. We use a semi-dense task-completion reward:

$$R_{\text{env}}(s_t, a_t) = \begin{cases} 0.5 & \text{if the target object is grasped,} \\ 0.5 & \text{if the task is successfully completed,} \\ 0 & \text{otherwise.} \end{cases} \quad (6)$$

Rewards are accumulated over the N actions within each executed chunk:

$$R_t = \sum_{j=0}^{N-1} r_{t,j}. \quad (7)$$

The resulting chunk-level rewards provide the training signal for adapting the action expert to the execution dynamics encountered during interaction.

c) Controller Randomization.: To expose the policy to diverse execution dynamics during training, we randomize the parameters of the underlying low-level controller. We use a standard PD controller with gravity compensation:

$$\tau_t = K_p(q_t^d - q_t) + K_d(\dot{q}_t^d - \dot{q}_t) + \tau_g(q_t), \quad (8)$$

where q_t^d and $\dot{q}_t^d$ denote the commanded joint position and velocity, q_t and $\dot{q}_t$ denote the measured joint position and velocity, and K_p and K_d denote the controller stiffness and damping. For each randomized controller parameter $k \in \{K_p, K_d, \dots\}$, we apply a multiplicative perturbation in log-space:

$$k' = k \exp(\sigma \epsilon), \quad \epsilon \sim \mathcal{U}(-1, 1), \quad (9)$$

where k and k' denote the nominal and randomized parameters, respectively, and σ controls the magnitude of the perturbation. We randomize stiffness, damping, and friction parameters to expose the policy to a family of closed-loop execution behaviors. Importantly, the randomized controller parameters are not provided to the policy; instead, the policy must infer the resulting execution behavior from the observed command-state history.

We evaluate $\sigma \in \{0, 0.1, 0.5, 1, 2\}$, with $\sigma = 0$ corresponding to the nominal controller without randomization. Increasing σ broadens the distribution of execution behavior and encourages the policy to rely on observed execution feedback rather than a fixed controller model. During training, we use $\sigma = 1$ for simulation experiments and $\sigma = 0.1$ for sim-to-real transfer. This randomization is central to our execution-aware formulation, as it trains the policy to adapt to controller-dependent variations in how commanded actions are realized while leaving the underlying low-level controller unchanged at deployment.

IV. EXPERIMENTS AND RESULTS

A. Experimental Setup

We evaluate EA-VLA across both simulated and real-world robotic experiments. Our experiments are designed to answer three key questions: (1) whether execution-state history improves VLA performance across different robot platforms and tasks, (2) how EA-VLA performs under different low-level robot controllers, and (3) whether the proposed execution-state representation provides benefits beyond generic temporal proprioceptive history.

a) WidowX Simulation Setup.: We utilize the WidowX robot in the ManiSkill simulation environment, following the physical configuration established by the BridgeData V2 dataset [44]. The simulation environments and assets are adapted from ReVLA [45] and RLinf [26]. Policies operate

