STOCHASTIC SELF-DISTILLATION FOR FLOW-BASED VISION-LANGUAGE-ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Flow-based vision–language–action (VLA) models achieve strong performance in robotics through imitation learning, yet fine-tuning them with on-policy reinforcement learning (RL) is limited. In group-based RL methods like Group Relative Policy Optimization (GRPO), every rollout step requires multi-step ODE integration, and the sampler's limited stochasticity leads to rollout groups with low variance. We propose VLA-SSD, which addresses both high inference latency and limited stochasticity. In particular, our method self-distills a pretrained flow-matching action head into a stochastic one-step policy by conditioning a single-jump generator on a Brownian path. We derive a conditional action density via our stochastic one-step policy, enabling efficient post-training via likelihood-ratio RL algorithms. We show that distilling VLA-SSD from the π_0 reduces the action head's forward-pass time by 7.7 times, with a 5.5% relative decrease in LIBERO success rate. Leveraging the increased sample diversity of VLA-SSD during group generation, we show that post-training with GRPO from a few-shot $\pi_{0.5}$ checkpoint raises LIBERO success from 77.1% to 93.0%, achieving a 1.6% relative improvement over four-step π_{RL} GRPO.

1 INTRODUCTION

Vision–language–action (VLA) models have achieved significant advances by leveraging knowledge from pre-trained vision–language models in robotics (Brohan et al., 2023; Kim et al., 2024). However, state-of-the-art VLAs have yet to achieve comparable generalization to other foundation models due to their reliance on imitation learning from expert demonstrations for adaptation to embodiments and environments (Black et al., 2024; Physical Intelligence et al., 2025; Kim et al., 2025; NVIDIA et al., 2025; Shukor et al., 2025). Frontier flow-based VLAs generate continuous action chunks with a flow-matching action head (Black et al., 2024; Physical Intelligence et al., 2025). Yet, enabling them to improve via on-policy reinforcement learning (RL) is still limited by the inference time of multi-step ODE integration and a lack of stochasticity.

To improve efficiency, one-step flow-based policies have been introduced for continuous control (Zhan et al., 2026; Ziakas et al., 2026), replacing multi-step ODE integration with a single transport step learned via self-distillation (Song et al., 2023; Boffi et al., 2024; Geng et al., 2025; Boffi et al., 2025). However, in RL methods using multi-sample advantage estimation, such as Group Relative Policy Optimization (GRPO), the baseline is derived directly from a group of trajectories given the current state. Because flow-matching rely entirely on the initial random noise draw for stochasticity, they restrict the action diversity within the sampled group, resulting in a degraded training signal (Shao et al., 2024). To enable exploration, existing approaches increase sample diversity by introducing an external noise network or reverting to multi-step samplers that inject noise at each integration step (Chen et al., 2025; Zhang et al., 2025; Li et al., 2026).

In this work, we address both the efficiency and stochasticity limitations of current VLAs by learning a stochastic one-step policy via self-distillation. We propose VLA-SSD, a method that utilizes Itô flow maps (Pan et al., 2026) to distill a multi-step flow-matching VLA head into a one-step stochastic policy. In contrast to prior work, we introduce stochasticity directly through self-distillation rather than relying on an external noise network or a multi-step sampler. To enable likelihood-ratio policy optimization such as GRPO, we derive a conditional action density that requires only a single additional Jacobian-vector product to approximate. In our experiments, we first distill VLA-SSD

