Failing Forward: Adaptive Failure-Informed Learning for Vision-Language-Action Models

Meng Zheng[1], **Samhita Marri**[1,2]*, **Anwesa Choudhuri**[1], **Benjamin Planche**[1], **Zhongpai Gao**[1], **Van Nguyen Nguyen**[1], **Terrence Chen**[1], **Girish Chowdhary**[2] and **Ziyan Wu**[1]

[1]United Imaging Intelligence, Boston, MA, USA
[2]University of Illinois Urbana-Champaign, Urbana, IL, USA

Abstract

Vision-language-action (VLA) models provide a promising paradigm for scalable robotic manipulation, yet their reliance on success-only behavioral cloning leaves them brittle; lacking corrective training signals, minor execution errors rapidly compound into unrecoverable, out-of-distribution failures. To address this limitation, we propose Adaptive Failure-Informed Learning (AFIL), an end-to-end framework that leverages failure trajectories as adaptive negative guidance for diffusion- and flow-based VLA policies. AFIL uses a pretrained VLA to generate failure rollouts online, avoiding the need for handcrafted failure-mode design or human-in-the-loop recovery. It then jointly trains Dual Action Generators (DAGs) for successful and failed behaviors while sharing a common vision-language backbone, enabling efficient failure-aware policy learning with limited parameter overhead. During sampling, the failure generator adaptively steers action generation away from failure-prone regions and toward more reliable success modes, with guidance strength determined by the per-diffusion-step distance between success and failure distributions. Experiments across in-domain and out-of-domain robotic manipulation tasks, covering both short- and long-horizon settings, show that AFIL consistently improves task success rates and robustness over existing VLA baselines, demonstrating its effectiveness, efficiency, and generality.

1 Introduction

Robotic manipulation has long been a central challenge in robotics and artificial intelligence, requiring agents to perceive complex environments, reason over long horizons, and execute precise control under uncertainty. Conventional learning-based manipulation policies, including behavior cloning and reinforcement learning (RL), have achieved notable success in structured settings, but often require task-specific reward design, extensive interaction data, and careful environment engineering, limiting their scalability and generalization to diverse real-world tasks. Recently, vision-language-action (VLA) models have attracted growing attention in robotics [Pertsch et al., 2025, Black et al., 2024, Intelligence et al., 2025, Liu et al., 2025, Chi et al., 2025, Zitkovich et al., 2023, Zhao et al., 2025], particularly following the release of OpenVLA [Kim et al., 2024], which demonstrated the feasibility of leveraging large-scale pretrained vision-language representations for embodied decision-making.

Despite their promise, VLA models often struggle to generalize during deployment, as training data typically consists primarily of teleoperated, success-only trajectories collected in confined environments [Kim et al., 2024, O'Neill et al., 2024, James et al., 2020, Mandlekar et al., 2021]. Just as a smooth sea never made a skilled sailor, *a smooth demonstration never made a skilled robot*. Such data can cause policies to overfit to narrow behavioral manifolds and lack corrective signals for

*This work was carried out during the internship of Samhita Marri at United Imaging Intelligence, Boston, MA

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

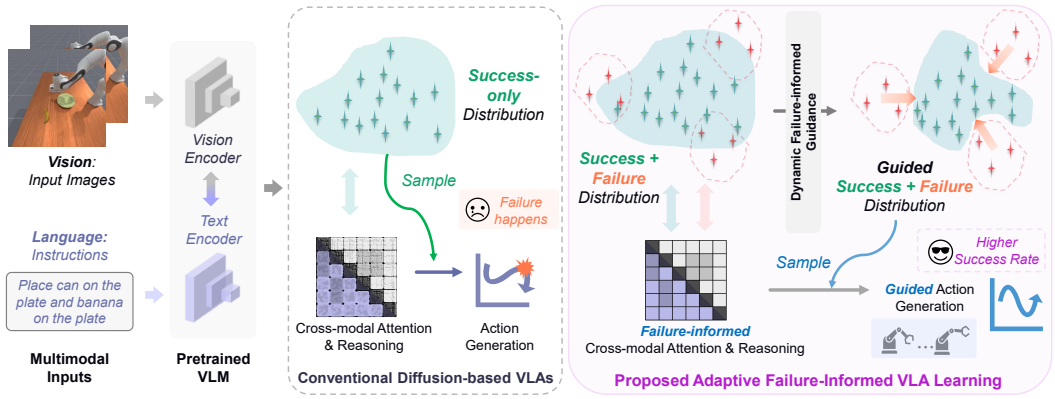


Figure 1: Overview of the proposed Adaptive Failure-Informed Learning (AFIL) pipeline. AFIL uses online-generated failure trajectories as adaptive negative guidance, steering action generation away from failure-prone regions and toward more reliable success modes.

recovering from suboptimal or unseen states, resulting in brittle execution at inference time. Recent work has therefore highlighted the value of learning from failure data [Lu et al., 2025, Lin et al., 2025, Huang et al., 2025, Duan et al., 2024], showing that unsuccessful trajectories can improve robustness and task success rates. However, existing approaches often rely on handcrafted or predefined failure modes; for example, Duan et al. [2024] manually specifies categories such as incomplete grasp, slip, and translation failures for data generation. These designs require domain expertise and may fail to capture the diversity of realistic deployment-time failures. Moreover, recovery often depends on task-specific heuristics or real-time human intervention [Lin et al., 2025, Duan et al., 2024, Shi et al., 2024, Zhang et al., 2024], limiting scalability and generalization to unseen tasks or embodiments.

Most VLA policies adopt diffusion-based [Ho et al., 2020, Chi et al., 2025] or flow-based [Lipman et al., 2023] generative architectures [Kim et al., 2024, Black et al., 2024, Intelligence et al., 2025, Chi et al., 2025], owing to their ability to model multimodal action distributions and capture complex dependencies between perception and control. However, since most existing diffusion-/flow-based VLA models are trained only on successful demonstrations, sampling from the learned success-conditioned distribution may be insufficient for robust deployment, as the policy lacks explicit signals to avoid nearby failure-prone actions. In image generation, guided diffusion methods such as classifier guidance (CG) [Dhariwal and Nichol, 2021] and classifier-free guidance (CFG) [Ho and Salimans, 2021, Rombach et al., 2022, Chung et al., 2025, Fu and Li, 2025] improve conditional generation by steering samples toward desired semantic attributes. Negative prompting further extends this idea by repelling samples away from undesirable regions of the generation space [Gandikota et al., 2023, Schramowski et al., 2023, Koulischer et al., 2025]. This naturally motivates an analogous strategy for VLA learning: treating failure trajectories as negative guidance to encourage the policy to avoid action distributions associated with unsuccessful outcomes. Directly applying negative guidance to VLAs, however, is non-trivial [Pearce et al., 2023, Reuss et al., 2023]. Unlike image generation, robotic action generation is sequential and must produce temporally consistent actions conditioned on evolving observations. In addition, conventional negative guidance methods often rely on separate models for desired and undesired distributions, which is computationally expensive for VLA systems built on large vision-language model (VLM) backbones. These challenges motivate an efficient, unified failure-informed guidance mechanism tailored to diffusion- and flow-based VLA policies.

To address these challenges, we propose Adaptive Failure-Informed Learning (AFIL), an end-to-end framework for diffusion- and flow-matching VLA models that dynamically guides action generation using failure examples (*c.f.* Figure 1). AFIL learns from mixed success and failure data by jointly training Dual Action Generator (DAG)-VLA for successful and failed behaviors while sharing a VLM backbone for visual perception and semantic grounding. During sampling, AFIL iteratively queries both success and failure action generators and uses the predicted failure actions as adaptive negative guidance, steering the generative process away from undesirable regions of the action distribution. Extensive experiments on both in-domain and out-of-domain robotic manipulation tasks show that AFIL consistently improves task success rates and robustness over strong VLA baselines, while introducing only moderate additional parameters. Our contributions are summarized as follows:

- We introduce Adaptive Failure-Informed Learning (AFIL) for diffusion- and flow-based VLAs, using online-generated failure rollouts as a principled form of adaptive negative guidance for sequential action generation.
- We propose a Dual Action Generator (DAG)-VLA architecture that jointly learns from success and failure data while sharing a common VLM backbone, enabling efficient failure-aware policy learning with limited parameter overhead.
- We demonstrate that AFIL consistently improves success rates across diverse in-domain and out-of-domain manipulation tasks, spanning both short- and long-horizon settings, highlighting its effectiveness, efficiency, and generality.

2 Related Work

2.1 Robotic Manipulation via Imitation Learning

Recent advances in robotic manipulation have widely adopted imitation learning, particularly behavior cloning, to address tasks ranging from grasping and pick-and-place to dexterous in-hand manipulation and assembly. Many prior works train deep visuomotor policies, typically based on convolutional neural networks, using simulation data or human demonstrations to enable efficient policy learning from visual observations. For example, Zhang et al. [2018], Florence et al. [2019], Ross et al. [2011], Avigal et al. [2022] employ explicit policy learning by directly regressing from the observed state to the action space. In contrast, another line of work formulates policy learning implicitly by modeling action distributions with Energy-Based Models (EBMs), allowing more expressive representations of multimodal behaviors [Florence et al., 2021, LeCun et al., 2006, Grathwohl et al., 2020, Dai et al., 2019]. We refer readers to Tang et al. [2025] for a comprehensive survey of imitation- and reinforcement-learning-based methods for robotic policy learning.

2.2 Vision-Language-Action (VLA) Models

VLA models have emerged recently and demonstrated impressive generalization across diverse robotic tasks. Early large-scale systems such as RT-2 [Zitkovich et al., 2023], PaLM-E [Driess et al., 2023] and Octo [Octo Model Team et al., 2024] showed that policies conditioned on vision and language can follow semantic instructions, but they remain computationally expensive and closed-source. OpenVLA [Kim et al., 2024] provides a powerful open-source alternative, leveraging large-scale cross-embodiment data and pretrained for improved generalist manipulation. Following Kim et al. [2024], numerous work have been proposed to improve visual action representation learning, *e.g*. Zhu et al. [2025] introduces a self-supervised framework that learns robust visual-action representations by reconstructing masked goal images from language instructions. CoT-VLA [Zhao et al., 2025] and Embodied CoT [Zawalski et al., 2024] incorporate Chain-of-Thought (CoT) reasoning and demonstrate that generating intermediate reasoning steps prior to action can improve task success and transparency. Black et al. [2024, 2025] introduces π_0/$\pi_{0.5}$, a general-purpose robot foundation model that combines internet-scale vision-language pre-training with a novel flow-matching architecture to enable versatile, dexterous manipulation across diverse robotic embodiments and complex, multi-stage tasks. However, these approaches primarily learn from successful demonstrations. When the agent encounters an unexpected or erroneous state, VLA learned from success action space alone often fails to support recovery, as the reasoning process is not grounded in corrective experience. This limits its effectiveness in real-world settings where errors are inevitable.