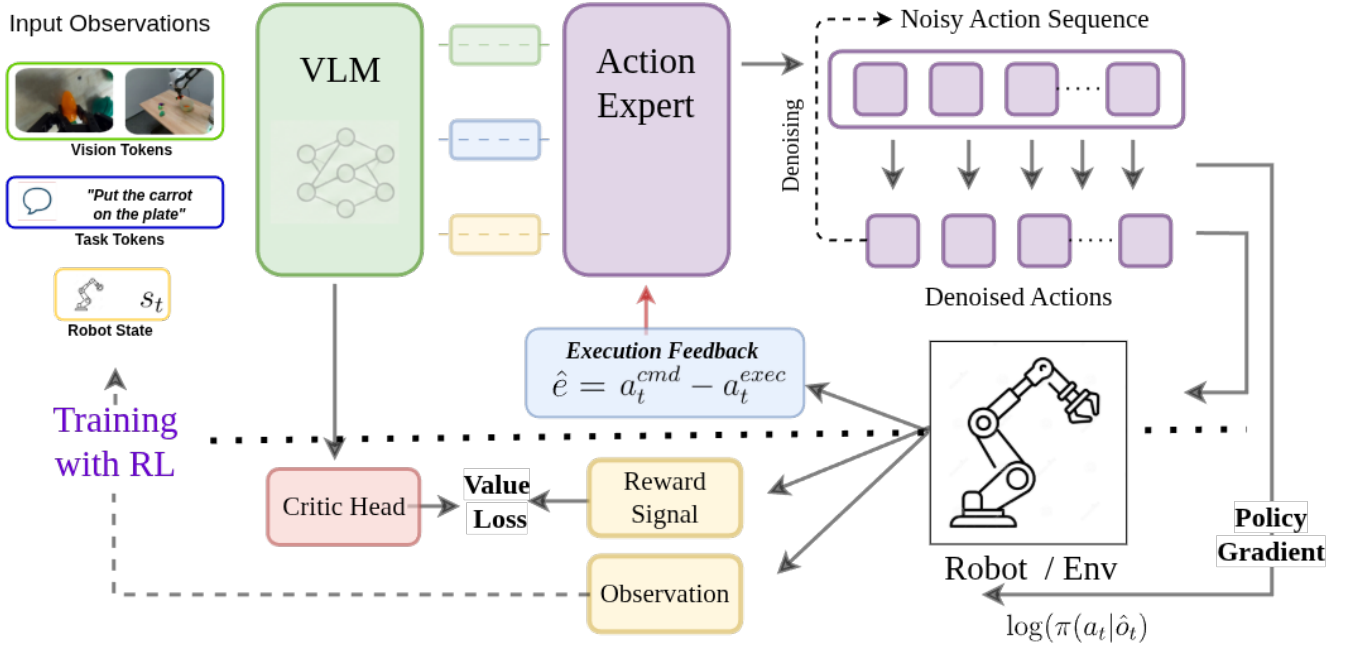


Fig. 3: **Execution Aware VLA**: We present our framework highlighting the Execution Feedback to the VLA model. We further propose an RL training paradigm with value-estimates from the VLM

at 5 Hz and are trained with batched domain randomization over 1024 parallel environments. Unless otherwise specified, we use $H = 2$ steps of execution-state history.

b) *Franka Dataset Setup.*: We further evaluate the proposed approach on real-world Franka manipulation tasks using a dataset setup closely following DROID [46]. For the Franka setup, we report results using both self-collected data and LoRA-based finetuning.

c) *Bridge Dataset Setup.*: For the physical WidowX experiments, we follow the hardware and observation configuration of the BridgeData V2 setup. The results on the WidowX robot are obtained in a zero-shot setting, without any further finetuning on robot-specific data.

B. Simulation Results

Model	Randomization Ratio (σ)				
	0.0	0.1	0.5	1	2
SFT Baseline	48.7%	49.7%	14.2%	4.3%	2.9%
+ RL	83.4%	51.1%	11.2%	8.1%	4.4%
EA-VLA	77.5%	80.1%	60.4%	28.6%	6.1%

TABLE I: Success rates across increasing degrees of dynamics randomization ($\sigma = 0.0$ to 2). EA-VLA demonstrates substantially greater robustness to dynamics perturbations.

To evaluate the robustness of our proposed model to physical parameter discrepancies, we subject the policies to varying degrees of system dynamics perturbations within the ManiSkill simulator. Table I reports task success rates across a range of domain randomization scales ($\sigma \in [0.0, 2]$). This parameter controls the variance of core physical attributes, including joint stiffness, damping, and object friction, relative to the nominal system-identified configuration ($\sigma = 0.0$).

RL finetuning without state information achieves the highest peak success rate (83.4%) under nominal conditions ($\sigma = 0.0$), but overfits to the simulator dynamics, lacking proprioceptive feedback and action history, the policy is unable to perform online tracking corrections. Consequently, even a minor controller perturbation ($\sigma = 0.1$) causes a sharp 32.3 percentage-point drop in performance, with performance collapsing for larger shifts ($\sigma \geq 0.5$).