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

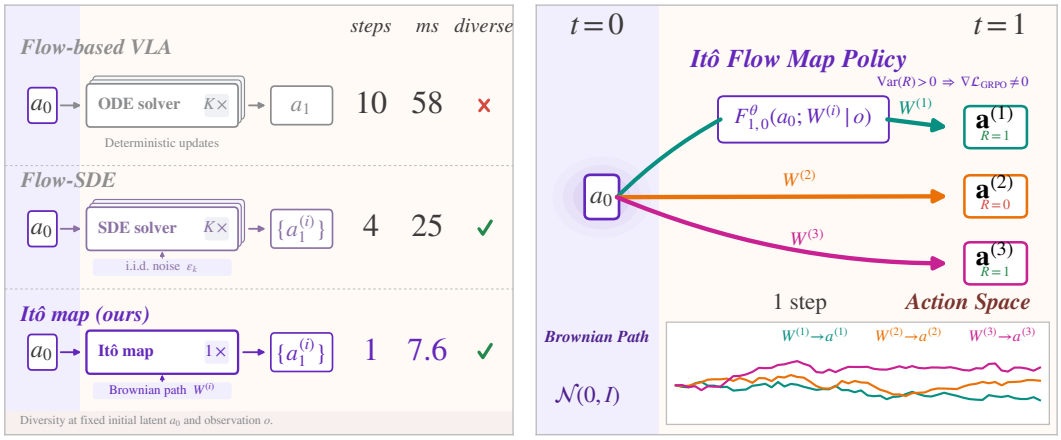

Figure 1: **Left:** an ODE sampler and Flow-SDE spend K action-expert evaluations per chunk; VLA-SSD spends one and keeps the spread of the SDE. **Right:** for a fixed s and a_0, the Itô flow map conditions on a Brownian path $W^{(k)}$, so G paths give G different one-step actions with different returns, and the GRPO group carries gradient; at $\sigma{=}0$ all G coincide.

from $\pi_{0.5}$, achieving a $7.7\times$ faster action head with a 0.3-point drop in success rate, while reducing zero-variance groups from 63% to 35% relative to the original sampler. Leveraging the stochasticity and efficiency of VLA-SSD, we post-train it with GRPO, reaching 93.0% on LIBERO, a 15.9-point gain over the SFT checkpoint, and matching four-step GRPO with π_{RL} (91.5%) while generating each action chunk $3.2\times$ faster during training rollouts.

In summary, our main contributions are as follows:

- **One-Step Stochastic VLA Policies:** We introduce VLA-SSD, a novel self-distillation framework that converts multi-step flow-matching VLA heads into efficient, one-step stochastic policies using Itô flow maps.
- **Density Evaluation for Likelihood-Ratio Optimization:** To overcome the lack of tractable likelihoods in flow-based models, we derive a closed-form conditional action density that evaluates efficiently via a single Jacobian-vector product.
- **Efficient Policy Distillation:** We demonstrate that distilling VLA-SSD from $\pi_{0.5}$ yields a $7.7\times$ faster action head with a minimal 0.3-point performance drop, while mitigating the exploration bottleneck by reducing zero-variance groups from 63% to 35%.
- **RL Post-Training:** When post-trained with GRPO, VLA-SSD reaches a 93.0% success rate on LIBERO, a 15.9-point gain over the SFT checkpoint, matching the standard four-step π_{RL} baseline while accelerating training rollouts by $3.2\times$.

2 Preliminaries

Markov decision process. We consider a Markov decision process (MDP) (Sutton & Barto, 1998) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, P, \gamma)$ with state space $\mathcal{S}$, action space $\mathcal{A} \subseteq \mathbb{R}^d$, reward function $r(s, a)$, transition kernel $P(s' \mid s, a)$, and discount factor $\gamma \in [0, 1]$. The state $s \in \mathcal{S}$ consists of the visual observations, the proprioceptive state, and the language instruction; the action $a \in \mathcal{A}$ is a chunk of H consecutive actions. A policy $\pi(a \mid s)$ is evaluated by its expected return $\mathcal{J}(\pi) = \mathbb{E}_\pi\big[\sum_{\tau=0}^{T-1} \gamma^\tau r(s^\tau, a^\tau)\big]$, where τ indexes decision steps and T is the episode horizon. We consider episodic manipulation tasks with a sparse binary reward, $r(s^\tau, a^\tau) = 1$ at the decision step at which the task is first completed and 0 otherwise, and take $\gamma = 1$, so that the return of an episode is $R = \sum_\tau r(s^\tau, a^\tau) \in \{0, 1\}$. Throughout, τ denotes MDP time; the flow time $t \in [0, 1]$ introduced next indexes the generation of a single action chunk and is distinct from τ.