2.3 Learning from Failure in Robotic Policies

Failure signals are typically discarded during VLA training, despite containing valuable information about unsafe actions and recovery strategies. Existing efforts to leverage failures can be broadly categorized into three areas. **(i) Failure Detection and Precursors.** Several works focus on identifying unsafe or failure-prone states. SAFE [Gu et al., 2026] introduces a multitask failure detection framework based on internal VLA representations, while UNISafe [Seo et al., 2025] proposes uncertainty-aware latent safety filters to prevent out-of-distribution failures. Other approaches identify precursors to failure using risk backpropagation or emergency-stop signals [Shangguan et al., 2025]. While effective for prediction, these methods typically halt execution without providing goal-directed recovery and are often limited to narrow manipulation settings. **(ii) Interactive and**

Physical Correction. Human-in-the-loop approaches incorporate external feedback to correct failures during execution. Shi et al. [2024] enables real-time language-based corrections, while Zhang et al. [2024] advocates physical intervention as a more natural interface. Although effective, these methods require continuous human supervision and do not scale to autonomous long-horizon deployment. **(iii) Autonomous Recovery and Optimization.** Recent work explores learning from failure using optimization and reinforcement learning. Fail2Progress [Huang et al., 2025] applies Stein variational inference to learn from failures but relies on a fixed skill library. From Mystery to Mastery [Sagar et al., 2025] combines RL-based exploration with VLM-based failure detection, though it focuses primarily on visual errors and offers limited insight into policy improvement. VLA with RL-based feedback, such as $\pi_{0.6}$ [Intelligence et al., 2025] or diagnostic systems like RoboFAC [Lu et al., 2025], demonstrate strong performance but require labor-intensive failure mode design and substantial in-domain data generation [Lu et al., 2025, Lin et al., 2025, Huang et al., 2025, Duan et al., 2024].

In contrast, our work targets data-efficient task completion for diffusion-/flow-based VLA models by providing two improvements: first we augment VLA training with failure-correction trajectories, enabling the policy to recover when failures occur. Second, improving sampling actions away from the failure manifold and sample in the failure-correction manifold when failure occurs.

2.4 Classifier-Free Guidance for Guided Diffusion Sampling

Classifier-Free Guidance (CFG) has demonstrated strong effectiveness in generative modeling [Ho and Salimans, 2021, Rombach et al., 2022, Chung et al., 2025, Fu and Li, 2025, Koulischer et al., 2025] by steering samples toward desired conditions through the contrast between conditional and unconditional predictions. Recent works have explored CFG-inspired strategies for guided diffusion sampling in control and policy generation settings [Reuss et al., 2023]. However, such approach relies on specialized numerical ODE solvers, making adaptation to existing pretrained VLA models nontrivial. More importantly, prior diffusion-based policy learning approaches do not explicitly exploit undesirable failure distributions, which provide informative counterexamples that can be leveraged through contrastive guidance objectives to further improve action generation.

Inspired by the idea of Negative Prompting for Text-to-Image (T2I) generation [Gandikota et al., 2023, Schramowski et al., 2023, Koulischer et al., 2025], we propose a failure-informed VLA learning mechanism that adaptively biases action sampling toward regions of the embedding space associated with successful outcomes and away from regions associated with failures. Rather than relying on explicit planners or safety filters, our approach leverages the diffusion step-wise distance between success and failure distributions to implicitly guide action sampling away from failure modes and toward a confined success-oriented action space. This allows the policy to remain flexible while avoiding failure-prone regions of the action manifold and recovering from unexpected states.

3 Proposed Methodology

In this section we describe our approach for Adaptive Failure-Informed Learning (AFIL) based on diffusion- and flow-based VLA models. Our method leverages two action generators: one trained on successful trajectories and another trained on failure trajectories. At inference/sampling time, the failure generator provides adaptive negatively prompted guidance that steers action sampling away from failure-prone regions of the action space. We first describe the dual action generator (DAG) architecture and training strategy, followed by adaptive failure-informed sampling mechanism. Please refer to Figure 2 for detailed illustration of the proposed AFIL pipeline.

3.1 Preliminaries

Vision-Language-Action (VLA) Models. We aim to learn vision-language-action (VLA) policies, $\pi_\theta(\boldsymbol{a}_t \mid \boldsymbol{o}_t, \ell)$, that generate continuous robot actions conditioned on visual observations and language instructions. At time step $t \in 1, \ldots, T$, where T denotes the length of a trajectory, the policy receives an observation $\boldsymbol{o}_t = (I_t, \boldsymbol{x}_t)$, where I_t denotes one or more RGB (or RGB-D) images and $\boldsymbol{x}_t$ denotes proprioceptive states, together with a language instruction ℓ. The policy outputs an action chunk $\boldsymbol{a}_{t:t+H} \in \mathbb{R}^{d \times H}$, where $\boldsymbol{a}_t^d$ is the d-DoF robot joint/gripper control vector (including joint states, joint angles and gripper states) at time t, H is the action horizon. VLA models [Kim et al., 2024, Black et al., 2025, 2024, Zhu et al., 2025] typically consist of a large-scale vision-language model

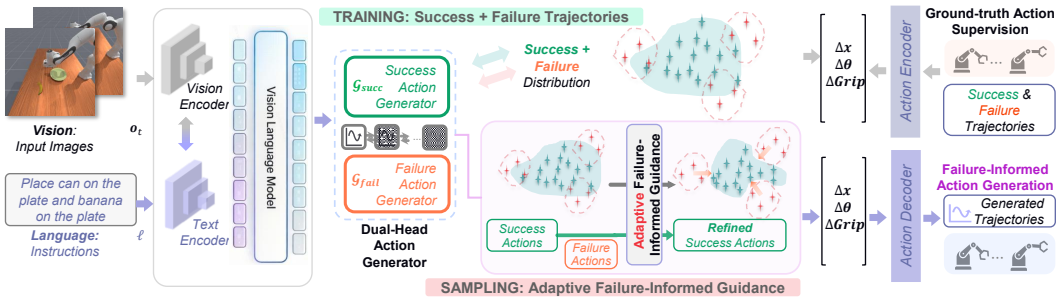


Figure 2: Dual Action Generator (DAG)-VLA with Adaptive Failure-Informed Learning (AFIL).

(VLM) backbone $\Phi_{\text{VLM}}(I_t, \ell)$ followed by a specialized action generation module $\mathcal{G}$. In this work, we focus on diffusion- and flow-based VLAs [Black et al., 2025, 2024, Zhu et al., 2025] due to their strong ability to model multimodal action distributions.

Classifier-Free Guidance Classifier-free guidance (CFG) [Chung et al., 2025] jointly train a diffusion model to handle both conditional and unconditional generation, moving the score estimate toward the conditional prediction to trade off sample diversity for higher visual fidelity and better command following. Specifically, Chung et al. [2025] applies Bayes rule to rewrite the sharpened posterior distribution $p_\eta(\mathbf{x}|\mathbf{c})$ at diffusion step η, given condition $\mathbf{c}$:

$$\nabla_{\mathbf{x}} \log p_\eta(\mathbf{x}|\mathbf{c}) = \nabla_{\mathbf{x}} \log p_\eta(\mathbf{x}) + \lambda(\nabla_{\mathbf{x}} \log p_\eta(\mathbf{x}|\mathbf{c}) - \nabla_{\mathbf{x}} \log p_\eta(\mathbf{x})) \tag{1}$$

Parameterizing with the score function [Ho et al., 2020], the classifier free-guided score $\mathbf{s}^*_{\text{CFG}}$ is a linear combination of unconditional score s_{uc} and conditional score s_c:

$$\mathbf{s}^*_{\text{CFG}} = s_{\mathbf{uc}} + \lambda(s_{\mathbf{c}} - s_{\mathbf{uc}}) \tag{2}$$

For Negative Prompting (NP) [Gandikota et al., 2023, Schramowski et al., 2023, Koulischer et al., 2025], the posterior $\log p_t(\mathbf{x}|\mathbf{c}^+)$ ($\mathbf{c}^+$, and $\mathbf{c}^-$ denote positive and negative prompt respectively):

$$\nabla_{\mathbf{x}} \log p_t(\mathbf{x}|\mathbf{c}^+) = \nabla_{\mathbf{x}} \log p_t(\mathbf{x}) - \lambda(\nabla_{\mathbf{x}} \log p_t(\mathbf{x}|\mathbf{c}^-) - \nabla_{\mathbf{x}} \log p_t(\mathbf{x})) \tag{3}$$

The NP guided score $\mathbf{s}^*_{\text{NP}}$ linearly combines negative prompt score $s_{\mathbf{c}^-}$ and unconditional score $s_{\mathbf{uc}}$:

$$\mathbf{s}^*_{\text{NP}} = s_{\mathbf{uc}} - \lambda(s_{\mathbf{c}^-} - s_{\mathbf{uc}}) \tag{4}$$

The static guidance scale can be further set to $\lambda(\mathbf{x}, \eta) \propto \frac{p_\eta(\mathbf{c}^-|\mathbf{x})}{1-p_\eta(\mathbf{c}^-|\mathbf{x})}$ values for state-dependent guidance [Koulischer et al., 2025].

3.2 Dual Action Generator (DAG) Architecture

In conventional success-only VLA training, the policy models an action distribution $p_{\text{succ}}(\boldsymbol{a} \mid \boldsymbol{o}, \ell)$ using large-scale successful robot demonstrations and trajectories [O'Neill et al., 2024, Black et al., 2025]. However, failure trajectories also contain valuable information about action patterns that lead to unsuccessful outcomes. We introduce an additional lightweight action generator that operates in parallel with $\mathcal{G}$, extending conventional VLA architectures to explicitly model both success and failure action distributions. Let $p_{\text{fail}}(\boldsymbol{a} \mid \boldsymbol{o}, \ell)$ denotes the distribution of actions associated with failure outcomes. Our objective is to (i) learn both success and failure action manifolds, and (ii) leverage them during inference to guide action sampling away from failure.