In contrast, **EA-VLA** maintains stable performance (77.5% at $\sigma = 0.0$ and 80.1% at $\sigma = 0.1$) and degrades much more gracefully. The training reward curves further show that access to execution feedback yields higher returns, Fig. 5. At the training randomization level ($\sigma = 1$), our approach achieves a 28.6% success rate, outperforming the +RL variant by more than three times. Notably, the RL finetuned baseline outperforms EA-VLA under nominal conditions ($\sigma = 0.0$), revealing a trade-off between peak in-domain performance under ideal dynamics and robustness to execution discrepancies.

1) *Ablation Study*: We ablate the robot-state history length H under increasing dynamics randomization ratios (σ). As shown in Tab. II, removing history ($H = 0$) achieves 83.4% success under nominal conditions but drops to 51.1% under minor perturbations ($\sigma = 0.1$), indicating poor robustness to the execution gap. Adding state history substantially improves robustness, with $H = 2$ reaching 60.4% success at $\sigma = 0.5$. Increasing the history to $H = 4$ provides negligible additional gains. We therefore use $H = 2$ as the default for all further experiments.

C. WidowX Robot: Sim-to-Real Transfer

We evaluate EA-VLA against the baseline on the WidowX robot, using the model trained in simulation, reporting

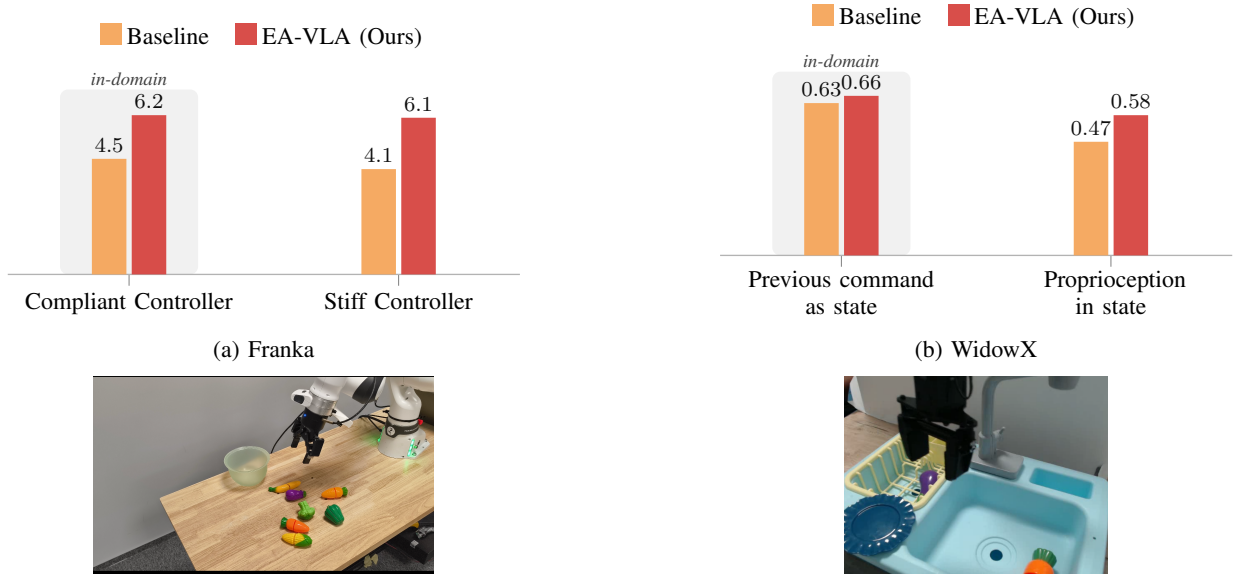


Fig. 4: **EA-VLA performance on (a) Franka and (b) WidowX.** For the Franka robot, we report throughput under two controller settings. For the WidowX robot, we report the average success rate. *in-domain* denotes the nominal condition: the compliant controller used for data collection (Franka) and previous-command state feedback (WidowX).