Flow-matching policies. A generative policy $a_1 \sim \pi_1(\cdot \mid s)$ can be formulated as a transport from a reference measure $p_0 \in \mathcal{P}(\mathbb{R}^d)$ to a target action distribution $p_1 \in \mathcal{P}(\mathbb{R}^d)$. The transport is defined

by a state-conditioned flow $\psi_t(\cdot \mid s)$ that solves the following ODE:

$$\frac{d}{dt}\psi_t(a_t \mid s) = v_t\left(\psi_t(a_t \mid s)\right), \quad \psi_0(a_0) = a_0, \tag{1}$$

where v_t is a time-dependent velocity field. The terminal map induces the policy $\pi_1(\cdot \mid s) = [\psi_1(\cdot \mid s)]_{\#}p_0$; it is deterministic for fixed a_0 and s, while sampling $a_0 \sim p_0$ induces a distribution over actions. To train the flow policy, we define a conditional probability path $p_t(\cdot \mid z, s)$ using the endpoints $z = (a_0, a_1)$ and the linear interpolation $a_t = (1-t)a_0 + ta_1$. The velocity field is learned by regressing against the target conditional velocity $v_t^*(a_t \mid z, s)$ using the conditional flow-matching objective (Lipman et al., 2023; Albergo et al., 2023):

$$\mathcal{L}_{\text{CFM}} = \mathbb{E}_{t,q(z),p_t(a_t|z,s)}\|v_t(a_t \mid s) - v_t^*(a_t \mid z, s)\|_2^2 = \mathbb{E}_{t,q(z),p_t(a_t|z,s)}\|v_t(a_t \mid s) - (a_1 - a_0)\|_2^2,$$

where $q(z)$ is a coupling over the states—e.g., independent coupling $q(z) = p_0(a_0)p_1(a_1)$—and in the last equality we substitute $v^*(a_t \mid z, s) = a_1 - a_0$ with its analytic linear speed target velocity.

Flow map policies. Instead of solving the ODE, we parameterize and learn its two-time solution operator, referred to as the flow map (Song et al., 2023; Frans et al., 2025; Geng et al., 2025; Boffi et al., 2024; Albergo et al., 2023; Ziakas et al., 2026). For $0 \le r \le t \le 1$, let $X_{r,t}(\cdot \mid s) : \mathbb{R}^d \to \mathbb{R}^d$ denote the flow map associated with the ODE in eq. (1). It transports an action from time r to t and satisfies the jump condition

$$X_{r,t}(a_r \mid s) = a_t. \tag{2}$$

The one-step policy draws $a_0 \sim p_0$ and returns $a_1 = X_{0,1}(a_0 \mid s)$, inducing the action distribution

$$\pi(\cdot \mid s) = [X_{0,1}(\cdot \mid s)]_{\#}p_0. \tag{3}$$

Following Boffi et al. (2024), we parameterize the flow map using the average action velocity $u_{r,t}$:

$$X_{r,t}(a_r \mid s) = a_r + (t-r)u_{r,t}(a_r \mid s), \qquad u_{r,t}(a_r \mid s) = \frac{1}{t-r}\int_r^t v_\tau(a_\tau \mid s)\,\mathrm{d}\tau. \tag{4}$$

Thus, $X_{r,t}$ represents a finite jump of size $t - r$ along the ODE trajectory. On the diagonal, the average velocity recovers the instantaneous velocity,

$$u_{t,t}(a_t \mid s) = v_t(a_t \mid s). \tag{5}$$

This diagonal relation connects the finite-time flow map to the instantaneous flow-matching velocity. We use it to construct the training objective for the stochastic map in section 3.1.

Group relative policy optimization. GRPO (Shao et al., 2024) is an on-policy reinforcement learning algorithm that replaces the learned value baseline of actor–critic methods with a Monte Carlo baseline computed within a group of rollouts. At each decision step τ the policy samples an action chunk $a^\tau \sim \pi_\theta(\cdot \mid s^\tau)$, and the objective of section 2 reduces to $\mathcal{J}(\theta) = \mathbb{E}_{\pi_\theta}[R]$ with the binary return $R \in \{0, 1\}$. GRPO samples a group of N trajectories from the same task and initial state, and assigns to every decision step of trajectory i an advantage $\hat{A}_i$ obtained by normalizing its return within the group:

$$\hat{A}_i = \frac{R_i - \bar{R}}{\text{std}(R) + \kappa}, \qquad \bar{R} = \frac{1}{N}\sum_{j=1}^{N} R_j. \tag{6}$$

If all N returns in a group coincide, $\hat{A}_i = 0$ for all members; we term such groups *degenerate*, as they consume rollout computation but contribute no gradient. Using these advantages, GRPO updates the policy by minimizing a PPO-style (Schulman et al., 2017) clipped surrogate:

$$\mathcal{L}_{\text{clip}}(\theta) = -\mathbb{E}\left[\min\left(\rho^\tau(\theta)\hat{A},\ \text{clip}\big(\rho^\tau(\theta), 1-\epsilon, 1+\epsilon\big)\hat{A}\right)\right], \quad \rho^\tau(\theta) = \frac{\pi_\theta(a^\tau \mid s^\tau)}{\pi_{\theta_{\text{old}}}(a^\tau \mid s^\tau)}. \tag{7}$$