Specifically, our proposed dual action generator (DAG) VLA composes of, (i) a *Success* action generator $\mathcal{G}_{\text{succ}}$ trained only on successful trajectories, and (ii) a *Failure* action generator $\mathcal{G}_{\text{fail}}$ trained only on failure trajectories. Both generators share the same vision-language backbone Φ_{VLM}. Given vision-language features $\mathbf{h} = \Phi_{\text{VLM}}(\boldsymbol{o}, \ell)$, the generators induce two generative action models, $p_{\theta_s}(\boldsymbol{a} \mid \boldsymbol{o}, \ell)$ and $p_{\theta_f}(\boldsymbol{a} \mid \boldsymbol{o}, \ell)$, parameterized by their action-head parameters θ_s and θ_f, respectively.

Failure Data Generation. Existing approaches for leveraging mixed success–failure datasets often rely on handcrafted failure taxonomies [Duan et al., 2024, Lin et al., 2025, Huang et al., 2025]. For

example, prior work categorizes errors such as unstable grasps or kinematic slips to guide synthetic data generation [Duan et al., 2024], but such heuristics are inherently limited in capturing the stochastic and diverse nature of real-world failures. In contrast, we incorporate online-inferred failure trajectories obtained directly from rollouts of a learned or pretrained VLA under task-conditioned execution. This design provides two advantages: it captures a richer, more realistic failure distribution, including emergent and compounding error modes, and it eliminates the need for manual annotation or handcrafted perturbation design by leveraging naturally occurring rollout failures. Consequently, the resulting failure data better approximates the on-policy state-action distribution, improving robustness and generalization. These failure trajectories can further be corrected via motion-planner-based replanning, enabling effective augmentation of success-only training data and improving overall VLA performance (see Section 4).

Training Objectives. Let training dataset $\mathcal{D} = \{\mathcal{D}_s, \mathcal{D}_f\}$, where $\mathcal{D}_s$ and $\mathcal{D}_f$ contain success and failure trajectories respectively. We train each action generator on its respective dataset using the same generative objective but with disjoint supervision. For diffusion-based policies [Ho et al., 2020, Black et al., 2024], each action generator predicts noise $\epsilon_\theta(\boldsymbol{a}^\eta, \boldsymbol{o}, \ell, \eta)$ at diffusion step η and minimizes,

$$\mathcal{L}_{\text{diff}}(\theta; \mathcal{D}) = \mathbb{E}_{(\boldsymbol{o},\ell,\boldsymbol{a})\sim\mathcal{D}}\, \mathbb{E}_{\eta,\epsilon}\left[\|\epsilon - \epsilon_\theta(\boldsymbol{a}^\eta, \boldsymbol{o}, \ell, \eta)\|_2^2\right]. \tag{5}$$

Alternatively, for flow-based policies [Black et al., 2025] the action head predicts a velocity field $v_\theta(\boldsymbol{a}^\eta, \boldsymbol{o}, \ell, \eta)$ that transports samples from noise to the data distribution. The model is trained to match the ground-truth velocity between noisy actions and target actions using the standard flow-matching objective. Our method does not modify underlying generative VLA training objective.

3.3 Adaptive Failure-Informed Sampling

At inference, naive sampling from p_{θ_s} can propose actions that drift into failure-prone regions. Inspired by Negative Prompting for Text-to-Image (T2I) generation [Gandikota et al., 2023, Schramowski et al., 2023, Koulischer et al., 2025], we propose to perform *adaptive failure-informed sampling*, given the learned success and failure distributions from the DAG (Sec. 3.2) to fully explore the repulsive nature of the undesired distributions, which is ignored in conventional VLA learning.

Let $\epsilon_{\text{succ}}(\boldsymbol{a}^\eta, \boldsymbol{o}, \ell, \eta)$ denote the predicted noise from the success generator $\mathcal{G}_{\text{succ}}$ and $\epsilon_{\text{fail}}(\boldsymbol{a}^\eta, \boldsymbol{o}, \ell, \eta)$ denotes the predicted noise by the failure action generator at diffusion time step η. Here we use diffusion models for illustration purpose. For flow-matching models, $\epsilon_{\text{succ}}(\boldsymbol{a}^\eta, \boldsymbol{o}, \ell, \eta)$ and $\epsilon_{\text{fail}}(\boldsymbol{a}^\eta, \boldsymbol{o}, \ell, \eta)$ can be exchanged with velocity field $v_{\text{succ}}(\boldsymbol{a}^\eta, \boldsymbol{o}, \ell, \eta)$ and $v_{\text{fail}}(\boldsymbol{a}^\eta, \boldsymbol{o}, \ell, \eta)$ respectively. In the following, we omit the explicit conditioning on $(\boldsymbol{o}, \ell)$ and the timestep argument η, and write $\epsilon_{\text{succ}}(\boldsymbol{a}^\eta)$ and $\epsilon_{\text{fail}}(\boldsymbol{a}^\eta)$ for notational simplicity.

Recall Equation 4, Negative Prompting (NP) guidance can be applied to steer the diffusion sampling more accurately towards desired positive distributions by linearly combining negative prompt score and unconditional score. For the proposed DAG-VLA, we have failure-informed (FI) score with *adaptive guidance* to steer the diffusion sampling away from failure:

$$\epsilon^*_{\text{FI}} = \epsilon_{\text{succ}}(\boldsymbol{a}^\eta, \varnothing) - \lambda_\eta(\boldsymbol{a}^\eta)\,\left(\epsilon_{\text{fail}}(\boldsymbol{a}^\eta) - \epsilon_{\text{succ}}(\boldsymbol{a}^\eta, \varnothing)\right) \tag{6}$$

$$\Leftrightarrow\ \epsilon_{\text{succ}}(\boldsymbol{a}^\eta, \varnothing) - \hat{\lambda}_\eta(\boldsymbol{a}^\eta)\epsilon_{\text{fail}}(\boldsymbol{a}^\eta) \tag{7}$$

Diffusion guidance in Equation 7 approximates score combination: $\nabla_{\boldsymbol{a}} \log p_{\text{succ}}(\boldsymbol{a}^\eta) - \hat{\lambda}_\eta(\boldsymbol{a}^\eta)\nabla_{\boldsymbol{a}} \log p_{\text{fail}}(\boldsymbol{a}^\eta)$, we choose $\hat{\lambda}_\eta(\boldsymbol{a}^\eta)$ to be proportional to success-failure distribution distance $D(p_{\text{succ}}, p_{\text{fail}})$, where $D(\cdot\,,\cdot)$ is the divergence metric between two distributions.

Intuitively, this quantity captures how distinguishable successful and failed behaviors are in the local state: when the two scores are highly aligned, indicating that success and failure modes are locally ambiguous, strong negative guidance may suppress valid actions. Conversely, when the scores are dissimilar or oppositely aligned, the failure model provides complementary information about undesirable directions, making stronger negative guidance beneficial for steering the trajectory away from failure modes. This adaptive mechanism is particularly important in robotic manipulation, where successful and failed trajectories may share overlap before diverging. *E.g.*, at the beginning of

an episode, both successful and failed rollouts may move from the home pose toward the target object using similar reaching motions. In such states, applying strong failure repulsion would be undesirable, the guidance scale should remain small. In contrast, once a clear failure mode emerges, such as dropping, unstable grasping, or drifting away from the target, the success and failure predictions become more distinguishable. At these stages, increasing the guidance scale allows the policy to actively suppress failure-prone actions and bias sampling toward corrective behaviors. This yields a simple, fully local, and computation-efficient mechanism that adaptively balances attraction toward successful behaviors and repulsion from failure modes only when such correction is necessary.

Specifically we adopt a cosine similarity-based strategy to adaptively modulate the strength of negative guidance during diffusion sampling, *i.e.*,

$$\hat{\lambda}_\eta(\boldsymbol{a}^\eta) = \alpha(\, 1 - \cos(\epsilon_{\text{succ}}, \epsilon_{\text{fail}})\,), \tag{8}$$

where α is the guidance scale. While several metrics can quantify the divergence between success and failure modes, we found the cosine distance to be uniquely suited for the adaptive weighting of VLA diffusion scores, considering several aspects: a) Distribution-based metrics such as Maximum Mean Discrepancy (MMD) or Jensen-Shannon Divergence (JSD) provide theoretically robust measures of divergence but require processing batches of trajectories at each denoising step. For real-time VLA applications, the overhead of calculating kernel matrices or density estimations at every iteration η is computationally prohibitive. b) In robotics and motion synthesis, the "semantic" intent of an action is encoded more heavily in the angular orientation of the joint-space or end-effector vector than in its absolute coordinate-wise difference. Metrics like Manhattan (L_1) or Chebyshev distance treat each dimension independently and are sensitive to the coordinate system's scaling. Cosine distance, by contrast, captures the global "intent" of the motion, muting the guidance when the success and failure trajectories are semantically aligned and amplifying it only when they represent distinct behavioral choices. Please see supplementary material for mathematical justifications.

4 Evaluation

To evaluate the proposed AFIL framework, we conduct experiments on both in-domain and out-of-domain manipulation tasks across multiple simulation environments, including ManiSkill [Tao et al., 2025, Gu et al., 2023] and LIBERO [Liu et al., 2023]. We additionally provide real-robot validation on a tabletop manipulation task with a Trossen ViperX 300S 6-DoF robotic arm. All experiments are built upon the $\pi_{0.5}$ architecture [Black et al., 2025]; however, AFIL is model-agnostic and can be readily integrated with diffusion- or flow-matching-based VLA policies. See supplementary material for additional implementation, experiment details and parameter/runtime analysis.

4.1 Environment Setups

We evaluate DAG-VLA under two experimental settings:

i) Maniskill [Tao et al., 2025]: We perform both in-domain and out-of-domain evaluations in the ManiSkill environment using a Franka Emika Panda 7-DoF robot arm. For in-domain evaluation, we consider three short-horizon tasks - *stack red cube on green cube*, *lift the peg upright*, and *pull cube to the red and white target* — as well as two long-horizon tasks: *place bowl on the plate and spoon on the bowl*, and *place can on the plate and banana on the plate*. We train our DAG-VLA on 1500 success, 1000 failure-corrected, and 1500 failure trajectories. For in domain evaluation, we evaluate DAG-VLA on 50 initialization configurations (varying initial states, object/robot poses) with 3 runs for each configuration, *i.e.* 50×3=150 rollouts, and report the absolute success rate for each task.