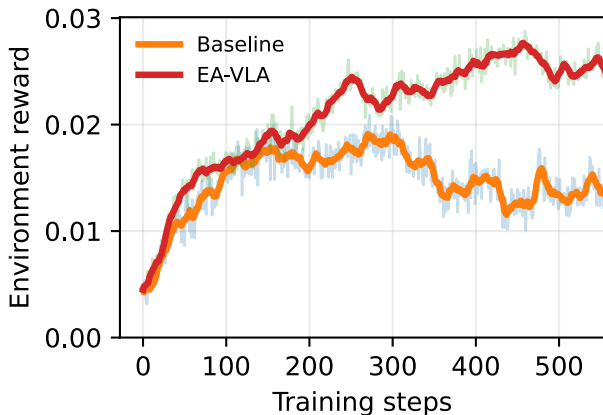


Fig. 5: Training reward comparison between the RL-Baseline VLA and EA-VLA.

results across two distinct inference strategies: (a) previous command as the state (b) using actual proprioceptive state as input (Fig. 4). The reported results are averaged over 25 trials on pick-place tasks from Bridge dataset. To ensure efficient sim-to-real transfer, we use σ value 0.1 and add a constant additive drift during RL adaptation phase for WidowX experiments.

The first strategy provides a heuristic proxy for execution feedback by feeding the desired robot state by the previous commanded action back to the policy. This can mitigate the accumulation of tracking errors, which is particularly relevant for VLAs that typically predict delta actions. By retaining the desired state as the policy input, the model can account for its recent action history when generating

History	Randomization Ratio (σ)				
Length (H)	0.0	0.1	0.5	1	2
$H = 0$	83.4%	51.1%	11.2%	8.1%	4.4%
$H = 1$	70.3%	70.5%	51.7%	22.2%	6.9%
$H = 2$	77.5%	80.1%	60.4%	28.6%	6.1%
$H = 4$	76.3%	76.3%	60.3%	26.4%	5.6%

TABLE II: Ablation study on the effect of history length H . Increasing the history length substantially improves robustness, while extending the history from $H = 2$ to $H = 4$ provides diminishing returns.

subsequent actions.

In contrast, the second strategy relies on true proprioceptive feedback, representing a more principled formulation that directly aligns with our training setup. Under this setting, the baseline performance decreased by 16%, whereas EA-VLA exhibits substantially less degradation, with an - 8% decrease. This result demonstrates that EA-VLA can leverage realized robot-state feedback to generate actions that compensate for underlying execution drift.

D. Franka Robot: Finetuning Results

We further investigate whether the benefits of execution-state history persist when adapting the VLA to a different robot platform and manipulation tasks. We finetune both the baseline and EA-VLA on the Franka dataset and evaluate them on bussing task involving seven objects, using 50 collected demonstrations. The data are collected using a compliant controller and subsequently evaluated under both compliant and stiff controller settings.

As shown in Fig. 4, EA-VLA improves task throughput by 38% and 49% compared to the baseline on the bussing task under compliant and stiff controller. The reported results are averaged over 15 trials. For Franka, we report throughput,

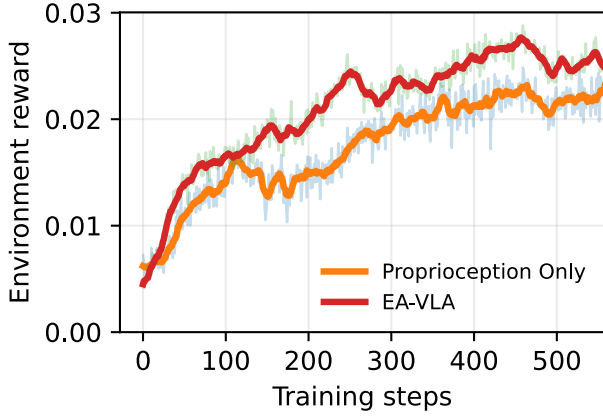


Fig. 6: Training reward comparison between a VLA conditioned on temporal proprioceptive history and our proposed execution-state formulation.

i.e., number of objects placed in the bowl in a fixed time horizon (200 steps), rather than binary task success to capture additional aspects of execution quality, including successful grasps, retries, and action efficiency.