3 METHOD

3.1 STOCHASTIC FLOW MAP POLICIES

Introducing stochasticity via Itô maps. While standard flow maps enable one-step generation, their stochasticity is limited to the initial random noise a_0. To this end, we extend the flow map in eq. (4) to an Itô map, which takes an intermediate action and a realization of a standard Brownian motion W over $[0, 1]$ as inputs and predicts a future action on the same stochastic path (Pan et al., 2026). Evaluated from $r = 0$ to $t = 1$, it defines a stochastic one-step policy. In practice, we condition the network on a finite-dimensional representation of W.

Following the Itô-map construction of Pan et al. (2026), its martingale component M_t captures the Brownian noise accumulated from time 0 to t, weighted by the diffusion profile $\beta(t)$ and scaled by the noise level σ:

$$M_t := \sigma \int_0^t \beta(\zeta)\, \mathrm{d}W_\zeta, \qquad \Delta M_{r,t} := M_t - M_r, \tag{8}$$

where $\beta(t)$ is the diffusion profile and $\sigma \geq 0$ is the noise scale. The Itô map then combines a path-dependent average drift with the martingale increment $\Delta M_{r,t}$:

$$F^\theta_{r,t}(a; W \mid s) = a + (t - r)\, G^\theta_{r,t}(a; W \mid s) + \Delta M_{r,t}, \tag{9}$$

where $G^\theta_{r,t}$ approximates the path-conditioned average finite-variation term in the Itô-map decomposition. At $\sigma = 0$, the martingale M_t vanishes and the stochastic construction reduces to the deterministic flow map in eq. (4). For each action coordinate, we use the truncated Karhunen–Loève expansion $W_t = \sum_{k=1}^{L} \phi_k(t)\xi_k$, where $\xi \sim \mathcal{N}(0, I_L)$. The increment is then linear in the coefficients,

$$\Delta M_{r,t} = \sigma\, c(r, t)^\mathsf{T} \xi, \qquad c_k(r, t) = \int_r^t \beta(\zeta)\, \dot{\phi}_k(\zeta)\, \mathrm{d}\zeta, \tag{10}$$

so $\Delta M_{r,w} + \Delta M_{w,t} = \Delta M_{r,t}$ holds pathwise for $r \leq w \leq t$, and the same ξ conditions G^θ on every subinterval. For $m = \Delta M_{0,1}$, the stochastic policy is

$$a_1 = a_0 + G^\theta_{0,1}(a_0; W \mid s) + m. \tag{11}$$

Self-distillation. Starting from the pretrained velocity field, we train the Itô map without an external teacher (Boffi et al., 2025; Ziakas et al., 2026). On the diagonal $r = t$, the drift $G^\theta_{t,t}$ is independent of the Brownian path, and we evaluate it using the zero path $G^\theta_{t,t}(a; 0 \mid s)$. Compared with standard flow matching, its regression target includes the score correction of the marginal-preserving SDE. Under $\beta(t) = \sqrt{2(1 - t)}$, this target becomes $a_1 - (1 + \sigma^2)a_0$; the derivation is given in section B. Following Boffi et al. (2024), we consider Lagrangian (LSD), Eulerian (ESD), and progressive (PSD) self-distillation. These enforce consistency at the transported endpoint, at the starting point, and between direct and composed jumps, respectively (section A).

In this work,, we use Lagrangian self-distillation (LSD) (Boffi et al., 2024; Pan et al., 2026). At fixed a_r and Brownian coefficients ξ, let

$$y = F^\theta_{r,t}(a_r; W \mid s) \tag{12}$$

denote the transported action. LSD matches the endpoint derivative of the finite-variation component to the diagonal drift evaluated at y. The diagonal loss is

$$\mathcal{L}_{\text{Diag}} = \mathbb{E}\left[\left\| G^\theta_{t,t}(a_t; 0 \mid s) - \big(a_1 - (1 + \sigma^2)a_0\big)\right\|_2^2\right]. \tag{13}$$