For out-of-domain evaluation, we assess generalization to unseen objects with different geometries and textures in cluttered tabletop scenes. Specifically, we evaluate two short-horizon tasks: **Lift Cylinder** (*lift the cylinder upright*, with unseen geometry and color) and **Stack Cubes** (*stack red cube on green cube*, in a cluttered scene with distractors), along with one long-horizon task, **Fruit Sorting** (*place the lemon and strawberry into a blue ceramic bowl*, with unseen objects and color). For out-of-domain tasks, we evaluate on 25 (configs)×3 (runs) = 75 rollouts for each task. Please refer to Figure 3 for visualizations of the task setups in Maniskill.

ii) LIBERO [Liu et al., 2023]: To enable fair comparison with prior work, we follow the standard LIBERO benchmark evaluation protocol [Kim et al., 2024, Black et al., 2025]. The benchmark

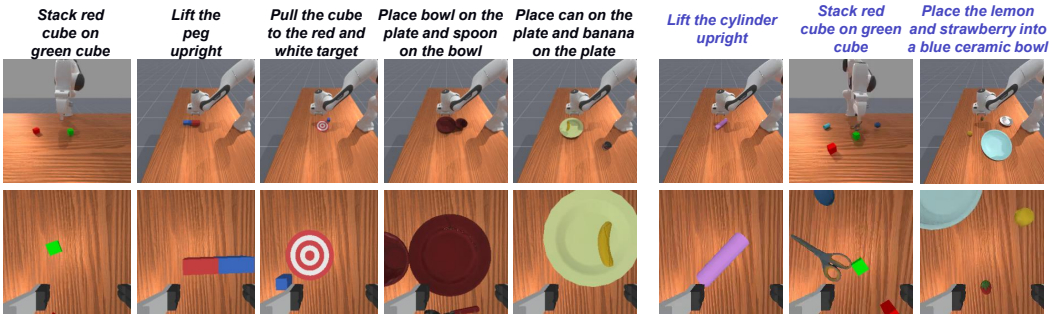


Figure 3: Visualization of in-domain (left) and out-of-domain (right) manipulation task setup. Row 1: exterior camera image, row 2: wrist camera image. See supplementary for more visualizations.

comprises four task suites, each containing 10 tasks with 50 human-teleoperated demonstrations. These suites capture complementary generalization challenges: *LIBERO-Spatial* (varying layouts), *LIBERO-Object* (varying object identities), *LIBERO-Goal* (varying task objectives), and *LIBERO-Long (LIBERO-10)* (long-horizon tasks with diverse objects, layouts, and goals). We collect failure trajectories from pretrained $\pi_{0.5}$ (finetuned on LIBERO) rollout executions and use them to train the proposed DAG-VLA model. We report success rate as the average over 3 random seeds x 500 rollouts each (10 tasks $\times$ 50 rollouts per task) following Kim et al. [2024].

4.2 In-Domain Evaluation

4.2.1 Manipulation Tasks in Maniskill

In Table 1, we present success rates of proposed DAG-VLA under various ablated settings. Success-only follows standard VLA training, where $\pi_{0.5}$ is trained solely on successful demonstration trajectories. Success + Failure Correction augments this data by first executing the success-only policy to collect failure cases, which are then corrected via motion-planner-based replanning and used for fine-tuning. Building on this, Static-FIL trains the proposed DAG-VLA using both success data and (corrected and raw) failure trajectories, and applies failure-informed sampling at inference with a fixed guidance strength λ (empirically set to 0.05 which works best for this setting; see supplementary material for ablations). Finally, Adaptive-FIL uses the same DAG-VLA training data but replaces the static weighting with adaptive failure-informed sampling (Eq. 7 and Eq. 8), where the guidance scale is adaptively adjusted during rollout based on failure signals. Guidance strength α (*c.f*. Eq. 8) is set to 1.0, please see next section for ablation studies.

Across both short- and long-horizon manipulation tasks, the proposed DAG-VLA with Adaptive Failure-Informed Learning (Adaptive-FIL) consistently achieves the strongest performance. Compared to training with success-only demonstrations, incorporating failure correction already yields substantial gains, particularly on less challenging tasks such as *Pull cube to the red and white target*. Introducing static failure-informed sampling further improves results, demonstrating the benefit of leveraging failure signals during inference. However, Adaptive-FIL provides the most consistent and significant improvement across all tasks, outperforming both static sampling and all baselines. Notably, it achieves the largest gains on long-horizon tasks, where compounding errors are more prevalent (e.g., +7.4% and +12.0% over static-FIL on the two long-horizon settings). These results indicate that dynamically adapting failure guidance during rollout is crucial for robust long-horizon, multi-step manipulation and generalization in complex environments.

Ablation Study on Guidance Scale We conduct an ablation study to examine how the scale of the adaptive guidance, α (*c.f*. Eq. 8), affects performance on two manipulation tasks: *stack red cube on green cube*, a short-horizon task, and *place can on the plate and banana on the plate*, in Table 2.

Ablation Study on Failure Correction Data Failure correction and failure-informed guidance serve distinct roles. Corrected trajectories augment the positive distribution with recovery behaviors, whereas raw failure trajectories train the failure generator and provide negative guidance. AFIL does not mathematically require corrected trajectories: the success generator can be trained on the original successful demonstrations, while the failure generator is trained on raw rollout failures.

Table 1: Evaluation and ablation study of proposed DAG-VLA on various manipulation tasks in Maniskill environments. For each task, success rates (%) are evaluated over 50 configs $\times$ 3 runs.

TASK		Success Only	Success + Failure Correction	DAG-VLA + Static-FIL ($\lambda = 0.05$)	DAG-VLA + Adaptive-FIL ($\alpha = 1.0$)
Short Horizon	*Stack red cube on green cube*	71.3	76.7	80.0	**84.7**
	Lift the peg upright	70.7	80.0	86.7	**90.7**
	Pull cube to the red and white target	60.0	92.0	94.7	**98.0**
Long Horizon	*Place bowl on the plate and spoon on the bowl*	38.6	45.3	49.3	**56.7**
	Place can on the plate and banana on the plate	39.3	56.0	59.3	**71.3**

Table 2: Ablation study on guidance scale α for AFIL. Success rates reported in %.

Task	$\alpha = 0.5$	$\alpha = 1.0$	$\alpha = 2.0$	$\alpha = 5.0$
Stack red cube on green cube	84.0	**84.7**	79.3	76.0
Place can on the plate and banana on the plate	68.7	**71.3**	69.3	66.0

We therefore perform an ablation comparing DAG-VLA trained with and without corrected trajectories under the AFIL evaluation setting on one short-horizon task and one long-horizon task in Table 3. The results show consistent improvement of our introduced AFIL (adaptive negative guidance) with and without failure correction data.

Table 3: Success rates (%) with and without AFIL under different failure-correction settings.

TASK	No Failure Correction		With Failure Correction	
	w/o AFIL	**w/ AFIL**	w/o AFIL	**w/ AFIL**
Lift the Peg Upright	71.3	**81.3**	76.7	**90.7**
Place Bowl on Plate and Spoon on Bowl	38.6	**50.7**	45.3	**56.7**

4.2.2 Benchmark Evaluation on LIBERO

In Table 4, we compared proposed DAG-VLA ($\pi_{0.5}$ backbone) with existing VLA backbones on LIBERO benchmark. Across the LIBERO benchmark suites, DAG-VLA with AFIL consistently achieves the strongest performance among all methods. Compared to prior VLA models - including Diffusion Policy, Octo, OpenVLA, CoT-VLA, and a strong $\pi_{0.5}$ baseline - our method improves performance across all four task suites, with particularly notable gains on the most challenging LIBERO-10 *long-horizon* setting. While $\pi_{0.5}$ already provides a strong baseline, incorporating AFIL further improves average performance from 96.9% to 98.4%, indicating that even strong pretrained policies benefit from adaptive failure-informed guidance. These results highlight the effectiveness of AFIL in improving robustness and long-horizon generalization by dynamically adjusting failure guidance during inference, rather than relying on fixed or purely success-driven rollouts.

Table 4: Performance comparison on LIBERO [Liu et al., 2023] benchmark. Success rates in %.

Method	**LIBERO Spatial**	**LIBERO Object**	**LIBERO Goal**	**LIBERO 10**	**Average**
Diffusion Policy from scratch [Chi et al., 2025]	78.3 ± 1.1	92.5 ± 0.7	68.3 ± 1.2	50.5 ± 1.3	72.4 ± 0.7
Octo fine-tuned [Octo Model Team et al., 2024]	78.9 ± 1.0	85.7 ± 0.9	84.6 ± 0.9	51.1 ± 1.3	75.1 ± 0.6
OpenVLA fine-tuned [Kim et al., 2024]	84.7 ± 0.9	88.4 ± 0.8	79.2 ± 1.0	53.7 ± 1.3	76.5 ± 0.6
CoT-VLA-7B [Zhao et al., 2025]	87.5 ± 1.4	91.6 ± 0.5	87.6 ± 0.6	69.0 ± 0.8	81.1 ± 0.6
$\pi_{0.5}$ [Black et al., 2025]	$98.8 \pm$ na	$98.2 \pm$ na	$98.0 \pm$ na	$92.4 \pm$ na	$96.9 \pm$ na
DAG-VLA$_{\pi_{0.5}}$ + **AFIL**	$\mathbf{99.8 \pm 0.4}$	$\mathbf{99.7 \pm 0.8}$	$\mathbf{98.1 \pm 0.4}$	$\mathbf{95.8 \pm 1.2}$	$\mathbf{98.4 \pm 0.7}$

4.3 Out-of-Domain (OOD) Evaluation

For OOD evaluation, we test generalization to unseen object instances, novel geometries/textures, and cluttered tabletop environments (Table 5). The success-only $\pi_{0.5}$ policy degrades substantially under these distribution shifts, while incorporating corrected failure data improves performance across all tasks. Our DAG-VLA$_{\pi_{0.5}}$ with AFIL achieves the best results in all settings, showing consistent gains on both short- and long-horizon tasks. These results demonstrate that adaptive failure-informed guidance improves robustness to visual, geometric, and task-level distribution shifts.