E. Execution Gap vs. Temporal Proprioception

Finally, we investigate whether the observed gains arise specifically from the proposed execution-state formulation or simply from providing additional temporal proprioceptive information. We compare three variants: a baseline without state history, a model conditioned on temporal proprioceptive history, and EA-VLA, which is explicitly conditioned on the execution gap.

As shown in Fig. 6, explicitly modeling the execution-gap produces a substantially stronger reward trajectory than temporal proprioception alone. This results suggests that the key benefit comes from capturing the relationship between commanded and executed motion, rather than simply providing additional temporal context.

V. CONCLUSIONS

In this work, we identify the execution gap as an important failure mode for pretrained generalist robot policies and propose EA-VLA as a solution that accounts for this gap through execution-state-aware feedback. By conditioning action generation on this feedback and adapting the policy under controller randomization, EA-VLA demonstrates robust performance across varying execution settings where the baseline VLA struggles. We demonstrate robustness to variations in controller parameters in simulation, robustness to stiff and compliant controllers on the Franka robot, and improved performance from incorporating proprioception and previous commands as execution-state feedback on the WidowX robot.

These results suggest that, as VLA systems are deployed across increasingly diverse robots and controllers, robustness to execution variability is as essential to reliable deployment as robustness to visual and semantic distribution shifts.

a) *Limitations and Future Work.*: Our findings expose several limitations and opportunities for future research. First, demonstration datasets including human and UMI-style, which are not collected through tele-operation, do not provide execution or controller feedback. Extending execution-aware learning to policies trained on such data remains an important challenge. Second, while we study robustness across controller variations, it remains unclear whether execution-aware representations can generalize across embodiments with different morphology and control interfaces. Finally, our current formulation only uses proprioceptive state and command history. Incorporating richer feedback modalities such as torque, force, and tactile sensing is a promising avenue for future research.

REFERENCES

- [1] K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. R. Equi, C. Finn, N. Fusai, M. Y. Galliker, D. Ghosh, L. Groom, K. Hausman, brian ichter, S. Jakubczak, T. Jones, L. Ke, D. LeBlanc, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, A. Z. Ren, L. X. Shi, L. Smith, J. T. Springenberg, K. Stachowicz, J. Tanner, Q. Vuong, H. Walke, A. Walling, H. Wang, L. Yu, and U. Zhilinsky, “ $\pi_{0.5}$: a vision-language-action model with open-world generalization,” in *9th Annual Conference on Robot Learning*, 2025. [Online]. Available: <https://openreview.net/forum?id=vlhoswksBO>
- [2] N. Nikolov, G. Albanese, S. Dey, A. Yanev, L. V. Gool, J.-N. Zaech, and D. P. Paudel, “Spear-1: Scaling beyond robot demonstrations via 3d understanding,” 2025. [Online]. Available: <https://arxiv.org/abs/2511.17411>
- [3] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, Q. Vuong, V. Vanhoucke, H. Tran, R. Soricut, A. Singh, J. Singh, P. Sermanet, P. R. Sanketi, G. Salazar, M. S. Ryoo, K. Reymann, K. Rao, K. Pertsch, I. Mordatch, H. Michalewski, Y. Lu, S. Levine, L. Lee, T.-W. E. Lee, I. Leal, Y. Kuang, D. Kalashnikov, R. Julian, N. J. Joshi, A. Irpan, B. Ichter, J. Hsu, A. Herzog, K. Hausman, K. Gopalakrishnan, C. Fu, P. Florence, C. Finn, K. A. Dubey, D. Driess, T. Ding, K. M. Choromanski, X. Chen, Y. Chebotar, J. Carbajal, N. Brown, A. Brohan, M. G. Arenas, and K. Han, “Rt-2: Vision-language-action models transfer web knowledge to robotic control,” in *Proceedings of The 7th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, J. Tan, M. Toussaint, and K. Darvish, Eds., vol. 229. PMLR, 11 2023, pp. 2165–2183.
- [4] G. R. Team, S. Abeyruwan, J. Ainslie, J.-B. Alayrac, M. G. Arenas, T. Armstrong, A. Balakrishna, R. Baruch, M. Bauza, M. Blokzijl *et al.*, “Gemini robotics: Bringing ai into the physical world,” *arXiv preprint arXiv:2503.20020*, 2025.
- [5] J. Luo, C. Xu, F. Liu, L. Tan, Z. Lin, J. Wu, P. Abbeel, and S. Levine, “Fmb: a functional manipulation benchmark for generalizable robotic learning,” *The International Journal of Robotics Research*, 2024.
- [6] NVIDIA, J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. J. Fan, Y. Fang, D. Fox, F. Hu, S. Huang, J. Jang, Z. Jiang, J. Kautz, K. Kundalia, L. Lao, Z. Li, Z. Lin, K. Lin, G. Liu, E. Llontop, L. Magne, A. Mandlekar, A. Narayan, S. Nasiriany, S. Reed, Y. L. Tan, G. Wang, Z. Wang, J. Wang, Q. Wang, J. Xiang, Y. Xie, Y. Xu, Z. Xu, S. Ye, Z. Yu, A. Zhang, H. Zhang, Y. Zhao, R. Zheng, and Y. Zhu, “Gr00t n1: An open foundation model for generalist humanoid robots,” 2025. [Online]. Available: <https://arxiv.org/abs/2503.14734>
- [7] S. Ross, G. Gordon, and D. Bagnell, “A reduction of imitation learning and structured prediction to no-regret online learning,” in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, G. Gordon, D. Dunson, and M. Dudík, Eds., vol. 15. Fort Lauderdale, FL, USA: PMLR, 11–13 Apr 2011, pp. 627–635. [Online]. Available: <https://proceedings.mlr.press/v15/ross11a.html>
- [8] A. Bronars, Y. Park, and P. Agrawal, “Tune to learn: How controller gains shape robot policy learning,” *arXiv preprint arXiv:2604.02523*, 2026.