The LSD loss is

$$\mathcal{L}_{\text{LSD}} = \mathbb{E}_{r<t,\, a_r,\, \xi}\left[\left\| G^\theta_{r,t}(a_r; W \mid s) + (t - r)\partial_t G^\theta_{r,t}(a_r; W \mid s) - \mathtt{sg}\big(G^\theta_{t,t}(y; 0 \mid s)\big)\right\|_2^2\right]. \tag{14}$$

Here, $\mathtt{sg}$ denotes a stop-gradient. The derivative is taken at fixed a_r and ξ on the finite-variation component

$$F^\theta_{r,t}(a_r; W \mid s) - \Delta M_{r,t} = a_r + (t - r) G^\theta_{r,t}(a_r; W \mid s),$$

so the martingale increment itself is not differentiated.

At $\sigma = 0$, the score correction and martingale component vanish, and eq. (14) reduces to deterministic flow-map self-distillation. We use LSD because the Eulerian objective does not identify the Itô map (Pan et al., 2026, Prop. 2), while the progressive objective requires two sequential forward passes. The full objective is $\mathcal{L}_{\text{Diag}} + \lambda \mathcal{L}_{\text{LSD}}$; implementation details are provided in section D.

3.2 On-Policy Optimization of Stochastic Flow Maps

Intractable action likelihood. To optimize our policy via likelihood-ratio RL, we need a tractable closed-form evaluation for the conditional action density $\pi_\theta(a \mid s)$. Since continuous flow-based models lack this, prior work converts the flow model into a discrete-time Markov process and computes likelihood ratios over K intermediate Gaussian denoising steps rather than the final action (Chen et al., 2025; Zhang et al., 2025). Our approach bypasses this limitation by deriving an auxiliary-variable conditional density. To do so, we isolate the Gaussian component of the Brownian path that generates the martingale increment m in eq. (11), enabling us to treat the one-step action as an invertible transformation of a single Gaussian variable, conditioned on the remaining path randomness.

To isolate the martingale increment m from the rest of the Brownian path, we project the path's random coefficients into two independent components. Let $c = c(0,1) \neq 0$ be the full-interval coefficient from eq. (10), and define its unit direction $e = c/\|c\|_2$. For each action coordinate $j = 1, \ldots, d$, the increment is $m_j = \sigma c^\mathsf{T} \xi_j$. We decompose the underlying standard Gaussian ξ_j into a component parallel to e (which perfectly determines m_j) and an orthogonal remainder $\xi_{\perp,j}$:

$$\xi_j \;=\; e\,\frac{m_j}{\sigma\|c\|_2} \;+\xi_{\perp,j}, \qquad \xi_{\perp,j} \;=\; (I_L - ee^\mathsf{T})\xi_j. \tag{15}$$

Let $m = (m_1, \ldots, m_d)$ and $\xi_\perp = (\xi_{\perp,1}, \ldots, \xi_{\perp,d})$. We group all the randomness that we want to condition on into a single auxiliary variable $h := (a_0, \xi_\perp)$. Because orthogonal projections of a standard Gaussian are independent, the additive noise follows $m \sim \mathcal{N}(0, \Sigma I_d)$ with variance $\Sigma := \sigma^2\|c\|_2^2$, completely independent of $\xi_\perp$. Since the initial latent a_0 is also drawn independently, m is independent of the entire auxiliary variable h.

This decomposition is highly useful because it isolates m as the only free variable. Conditioned on a specific state and auxiliary variable (s, h), the entire Brownian path is deterministically reconstructed from m. Let $g_\theta(m; s, h)$ denote the full-jump drift evaluated on this reconstructed path. Given a stored rollout (s, a, h), we can now ask: what is the density of a under updated parameters θ? The following proposition shows that, by holding h fixed, the action becomes an invertible function of a single Gaussian variable m, allowing us to compute its exact density via a standard change of variables.

Proposition 1 (Conditional action density). *Let $h = (a_0, \xi_\perp)$ and $\Sigma = \sigma^2\|c\|_2^2$. Define*

$$T_\theta(m; s, h) = a_0 + g_\theta(m; s, h) + m, \qquad J_\theta(m; s, h) \;=\; \frac{\partial g_\theta(m; s, h)}{\partial m}. \tag{16}$$

Assume $\sigma > 0$ and $\sup_m \|J_\theta(m; s, h)\|_2 < 1$. Then T_θ is invertible, and

$$\log \pi_\theta(a \mid s, h) \;=\; \log \mathcal{N}(m_\theta^\star; 0, \Sigma I_d) - \log\left|\det(\, I_d + J_\theta(m_\theta^\star; s, h)\,)\right|, \tag{17}$$

where $m_\theta^\star = T_\theta^{-1}(a; s, h)$.