Table 5: Performance comparison on out-of-domain manipulation tasks. Success rates (%) reported over 25 configs $\times$ 3 runs for each task.

Method	**Short Horizon**		**Long Horizon**
	Lift Cylinder (unseen object/color)	**Stack Cubes** (cluttered scene)	**Fruit Sorting** (unseen objects + clutter)
$\pi_{0.5}$ (success only)	14.7	16.0	10.7
$\pi_{0.5}$ (success + failure correction)	53.3	41.3	54.7
DAG-VLA$_{\pi_{0.5}}$ + **AFIL**	**62.7**	**48.0**	**64.0**

4.4 Real-Robot Validation

To examine whether the benefit of failure-informed guidance persists beyond simulation, we conduct a controlled proof-of-concept evaluation on a Trossen ViperX 300S 6-DoF robotic arm using the tabletop manipulation task, *Pick up the yellow cube*. We collected 50 successful and failure-corrected trajectories and 50 raw failure trajectories to fine-tune DAG-VLA from the base $\pi_{0.5}$ checkpoint. We then evaluated the same trained DAG-VLA model with and without adaptive negative guidance (AFIL) over 50 real-robot trials, covering five initial cube configurations with 10 trials per configuration. The success rates are reported in Table 6. AFIL improves the average real-robot success rate from 0.50 to 0.92, and improves performance across all five initial configurations. These results provide real-world evidence that the learned failure-conditioned guidance transfers beyond simulation and can improve robustness under physical perception, contact, and actuation uncertainties. Please see supplementary material for workspace setup, qualitative results, and rollout visualizations.

Table 6: Performance comparison on real-world tabletop manipulation task. Success rates (0.00-1.00) are evaluated over 5 configurations with 10 trials per configuration.

Method	Config. 0	Config. 1	Config. 2	Config. 3	Config. 4	Average
DAG-VLA$_{\pi_{0.5}}$, No FIL	0.20	0.00	0.70	0.90	0.70	0.50
DAG-VLA$_{\pi_{0.5}}$ + AFIL	**1.00**	**0.80**	**0.90**	**1.00**	**0.90**	**0.92**

5 Conclusion

We introduced Adaptive Failure-Informed Learning (AFIL), a framework for improving diffusion- and flow-based VLA policies by using failure trajectories as adaptive negative guidance. AFIL augments conventional success-only VLA learning with a Dual Action Generator architecture that jointly models successful and failed behaviors while sharing a common vision-language backbone. During inference, the failure generator adaptively guides action sampling away from failure-prone regions, improving robustness without requiring handcrafted failure modes, task-specific recovery heuristics, or separate large-scale VLA models. Across ManiSkill and LIBERO evaluations, AFIL consistently improves success rates over strong VLA baselines, with particularly notable gains on long-horizon and out-of-domain manipulation tasks. These results show that explicitly modeling failure distributions can provide an efficient and scalable mechanism for robust action generation. We believe failure-informed guidance offers a promising direction for developing VLA policies that better generalize to realistic deployment settings where errors, distribution shifts, and recovery demands are unavoidable.

References

Yahav Avigal, Lars Berscheid, Tamim Asfour, Torsten Kroger, and Ken Goldberg. Speedfolding: Learning efficient bimanual folding of garments. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2022.

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.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y Galliker, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In *Annual Conference on Robot Learning*, 2025.

Remi Cadene, Simon Alibert, Francesco Capuano, Michel Aractingi, Adil Zouitine, Pepijn Kooijmans, Jade Choghari, Martino Russi, Caroline Pascal, Steven Palma, Mustafa Shukor, Jess Moss, Alexander Soare, Dana Aubakirova, Quentin Lhoest, Quentin Gallouédec, and Thomas Wolf. LeRobot: An open-source library for end-to-end robot learning. In *International Conference on Learning Representations*, 2026. URL `https://arxiv.org/abs/2602.22818`.

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. *The International Journal of Robotics Research*, 2025.

Hyungjin Chung, Jeongsol Kim, Geon Yeong Park, Hyelin Nam, and Jong Chul Ye. CFG++: Manifold-constrained classifier free guidance for diffusion models. In *International Conference on Learning Representations*, 2025.

Bo Dai, Zhen Liu, Hanjun Dai, Niao He, Arthur Gretton, Le Song, and Dale Schuurmans. Exponential family estimation via adversarial dynamics embedding. In *Neural Information Processing Systems*, 2019.

Prafulla Dhariwal and Alex Nichol. Diffusion models beat GANs on image synthesis. In *International Conference on Neural Information Processing Systems*, 2021.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. PaLM-E: an embodied multimodal language model. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

Jiafei Duan, Wilbert Pumacay, Nishanth Kumar, Yi Ru Wang, Shulin Tian, Wentao Yuan, Ranjay Krishna, Dieter Fox, Ajay Mandlekar, and Yijie Guo. Aha: A vision-language-model for detecting and reasoning over failures in robotic manipulation. *arXiv preprint arXiv:2410.00371*, 2024.

Pete Florence, Corey Lynch, Andy Zeng, Oscar A Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *Annual Conference on Robot Learning*, 2021.

Peter R. Florence, Lucas Manuelli, and Russ Tedrake. Self-supervised correspondence in visuomotor policy learning. *IEEE Robotics and Automation Letters*, 2019.

Xiaomeng Fu and Jia Li. Tcfg: Truncated classifier-free guidance for efficient and scalable text-to-image acceleration. In *IEEE/CVF International Conference on Computer Vision*, 2025.

Rohit Gandikota, Joanna Materzynska, Jaden Fiotto-Kaufman, and David Bau. Erasing concepts from diffusion models. In *IEEE/CVF International Conference on Computer Vision*, 2023.

Will Grathwohl, Kuan-Chieh Wang, Joern-Henrik Jacobsen, David Duvenaud, and Richard Zemel. Learning the stein discrepancy for training and evaluating energy-based models without sampling. In *International Conference on Machine Learning*, 2020.

Jiayuan Gu, Fanbo Xiang, Xuanlin Li, Zhan Ling, Xiqiang Liu, Tongzhou Mu, Yihe Tang, Stone Tao, Xinyue Wei, Yunchao Yao, Xiaodi Yuan, Pengwei Xie, Zhiao Huang, Rui Chen, and Hao Su. ManiSkill2: A unified benchmark for generalizable manipulation skills. In *International Conference on Learning Representations*, 2023. Apache-2.0 license.

Qiao Gu, Yuanliang Ju, Shengxiang Sun, Igor Gilitschenski, Haruki Nishimura, Masha Itkina, and Florian Shkurti. SAFE: Multitask failure detection for vision-language-action models. In *Annual Conference on Neural Information Processing Systems*, 2026.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS Workshop on Deep Generative Models and Downstream Applications*, 2021.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.

Yixuan Huang, Novella Alvina, Mohanraj Devendran Shanthi, and Tucker Hermans. Fail2progress: Learning from failures with stein variational inference for robot manipulation. In *Workshop on Out-of-Distribution Generalization in Robotics at RSS*, 2025.

Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, et al. $\pi^*_{0.6}$: a vla that learns from experience. *arXiv preprint arXiv:2511.14759*, 2025.

Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 2020.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model. In *Annual Conference on Robot Learning*, 2024.

Felix Koulischer, Johannes Deleu, Gabriel Raya, Thomas Demeester, and Luca Ambrogioni. Dynamic negative guidance of diffusion models. In *International Conference on Learning Representations*, 2025.

Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, Fujie Huang, et al. A tutorial on energy-based learning. *Predicting structured data*, 2006.

Zijun Lin, Jiafei Duan, Haoquan Fang, Dieter Fox, Ranjay Krishna, Cheston Tan, and Bihan Wen. Failsafe: Reasoning and recovery from failures in vision-language-action models. *arXiv preprint arXiv:2510.01642*, 2025.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 2023.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. RDT-1b: a diffusion foundation model for bimanual manipulation. In *International Conference on Learning Representations*, 2025.

Weifeng Lu, Minghao Ye, Zewei Ye, Ruihan Tao, Shuo Yang, and Bo Zhao. Robofac: A comprehensive framework for robotic failure analysis and correction. *arXiv preprint arXiv:2505.12224*, 2025.

Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *Annual Conference on Robot Learning*, 2021.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Charles Xu, Jianlan Luo, Tobias Kreiman, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In *Proceedings of Robotics: Science and Systems*, 2024.

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 *IEEE International Conference on Robotics and Automation*, 2024.

Tim Pearce, Tabish Rashid, Anssi Kanervisto, Dave Bignell, Mingfei Sun, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Ida Momennejad, Katja Hofmann, and Sam Devlin. Imitating human behaviour with diffusion models. In *International Conference on Learning Representations*, 2023.

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.

Moritz Reuss, Maximilian Li, Xiaogang Jia, and Rudolf Lioutikov. Goal conditioned imitation learning using score-based diffusion policies. In *Robotics: Science and Systems*, 2023.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics*, 2011.

Som Sagar, Jiafei Duan, Sreevishakh Vasudevan, Yifan Zhou, Heni Ben Amor, Dieter Fox, and Ransalu Senanayake. From mystery to mastery: Failure diagnosis for improving manipulation policies. In *Workshop on Out-of-Distribution Generalization in Robotics at RSS*, 2025.

Patrick Schramowski, Manuel Brack, Björn Deiseroth, and Kristian Kersting. Safe latent diffusion: Mitigating inappropriate degeneration in diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

Junwon Seo, Kensuke Nakamura, and Andrea Bajcsy. Uncertainty-aware latent safety filters for avoiding out-of-distribution failures. In *9th Annual Conference on Robot Learning*, 2025.

Zeyu Shangguan, Rajas Chitale, Rutvik Patel, Satyandra Gupta, and Daniel Seita. Identifying precursors to failures in robotic lift-and-place tasks. In *Workshop on Out-of-Distribution Generalization in Robotics at RSS*, 2025.

Lucy Xiaoyang Shi, Zheyuan Hu, Tony Z Zhao, Archit Sharma, Karl Pertsch, Jianlan Luo, Sergey Levine, and Chelsea Finn. Yell at your robot: Improving on-the-fly from language corrections. *arXiv preprint arXiv:2403.12910*, 2024.

Chen Tang, Ben Abbatematteo, Jiaheng Hu, Rohan Chandra, Roberto Martín-Martín, and Peter Stone. Deep reinforcement learning for robotics: a survey of real-world successes. In *AAAI Conference on Artificial Intelligence*, AAAI'25/IAAI'25/EAAI'25, 2025.

Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse-Kai Chan, Yuan Gao, Xuanlin Li, Tongzhou Mu, Nan Xiao, Arnav Gurha, Viswesh N, Yong Woo Choi, Yen-Ru Chen, Zhiao Huang, Roberto Calandra, Rui Chen, Shan Luo, and Hao Su. ManiSkill3: GPU parallelized robot simulation and rendering for generalizable embodied AI. In *7th Robot Learning Workshop: Towards Robots with Human-Level Abilities*, 2025. URL `https://openreview.net/forum?id=GgTxudXaU8`. Apache-2.0 license.

Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning. In *Annual Conference on Robot Learning*, 2024.

Chuye Zhang, Yifei Simon Shao, Harshil Parekh, Junyao Shi, Pratik Chaudhari, Vijay Kumar, and Nadia Figueroa. Don't yell at your robot: Physical correction as the collaborative interface for language model powered robots. *arXiv preprint arXiv:2412.12602*, 2024.

Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *IEEE International Conference on Robotics and Automation*, 2018.

Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, Ankur Handa, Tsung-Yi Lin, Gordon Wetzstein, Ming-Yu Liu, and Donglai Xiang. CoT-VLA: Visual chain-of-thought reasoning for vision-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Xueyang Zhou, Yangming Xu, Guiyao Tie, Yongchao Chen, Guowen Zhang, Duanfeng Chu, Pan Zhou, and Lichao Sun. LIBERO-PRO: Towards robust and fair evaluation of vision-language-action models beyond memorization. *[arXiv preprint arXiv:2510.03827]*, 2025.

Chaoran Zhu, Hengyi Wang, Yik Lung Pang, and Changjae Oh. Lava-man: Learning visual action representations for robot manipulation. In *Conference on Robot Learning*, 2025.

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*, 2023.

Supplementary Material

1 Implementation Details

We augment the $\pi_{0.5}$ VLA architecture with Dual Action Generator (DAG), training the proposed DAG-VLA$_{\pi_{0.5}}$ using both success and failure trajectories (*c.f.* Section 3.2), and applying failure-informed sampling during inference (*c.f.* Section 3.3) for action rollout. During sampling, we use positively prompted noise $\mathbf{v}_{\text{succ}}(\boldsymbol{a}^\eta)$ to replace unguided model $\mathbf{v}_{\text{succ}}(\boldsymbol{a}^\eta, \varnothing)$ (Eq. 7) following Schramowski et al. [2023], Koulischer et al. [2025].

Data preprocessing, including conversion to the LeRobot format [Cadene et al., 2026], and normalization follow Black et al. [2025]. We set the action dimension to 32 and the action horizon to 30 steps. Training is initialized from the public $\pi_{0.5}$ base checkpoint and runs for 300k steps with a batch size of 4. All experiments are conducted on 4 NVIDIA A100-SXM4-80GB GPUs, with a total training time of approximately 8 hours.

2 Qualitative Results

Please refer to the supplementary videos for qualitative results. For in-domain and out-of-domain evaluations in the ManiSkill environment (corresponding to Section 4.2.1 and Section 4.2.3 in the main paper), we visually compare the proposed Dual Action Generator (DAG)-VLA under three settings: training with success-only data, training with success plus failure-correction data, and with Adaptive Failure-Informed Learning (AFIL). Quantitative comparisons are reported in Table 1 of the main paper. The videos show that VLAs trained only on successful demonstrations, following the conventional setting, are prone to failure and lack effective recovery capability. Incorporating failure-correction data mitigates this issue, while DAG-VLA with AFIL enables more accurate failure avoidance and correction, leading to higher task success rates.

For evaluation in the LIBERO environment, we follow the protocols of Kim et al. [2024], Zhao et al. [2025], Black et al. [2025] and present qualitative comparisons between the official $\pi_{0.5}$ checkpoint fine-tuned on LIBERO[1] and our proposed DAG-VLA with AFIL built upon the $\pi_{0.5}$ backbone. We include representative tasks from the LIBERO Spatial, Object, Goal, and Long test suites.

More complete environment setup visualizations are presented in Figure 4.

3 Justification of the Cosine-Disagreement Guidance Scale

Adaptive Negative Guidance via Score Disagreement. At diffusion timestep η, let $s_{\text{succ}}(\boldsymbol{a}) = \nabla \log p_s(\boldsymbol{a}_\eta)$ and $s_{\text{fail}}(\boldsymbol{a}_\eta) = \nabla \log p_f(\boldsymbol{a}_\eta)$ denote the score functions of the success and failure models, respectively. We consider a guided score of the form (*c.f.* Eq. (7) in the main paper):

$$s_\lambda(\boldsymbol{a}_\eta) = s_{\text{succ}}(\boldsymbol{a}_\eta) - \lambda_\eta(\boldsymbol{a}_\eta)\, s_{\text{fail}}(\boldsymbol{a}_\eta), \tag{9}$$

where $\lambda_\eta(\boldsymbol{a}_\eta)$ is an adaptive guidance scale.

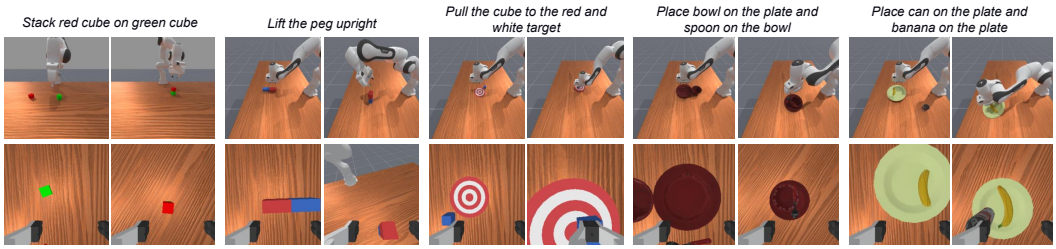
Optimal Guidance as Projection. Assume there exists an unknown optimal score function $s^*(\boldsymbol{a}_\eta)$ that induces task-optimal behavior. We seek λ_η such that s_λ best approximates s^* in a least-squares sense:

$$\lambda^*_\eta(\boldsymbol{a}_\eta) = \arg\min_\lambda \| s_{\text{succ}}(\boldsymbol{a}_\eta) - \lambda\, s_{\text{fail}}(\boldsymbol{a}_\eta) - s^*(\boldsymbol{a}_\eta) \|_2^2 . \tag{10}$$

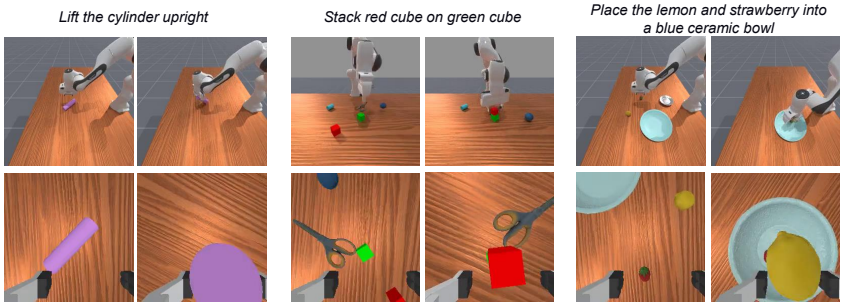
Taking derivative with respect to λ and setting it to zero yields:

$$\lambda^*_\eta(\boldsymbol{a}_\eta) = \frac{s_{\text{fail}}(\boldsymbol{a}_\eta)^\top \big(s_{\text{succ}}(\boldsymbol{a}_\eta) - s^*(\boldsymbol{a}_\eta)\big)}{\| s_{\text{fail}}(\boldsymbol{a}_\eta) \|_2^2}. \tag{11}$$

[1] Official checkpoint released at: https://github.com/Physical-Intelligence/openpi/blob/main/examples/libero/README.md



(a) In-domain Environment Setup.



(b) Out-of-domain Environment setup.

Figure 4: Row 1: exterior camera image, row 2: wrist camera image. For each task, we select one frame at the beginning (left) and end (right) of the trajectories. Corresponding to Figure 3 in the main paper

Residual Error Approximation. We model the optimal score as a first-order correction of the success score:

$$s^*(\boldsymbol{a}_\eta) = s_{\text{succ}}(\boldsymbol{a}_\eta) + \delta(\boldsymbol{a}_\eta), \tag{12}$$

where $\delta(\boldsymbol{a}_\eta)$ denotes the residual error of the success model. Substituting gives:

$$\lambda_\eta^*(\boldsymbol{a}_\eta) = -\frac{s_{\text{fail}}(\boldsymbol{a}_\eta)^\top \delta(\boldsymbol{a}_\eta)}{\|s_{\text{fail}}(\boldsymbol{a}_\eta)\|_2^2}. \tag{13}$$

Since $\delta(\boldsymbol{a}_\eta)$ is not directly observable, we approximate it using the discrepancy between success and failure scores:

$$\delta(\boldsymbol{a}_\eta) \approx s_{\text{succ}}(\boldsymbol{a}_\eta) - s_{\text{fail}}(\boldsymbol{a}_\eta). \tag{14}$$

Substituting this approximation:

$$\lambda_\eta^*(\boldsymbol{a}_\eta) \approx -\frac{s_{\text{fail}}^\top (s_{\text{succ}} - s_{\text{fail}})}{\|s_{\text{fail}}\|^2} \tag{15}$$

$$= 1 - \frac{s_{\text{fail}}^\top s_{\text{succ}}}{\|s_{\text{fail}}\|^2}. \tag{16}$$

Cosine Similarity Approximation. Assuming comparable magnitudes $\|s_{\text{succ}}\| \approx \|s_{\text{fail}}\|$, we obtain:

$$\lambda_\eta^*(\boldsymbol{a}_\eta) \approx 1 - \cos\big(s_{\text{succ}}(\boldsymbol{a}_\eta),\, s_{\text{fail}}(\boldsymbol{a}_\eta)\big). \tag{17}$$

Interpretation. This result shows that the optimal guidance scale is proportional to the angular disagreement between the success and failure scores. When the two scores are aligned, the residual correction is small and negative guidance is suppressed. When they disagree, the failure score provides complementary information, and stronger guidance is applied. This leads to a simple and fully local approximation:

$$\lambda_\eta(\boldsymbol{a}_\eta) \propto 1 - \cos\big(s_{\text{succ}}(\boldsymbol{a}_\eta),\, s_{\text{fail}}(\boldsymbol{a}_\eta)\big), \tag{18}$$

which avoids explicit estimation of distribution distances while retaining a principled connection to optimal score approximation.