- [9] A. Liao, D.-K. Kim, M. O. Smith, A.-a. Agha-mohammadi, and S. Omidshafiei, "Delay-aware diffusion policy: Bridging the observation-execution gap in dynamic tasks," *arXiv preprint arXiv:2512.07697*, 2025.
- [10] K. Black, A. Z. Ren, M. Equi, and S. Levine, "Training-time action conditioning for efficient real-time chunking," *arXiv preprint arXiv:2512.05964*, 2025.
- [11] J. Tang, Y. Sun, Y. Zhao, S. Yang, Y. Lin, Z. Zhang, J. Hou, Y. Lu, Z. Liu, and S. Han, "Vlash: Real-time vlas via future-state-aware asynchronous inference," *arXiv preprint arXiv:2512.01031*, 2025.
- [12] N. R. Arachchige, Z. Chen, W. Jung, W. C. Shin, R. Bansal, P. Barroso, Y. H. He, Y. C. Lin, B. Joffe, S. Kousik *et al.*, "Sail: Faster-than-demonstration execution of imitation learning policies," *arXiv preprint arXiv:2506.11948*, 2025.
- [13] A. Padalkar, A. Pooley, A. Jain, A. Bewley, A. Herzog, A. Irpan, A. Khazatsky, A. Rai, A. Singh, A. Brohan *et al.*, "Open x-embodiment: Robotic learning datasets and rt-x models," *arXiv preprint arXiv:2310.08864*, 2023.
- [14] R. Cadene, S. Aliberts, F. Capuano, M. Aractingi, A. Zouitine, P. Kooijmans, J. Choghari, M. Russi, C. Pascal, S. Palma *et al.*, "Lerobot: An open-source library for end-to-end robot learning," *arXiv preprint arXiv:2602.22818*, 2026.
- [15] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi *et al.*, "Open-vla: An open-source vision-language-action model," *arXiv preprint arXiv:2406.09246*, 2024.
- [16] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter *et al.*, "pi-0: A vision-language-action flow model for general robot control," *arXiv preprint arXiv:2410.24164*, 2024.
- [17] K. Pertsch, K. Stachowicz, B. Ichter, D. Driess, S. Nair, Q. Vuong, O. Mees, C. Finn, and S. Levine, "Fast: Efficient action tokenization for vision-language-action models," *arXiv preprint arXiv:2501.09747*, 2025.
- [18] Q. Li, Y. Liang, Z. Wang, L. Luo, X. Chen, M. Liao, F. Wei, Y. Deng, S. Xu, Y. Zhang *et al.*, "Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation," *arXiv preprint arXiv:2411.19650*, 2024.
- [19] P. Intelligence, A. Amin, R. Aniceto, A. Balakrishna, K. Black, K. Conley, G. Connors, J. Darpinian, K. Dhabalia, J. DiCarlo *et al.*, "pi-0.6: a vla that learns from experience," *arXiv preprint arXiv:2511.14759*, 2025.
- [20] C. Pan, G. Anantharaman, N.-C. Huang, C. Jin, D. Pfaff, C. Yuan, F. Permenter, G. Qu, N. Boffi, G. Shi *et al.*, "Much ado about noising: Dispelling the myths of generative robotic control," *arXiv preprint arXiv:2512.01809*, 2025.