Proof sketch. The action is generated by $a = a_0 + g_\theta(m; s, h) + m$. The randomness comes from the initial latent a_0 and the Brownian coefficients ξ. As shown above, splitting ξ isolates the additive noise $m \sim \mathcal{N}(0, \Sigma I_d)$, which is entirely independent of the auxiliary variable $h = (a_0, \xi_\perp)$. section C. □

To evaluate eq. (17) in practice, we approximate the updated preimage with one fixed-point iteration initialized at the stored rollout increment, and retain the first-order term $\log\det(I_d + J_\theta) \approx \operatorname{tr}(J_\theta)$. We estimate this trace with a single Hutchinson probe, requiring one additional Jacobian-vector

product, and reuse the same probe for the old and updated policies to reduce variance in their score difference. The resulting score approximates the conditional log-density; the inverse update and scoring formula are provided in section C and eq. (30).

Likelihood ratios with auxiliary variables. Proposition 1 gives the action density conditioned on the auxiliary variable $h = (a_0, \xi_\perp)$, rather than the marginal density $\pi_\theta(a \mid s)$. Let $\nu(h)$ denote the distribution of h, which is independent of θ. The marginal policy is

$$\pi_\theta(a \mid s) = \int \nu(h)\pi_\theta(a \mid s, h)\,\mathrm{d}h. \tag{18}$$

We equivalently define the augmented policy

$$\bar{\pi}_\theta(h, a \mid s) := \nu(h)\pi_\theta(a \mid s, h). \tag{19}$$

The augmented policy samples the pair (h, a), while the environment receives only a. Marginalizing out h therefore recovers eq. (18), so $\bar{\pi}_\theta$ and π_θ induce the same trajectory distribution and expected return $\mathcal{J}$.

Because $\nu(h)$ is independent of θ,

$$\nabla_\theta \log \bar{\pi}_\theta(h, a \mid s) = \nabla_\theta \log \pi_\theta(a \mid s, h), \frac{\bar{\pi}_\theta(h, a \mid s)}{\bar{\pi}_{\theta_{\text{old}}}(h, a \mid s)} = \frac{\pi_\theta(a \mid s, h)}{\pi_{\theta_{\text{old}}}(a \mid s, h)}. \tag{20}$$

Consequently, before the approximations in section C, the policy-gradient estimator constructed from the conditional density is unbiased for $\nabla_\theta \mathcal{J}$. The augmented and marginal formulations optimize the same expected-return objective, although their likelihood-ratio variances and clipped surrogate objectives can differ.

The auxiliary variable sampled during rollout is stored and reused when rescoring the recorded action; the Brownian path is never resampled. Using the approximate score from eq. (30), the likelihood ratio is

$$\widetilde{\rho}_i^\tau(\theta) = \exp\left[\widetilde{\ell}_\theta(a_i^\tau \mid s_i^\tau, h_i^\tau) - \widetilde{\ell}_{\theta_{\text{old}}}(a_i^\tau \mid s_i^\tau, h_i^\tau)\right], \tag{21}$$

where i indexes trajectories and τ indexes decision steps.

GRPO. For each update, we collect groups of N trajectories from the same task and initial environment state. Group members use independent initial latents a_0 and Brownian coefficients ξ, and the auxiliary variables sampled at every decision step are stored for policy rescoring. Episode returns are converted into the group-normalized advantages in eq. (6). The Brownian path increases action diversity beyond that provided by a_0 alone. Without this additional stochasticity, the policy more frequently produces zero-variance groups, for which all advantages vanish; section 4.3 measures this effect. We minimize the clipped GRPO surrogate in eq. (7) using the approximate likelihood ratio $\widetilde{\rho}_i^\tau(\theta)$ from eq. (21). The stored auxiliary variable h_i^τ is reused when evaluating both the old and updated policies. Training details are provided in section D.