4 Additional Experimental Results

Additional Experiments on LIBERO-PRO We performed additional experiments on LIBERO-PRO Liu et al. [2023], Zhou et al. [2025] and report the success rates of the base $\pi_{0.5}$, using the results reported in the original publication, and our proposed DAG-VLA$_{\pi_{0.5}}$ with AFIL on LIBERO-10 and LIBERO-Spatial in Table S1. The results show that our proposed DAG-VLA$_{\pi_{0.5}}$ consistently outperforms the baseline $\pi_{0.5}$ pipeline across five perturbation settings introduced in LIBERO-PRO benchmark – object attributes (Obj.), initial positions (Pos.), task variations (Task), semantic variations (Sem.), and environmental changes (Env.) – demonstrating improved generalization through adaptive negative guidance.

Table S1: Success rates (%) under object (Obj.), position (Pos.), semantic (Sem.), task, and environment (Env.) perturbations on LIBERO-10 and LIBERO-Spatial. Bold values indicate improvements over the corresponding $\pi_{0.5}$ baseline.

Method	Benchmark	Obj.	Pos.	Sem.	Task	Env.
$\pi_{0.5}$	LIBERO-10	92.0	0.08	93.0	0.01	46.0
DAG-VLA$_{\pi_{0.5}}$ with AFIL	LIBERO-10	**93.4**	**8.2**	**93.6**	**7.0**	**51.0**
$\pi_{0.5}$	LIBERO-Spatial	97.0	20.0	97.0	0.01	46.0
DAG-VLA$_{\pi_{0.5}}$ with AFIL	LIBERO-Spatial	**97.5**	**46.8**	**97.4**	**0.2**	46.0

Ablation Study of Guidance Scale for Static Failure-Informed Learning In this section, we analyze the effect of the guidance scale λ on the performance of the proposed DAG-VLA with failure-informed learning under the static guidance setting, corresponding to "DAG-VLA+Static-FIL" in Table 1 of the main paper. We evaluate on two tasks: *Lift the peg upright* (short-horizon) and *Place bowl on the plate and spoon on the bowl* (long-horizon), in the in-domain setting. The results shown in Table S2 show that $\lambda = 0.05$ empirically achieves the best performance across all tasks. In contrast, setting λ either too small or too large degrades manipulation performance, indicating the importance of properly balancing attraction toward successful behaviors and repulsion from failure-prone actions.

Table S2: Ablation study on guidance scale λ for static failure-informed learning. Success rates reported in %.

Task	$\lambda = 0.01$	$\lambda = 0.05$	$\lambda = 0.1$	$\lambda = 0.2$
Lift the peg upright	84.7	**86.7**	82.7	78.0
Place bowl on the plate and spoon on the bowl	46.8	**49.3**	47.1	44.0

Error Bars While diffusion-based policies introduce inherent stochasticity during inference, as they generate actions through an iterative denoising process initialized from random noise. We report standard deviation for proposed DAG-VLA with AFIL in in-domain setting (*c.f.* Table 1 in the main paper) in Table S3 and Table S4.

Table S3: Standard deviation (in percentage points) of 3 test-runs (per task) on DAG-VLA with AFIL in Maniskill Gu et al. [2023] in-domain evaluation settings. Each test-run with 50 configs $\times$ 3 runs = 150 rollouts.

Task	**Std Dev (s)**
Stack red cube on green cube	<0.01
Lift the peg upright	2.03
Pull cube to the red and white target	1.39
Place bowl on the plate and spoon on the bowl	1.41
Place can on the plate and banana on the plate	4.44

Failure Data Efficiency To further analyze the data efficiency of the failure generator, We therefore conduct a failure-data scaling study using 0% (corresponding to the "DAG-VLA, no AFIL" baseline in Table 1), 50%, 75%, 90%, and 100% (corresponding to "DAG-VLA with Adaptive-FIL" in Table 1) of the available raw failure trajectories, while keeping the original training data, model initialization,

Table S4: Standard deviation (in percentage points) of 3 test-runs (per task) on DAG-VLA with AFIL in Maniskill Gu et al. [2023] out-of-domain evaluation settings. Each test-run with 25 configs $\times$ 3 runs = 75 rollouts.

Task	**Std Dev** (s)
Lift Cylinder (unseen object/color)	1.26
Fruit Sorting (unseen objects + clutter)	<0.01
Stack Cubes (cluttered scene)	3.33

and evaluation protocol fixed. We evaluate one representative short-horizon task, *Stack red cube on green cube*, and one long-horizon task, *Place can on the plate and banana on the plate*, and report the success rates in Table S5. The results show consistent improvement as more failure trajectories are introduced. Notably, using only 50% of the available failures already improves the success rate by 4.0% on the short-horizon task and 12.7% on the long-horizon task. Increasing the amount of failure data further yields additional but diminishing gains: with 90% of the data, performance reaches 84.0% and 70.7%, close to the full-data results of 84.7% and 71.3%, respectively. These results indicate that useful failure-informed guidance can be learned from substantially fewer than 1,500 failure trajectories. At the same time, additional failure data continue to improve performance, suggesting that broader coverage of failure-prone behaviors remains beneficial.

Table S5: Task success rates (%) as the number of failure trajectories used for AFIL training increases. The percentage in each column indicates the fraction of the 1,500 available failure trajectories used.

Task	0 failures (0%)	750 failures (50%)	1,125 failures (75%)	1,350 failures (90%)	1,500 failures (100%)
Stack red cube on green cube	76.7	80.7	82.0	84.0	**84.7**
Place can on the plate and banana on the plate	56.0	68.7	69.3	70.7	**71.3**

Justification of Dual Action Generator Design A single action generator conditioned on trajectory quality, as in $\pi_{0.7}$, is a reasonable alternative for learning from mixed-quality data. To evaluate this design, we trained a $\pi_{0.5}$-based single action generator (Single-AG) using outcome-conditioned language prompts: successful trajectories were paired with "success: [task description]," while failed trajectories were paired with "failure: [task description]." However, this formulation consistently underperformed the proposed dual-generator (Dual-AG) design, with both approaches evaluated under the same training and evaluation protocols: We observed similar degradation with alternative

Table S6: Task success rates (%) on ManiSkill comparing single- and dual-action-generator VLA models with AFIL. The dual-generator model achieves higher success rates on all five tasks.

Task	Single-AG VLA with AFIL	Dual-AG VLA with AFIL
Stack red cube on green cube	72.0	**84.7**
Lift the peg upright	75.3	**90.7**
Pull cube to the red and white target	82.5	**98.0**
Place bowl on the plate and spoon on the bowl	45.3	**56.7**
Place can on the plate and banana on the plate	54.7	**71.3**

prompt formulations, including explicit statements such as "this is a successful/failed trajectory" placed before or after the task description. Therefore, we use separate success and failure generators with disjoint supervision. This design prevents raw failure actions from directly altering the success generator and its parameter statistics, better leverages the visual-language alignment learned during pretraining, and provides explicit estimates of the success and failure action distributions for guidance.

Success Only Baseline Ablation We perform an additional ablation comparing DAG-VLA trained with 750 successful and 750 failure-corrected trajectories against a success-only model trained with 1,500 successful trajectories, corresponding to the same setting as Table 1 in the main paper. The results are reported in Table S7.

Table S7: Task success rates (%) for DAG-VLA trained on 1,500 success-only trajectories or on 750 success and 750 failure-correction trajectories.

Task	DAG-VLA (1,500 success-only)	DAG-VLA (750 success / 750 failure-correction)
Stack red cube on green cube	71.3	74.0
Pull cube to the red and white target	60.0	82.5
Place can on the plate and banana on the plate	39.3	46.0

5 Parameter Analysis and Runtime

Parameter Analysis For the proposed DAG-VLA built upon $\pi_{0.5}$, we introduce only an additional action generator, implemented as a Gemma-300M expert contributing approximately 428M parameters, together with a dedicated output projection head containing only 32K parameters. Since the original $\pi_{0.5}$ model contains approximately 3.35B parameters, DAG-VLA$_{\pi_{0.5}}$ introduces a moderate increase in total parameter count, by 12.76%, while achieving consistent performance improvements across diverse manipulation tasks, environments, and domains.

Runtime Analysis We further analyze the inference-time overhead of the proposed adaptive failure-informed guidance. To ensure a fair comparison, we evaluate both the baseline $\pi_{0.5}$ VLA and DAG-VLA with AFIL over 10 episode rollouts under the same environment and hardware setting. The average runtime per action horizon step is 2.88 ms for baseline $\pi_{0.5}$, and 3.89 ms for DAG-VLA with AFIL, on a NVIDIA A100-SXM4-80GB GPU. The two generators can also be inferred/evaluated in parallel, reducing the runtime to 3.45 ms per action-horizon step, with GPU memory usage increasing only modestly from 14,426.61 MiB to 14,675.53 MiB. The moderate runtime increase is expected, as AFIL queries both the success and failure action generators with per-diffusion-step distance measuring during sampling, but remains practical for deployment while providing consistent robustness gains.

6 Significance Analysis

we conducted a matched A/B test on one short-horizon task, *Lift the Peg Upright*, and one long-horizon tas, *Place Can on the Plate and Banana on the Plate*, to isolate the effect of adaptive failure guidance. The resuls are shown in Table S8. For each task, the treatment and control conditions use the same trained DAG-VLA model, training data, evaluation configurations, and rollout protocol as reported in Table 1 of the main paper. The control condition disables failure-informed guidance and samples actions using only the success generator, corresponding to "DAG-VLA, no FIL" in Table 1. In contrast, the treatment condition applies Adaptive-FIL during action sampling, corresponding to "DAG-VLA + Adaptive-FIL" in Table 1. Both conditions are evaluated on the same 50 configurations, with three paired episodes per configuration, yielding 150 paired trials.

Table S8: Paired comparison of task success rates with and without Adaptive-FIL. Improvements and 95% paired confidence intervals (CI) are reported in percentage points (pp); p-values are from exact McNemar tests.