- [21] Octo Model Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, C. Xu, J. Luo, T. Kreiman, Y. Tan, L. Y. Chen, P. Sanketi, Q. Vuong, T. Xiao, D. Sadigh, C. Finn, and S. Levine, "Octo: An open-source generalist robot policy," in *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, 2024.
- [22] Y. Hu, J.-N. Zaeck, N. Nikolov, Y. Yao, S. Dey, G. Albanese, R. Detry, L. Van Gool, and D. Paudel, "Ar-vla: True autoregressive action expert for vision-language-action models," *arXiv preprint arXiv:2603.10126*, 2026.
- [23] H. Jang, S. Yu, H. Kwon, H. Jeon, Y. Seo, and J. Shin, "Contextvla: Vision-language-action model with amortized multi-frame context," *arXiv preprint arXiv:2510.04246*, 2025.
- [24] H. Li, Y. Zuo, J. Yu, Y. Zhang, Z. Yang, K. Zhang, X. Zhu, Y. Zhang, T. Chen, G. Cui *et al.*, "Simplevla-rl: Scaling vla training via reinforcement learning," *arXiv preprint arXiv:2509.09674*, 2025.
- [25] H. Zang, M. Wei, S. Xu, Y. Wu, Z. Guo, Y. Wang, H. Lin, L. Shi, Y. Xie, Z. Xu *et al.*, "Rlinf-vla: A unified and efficient framework for vla+ rl training," *arXiv preprint arXiv:2510.06710*, 2025.
- [26] C. Yu, Y. Wang, Z. Guo, H. Lin, S. Xu, H. Zang, Q. Zhang, Y. Wu, C. Zhu, J. Hu *et al.*, "Rlinf: Flexible and efficient large-scale reinforcement learning via macro-to-micro flow transformation," *arXiv preprint arXiv:2509.15965*, 2025.
- [27] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.
- [28] A. Ye, Z. Zhang, B. Wang, X. Wang, D. Zhang, and Z. Zhu, "Vla-rl: Enhancing reasoning in vision-language-action models," *arXiv preprint arXiv:2510.01623*, 2025.
- [29] A. Ren, J. Lidard, L. Ankile, A. Simeonov, P. Agrawal, A. Majumdar, B. Burchfiel, H. Dai, and M. Simchowitz, "Diffusion policy policy optimization," in *International Conference on Learning Representations*, vol. 2025, 2025, pp. 77 288–77 329.
- [30] K. Chen, Z. Liu, T. Zhang, Z. Guo, S. Xu, H. Lin, H. Zang, X. Li, Q. Zhang, Z. Yu *et al.*, "Pi-RL: Online RL Fine-tuning for Flow-based Vision-Language-Action Models," *arXiv preprint arXiv:2510.25889*, 2025.
- [31] J. Liu, G. Liu, J. Liang, Y. Li, J. Liu, X. Wang, P. Wan, D. Zhang, and W. Ouyang, "Flow-grpo: Training flow matching models via online rl," *Advances in neural information processing systems*, vol. 38, pp. 40 783–40 818, 2026.
- [32] A. Deshpande, M. Guru, R. Hendrix, S. Jauhri, A. Eftekhari, R. Tripathi, M. Argus, J. Salvador, H. Fang, M. Wallingford, W. Pumacay, Y. Kim, Q. Pfeifer, Y.-C. Lee, P. Wolters, O. Rayyan, M. Zhang, J. Duan, K. Farley, W. Han, E. Vanderbilt, D. Fox, A. Farhadi, G. Chaitzaki, D. Shah, and R. Krishna, "Molmob0t: Large-scale simulation enables zero-shot manipulation," 2026. [Online]. Available: <https://arxiv.org/abs/2603.16861>
- [33] F. Muratore, F. Ramos, G. Turk, W. Yu, M. Gienger, and J. Peters, "Robot learning from randomized simulations: A review," *Frontiers in Robotics and AI*, vol. 9, 2022.