Task	No AFIL	Adaptive-FIL	Improvement	95% Paired CI	Exact McNemar Test (p)
Lift the Peg Upright	80.00%	**90.67%**	**+10.67 pp**	**[+3.97, +17.37] pp**	**0.0037**
Place Can on the Plate and Banana on the Plate	56.00%	**71.33%**	**+15.33 pp**	**[+5.26, +25.41] pp**	**0.0052**

For *Lift the Peg Upright*, Adaptive-FIL succeeds in 136 of 150 trials, compared with 120 of 150 trials without AFIL. Among the discordant pairs, AFIL succeeds in 22 cases in which the control fails, while the control succeeds in only six cases in which AFIL fails. The exact McNemar test yields $p = 0.0037$, with a paired odds ratio of 3.67. For *Place Can on the Plate and Banana on the Plate*, Adaptive-FIL succeeds in 107 of 150 trials, compared with 84 of 150 trials without AFIL. AFIL uniquely succeeds in 43 discordant pairs, compared with 20 control-only successes, yielding an exact McNemar test result of $p = 0.0052$ and a paired odds ratio of 2.15.

Because repeated episodes within the same configuration may be correlated, we additionally performed configuration-level analyses. For the two tasks, cluster-bootstrap confidence intervals for the

success-rate improvements are $[+3.33, +18.00]$ and $[+4.00, +26.67]$ percentage points, respectively. Configuration-level permutation tests also confirm significant improvements, with $p = 0.0119$ and $p = 0.0165$, respectively. After applying the Holm correction across the two primary McNemar tests, both improvements remain statistically significant. These results provide evidence on representative short- and long-horizon tasks that the gains from Adaptive-FIL are unlikely to arise from evaluation variability alone.

7 Qualitative Failure Analysis

To better characterize the failures collected by our pipeline, we conducted a post hoc taxonomy analysis of 150 randomly sampled failed episodes, comprising 424 attempt-level video segments. Qwen3-VL-32B-Instruct was used to generate preliminary annotations, which were then manually reviewed and corrected. Each segment was assigned a primary failure category, an optional secondary category, and a brief visual rationale. Importantly, these annotations are used solely for analysis; AFIL training does not require failure-category labels or taxonomy supervision.

To avoid overcounting repeated attempts within an episode, we define the episode-level root cause as the earliest meaningful failure event. The resulting distribution is shown in Table S9. The collected rollouts therefore contain multiple semantically distinct failure modes spanning perception and target selection, reaching and grasping, object transport, release timing, and final placement stability. Representative examples include grasping the wrong object, closing the gripper beside the target, losing the object during transport, releasing it before reaching the target, and producing an initially valid but unstable stack. These failure categories were not specified during data collection or model training; they were identified only through a post hoc analysis of naturally occurring policy rollouts.

Table S9: Distribution of root failure modes across 424 failed episodes. Grasp failure accounts for the largest share.

Root Failure Mode	Episodes	Percentage
Grasp failed	267	63.0%
No action attempted	96	22.6%
Wrong-object interaction	22	5.2%
Released too early	16	3.8%
Misaligned reach	12	2.8%
Misaligned placement	8	1.9%
Unstable stack	2	0.5%
Dropped during transport	1	0.2%

Although grasp failures constitute the majority of failures for this task, the collected data also capture less frequent but behaviorally distinct errors, including target confusion, transport failure, premature release, placement misalignment, and instability after placement. This long-tailed distribution illustrates why a fixed, manually designed perturbation set may not fully represent the errors encountered by the learned policy. A qualitative of the failure modes is shown in Figure 5.

8 Limitations and Societal Impact

While AFIL provides an effective mechanism for leveraging failure trajectories in VLA learning, it has several limitations. First, the proposed failure-informed guidance is specifically designed for diffusion- and flow-based generative VLA policies, where action generation is performed through iterative sampling. Extending the same learning-from-failure principle to non-generative or directly autoregressive VLA architectures, such as OpenVLA Kim et al. [2024]-style policies, would require additional adaptation, since these models do not naturally expose intermediate denoising or flow steps for guidance injection. Second, although AFIL is model-agnostic within the class of diffusion- and flow-based VLA models, our current experiments are built on the $\pi_{0.5}$ architecture Black et al. [2025]. We leave the evaluation of AFIL on a broader range of generative VLA backbones to future work.

Societal Impact. AFIL aims to improve the robustness and reliability of VLA-based robotic manipulation by enabling policies to learn from failures and avoid failure-prone actions during

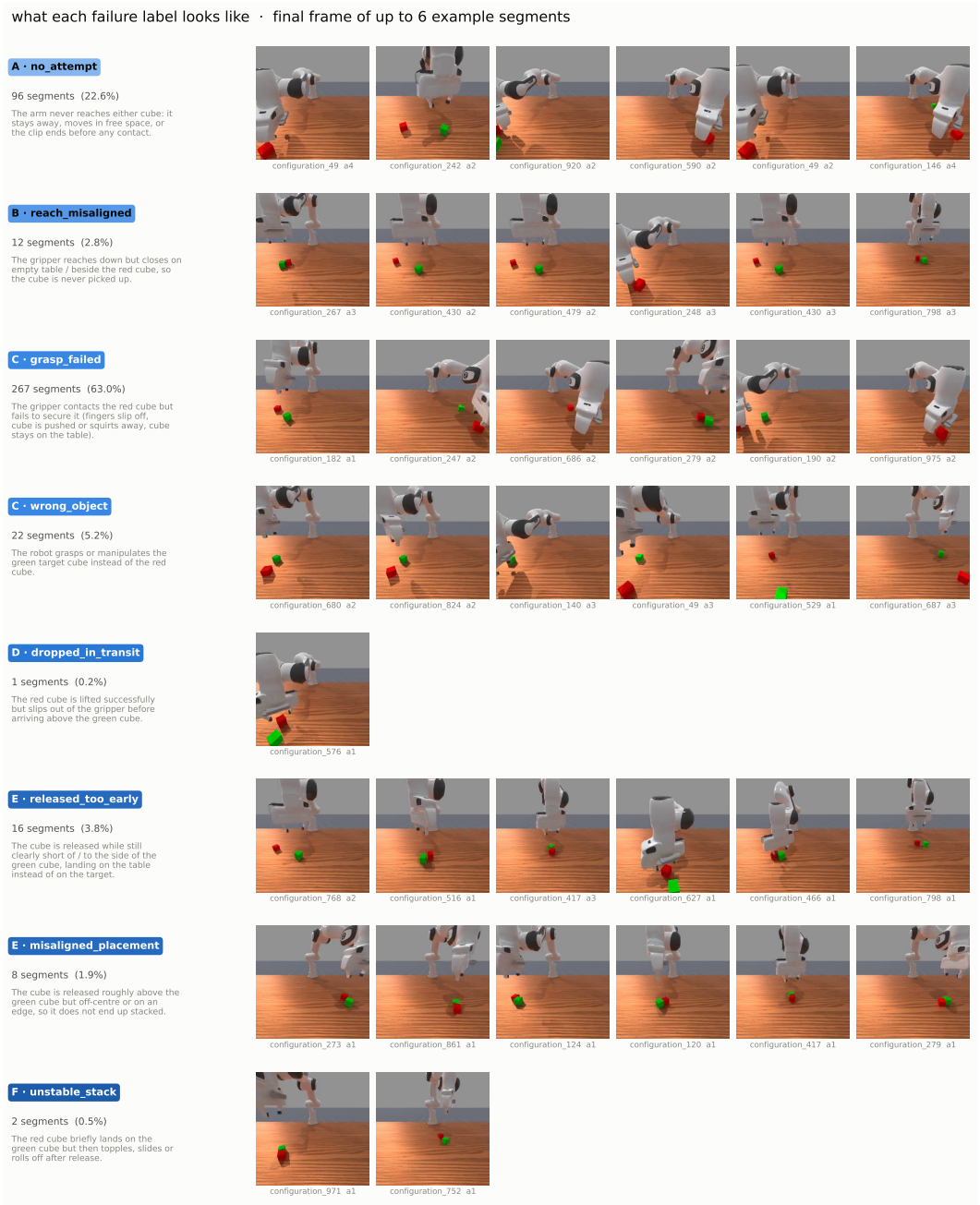


Figure 5: Qualitative analysis of the collected failure rollouts for training our proposed DAG-VLA with adaptive negative guidance (AFIL).

deployment. This may benefit applications such as assistive robotics, household manipulation, warehouse automation, and human-robot collaboration, where more reliable manipulation can reduce task disruption and potential physical damage. However, more robust robotic autonomy also raises safety and accountability concerns. AFIL does not guarantee safe behavior under all distribution shifts, hardware conditions, or human interactions. Real-world deployment should therefore include appropriate safety constraints, monitoring, physical safeguards, and human oversight, particularly in safety-critical environments. Failure data should also be collected and curated responsibly to avoid reinforcing unsafe or domain-specific behaviors.

NeurIPS Paper Checklist

1. **Claims**

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

   Answer: [Yes]

   Justification: Our claims are consistent with the experiments in Section 4 of the main paper.

   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: Limitations have been discussed in supplementary Section 8.

   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: The paper does not contain theoretical results that require proofs.

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: Section 4 of the main paper and the supplementary have sufficient details for the reproducibility of the experimental 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: [No]

Justification: Code, training/evaluation data and the trained DAG-VLA model with AFIL are currently pending release approval and will be publicly released upon acceptance of the paper.

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: Implementation details are discussed in Section 1 in supplementary material, ablation studies of hyperparameters are presented in Section 4 in the main paper and Section 4 in supplementary material.

   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: [Yes]

   Justification: Error bars of LIBERO environment evaluations are reported in Table 3 in the main paper, following prior work Kim et al. [2024], Octo Model Team et al. [2024], Zhao et al. [2025]. Statistical significance of the experiments is analyzed in Supplementary Section 4.

   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: Details are presented in Section 1 in the supplementary.

   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 conducted in this paper conforms to the NeurIPS code of ethics to the best of our knowledge.

   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: Societal impact discussed in Supplementary Section 8.

    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: To our knowledge, this paper does not pose such risks.

    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 open-source simulation environments are properly cited in the paper. License information is included in the References.

    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: [Yes]

    Justification: The open-source simulation environments are properly cited in the paper. License information is included in the References. Data generation and processing details are discussed in Supplementary Section 1 and Section 4 in the main 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 nor 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: The paper does not involve crowdsourcing nor research with human subjects.

    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: LLM is used only for writing, editing, or formatting purposes for this paper.

    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.