- [34] T. G. W. Lum, M. Matak, V. Makoviychuk, A. Handa, A. Allshire, T. Hermans, N. D. Ratliff, and K. Van Wyk, "Dextrah-g: Pixels-to-action dexterous arm-hand grasping with geometric fabrics," *arXiv preprint arXiv:2407.02274*, 2024.
- [35] R. Singh, A. Allshire, A. Handa, N. Ratliff, and K. Van Wyk, "Dextrah-rgb: Visuomotor policies to grasp anything with dexterous hands," *arXiv preprint arXiv:2412.01791*, 2024.
- [36] M. N. Qureshi, S. Garg, F. Yandun, D. Held, G. Kantor, and A. Silwal, "SplatSim: Zero-shot sim2real transfer of rgb manipulation policies using gaussian splatting," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025, pp. 6502–6509.
- [37] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," *The International Journal of Robotics Research*, vol. 43, no. 4, pp. 572–587, 2024.
- [38] N. Fey, G. B. Margolis, M. Peticco, and P. Agrawal, "Bridging the sim-to-real gap for athletic loco-manipulation," *arXiv preprint arXiv:2502.10894*, 2025.
- [39] O. Kaidanov, F. Al-Hafez, Y. Suvari, B. Belousov, and J. Peters, "The role of domain randomization in training diffusion policies for whole-body humanoid control," *arXiv preprint arXiv:2411.01349*, 2024.
- [40] T. He, Z. Wang, H. Xue, Q. Ben, Z. Luo, W. Xiao, Y. Yuan, X. Da, F. Castañeda, S. Sastry *et al.*, "Viral: Visual sim-to-real at scale for humanoid loco-manipulation," *arXiv preprint arXiv:2511.15200*, 2025.
- [41] J. Ye, K. Wang, C. Yuan, R. Yang, Y. Li, J. Zhu, Y. Qin, X. Zou, and X. Wang, "Dex1b: Learning with 1b demonstrations for dexterous manipulation," *arXiv preprint arXiv:2506.17198*, 2025.
- [42] H. Gupta, M. A. Mirzaee, and W. Yuan, "Grasp to act: Dexterous grasping for tool use in dynamic settings," *IEEE Robotics and Automation Letters*, vol. 11, no. 5, pp. 6288–6295, 2026.
- [43] Y. Lipman, M. Havasi, P. Holderrieth, N. Shaul, M. Le, B. Karrer, R. T. Chen, D. Lopez-Paz, H. Ben-Hamu, and I. Gat, "Flow matching guide and code," *arXiv preprint arXiv:2412.06264*, 2024.
- [44] H. Walke, K. Black, A. Lee, M. J. Kim, M. Du, C. Zheng, T. Zhao, S. Hansen-Estruch, Q. Vuong, A. He, V. Myers, K. Fang, C. Finn, and S. Levine, "Bridgedata v2: A dataset for robot learning at scale," in *Proceedings of the Conference on Robot Learning (CoRL)*, 2023.
- [45] S. Dey, J.-N. Zaeck, N. Nikolov, L. Van Gool, and D. P. Paudel, "Revla: Reverting visual domain limitation of robotic foundation models," *arXiv preprint arXiv:2409.15250*, 2024.
- [46] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis *et al.*, "Droid: A large-scale in-the-wild robot manipulation dataset," in *Robotics: Science and Systems*, 2024.