4 Experiments

The experiments evaluate the end-to-end performance of VLA-SSD and analyze its underlying mechanics. We first evaluate the final on-policy reinforcement learning success rates and computational efficiency against multi-step baselines (section 4.2). We then analyze the source of the one-step policy's sample diversity and its impact on the GRPO signal (section 4.3). Finally, we ablate the initial self-distillation phase to isolate the impact of different consistency objectives on the base policy (section 4.4).

4.1 Experimental Setup

Model and checkpoints. We use $\pi_{0.5}$ (Physical Intelligence et al., 2025), a 3.3B-parameter VLA model with a 300M-parameter action expert serving as the flow-matching head described in section 3. The vision–language backbone is kept frozen throughout training. All runs start from the SFT checkpoints released with π_{RL} (Chen et al., 2025), so differences from its published numbers come from the post-SFT procedure alone. On LIBERO, reinforcement learning starts from the few-shot

Table 1: Main results. Success on the LIBERO four-suite mean and MetaWorld MT50 (ours at σ=0); passes is action-expert evaluations per chunk, forward its time per chunk on one A100. π_{RL} rows are its published numbers from the same SFT checkpoints. Bold is best per column.

Policy	Sampler	Passes	Forward (ms)	LIBERO	MetaWorld
$\pi_{0.5}$ SFT	ODE, K=10	10	58.4	77.1	51.8
π_{RL}, GRPO	SDE, K=4	4	24.5	91.5	—
π_{RL}, PPO	SDE, K=4	4	24.5	**98.3**	70.7
Itô student (ours)	Itô, K=1	1	7.6	68.6	49.0
Itô + GRPO (ours)	Itô, K=1	1	7.6	93.0	**72.0**

checkpoint (77.1%), as in π_{RL}; on MetaWorld the released checkpoint scores 51.8% under our protocol.

Benchmarks and protocol. LIBERO (Liu et al., 2023) is evaluated on its four ten-task suites (spatial, object, goal, long) with 500 episodes per suite; a four-suite mean therefore averages 2,000 episodes and has a standard error of 0.5–0.7 points. MetaWorld MT50 (Yu et al., 2020) is evaluated on 50 tasks $\times$ 10 fixed reset states at horizon 100, following π_{RL}. Unless stated otherwise, success rates are deterministic evaluations.

Baselines. We compare against π_{RL}'s GRPO with Flow-SDE and PPO with Flow-Noise at their operating depth K=4, using the results that work reports on the same checkpoints and protocol; against the deterministic flow map distilled with the same recipe at $\sigma = 0$; and against the released ODE sampler at K=4 and K=10.

Training. Distillation uses the Lagrangian objective, with the sampling interval gradually expanded over the first quarter of training, L=5 Karhunen–Loève modes, and σ=0.1. We use the profile $\beta(t) = \sqrt{2(1-t)}$, for which the diagonal correction in eq. (14) equals the constant σ^2. GRPO runs in RLinf (Yu et al., 2025) on four A100 GPUs with groups of N=8 rollouts sharing a task and initial state. Section D provides all settings.

4.2 Main Results: On-Policy RL Performance

On both benchmarks in table 1, the performance improvement from GRPO far exceeds the initial drop caused by one-step distillation. The final policy improves over the released SFT checkpoint by 15.9 percentage points on LIBERO and 20.2 points on MetaWorld. Under GRPO, our one-step policy achieves 93.0% on LIBERO, outperforming π_{RL}'s four-step GRPO policy (91.5%). While four-step PPO achieves a higher 98.3% on LIBERO, it requires the substantial overhead of training an additional vision–language critic. On MetaWorld, our one-step GRPO policy achieves 72.0%, outperforming the four-step PPO baseline (70.7%).

4.3 Analysis: Stochasticity and Sample Diversity

Source of sample diversity. Table 2 distinguishes variation induced by the initial latent from variation introduced during generation by fixing a_0 (fig. 2). Both the ODE sampler and the deterministic map are deterministic conditional on a_0, and their effective rank drops to zero when a_0 is fixed. With independently sampled latents, the ODE sampler has an effective rank of 3.2. The Itô map retains a similar rank with fixed and free latents (18.5 and 18.2), showing that substantial diversity can be generated through the Brownian path alone. Its fixed-latent rank exceeds that of four-step Flow-SDE (14.3) and matches that of isotropic Gaussian output noise. Their effects on return variation nevertheless differ, as examined next.

Zero-variance groups. The group-relative reward term in GRPO provides no learning signal when all members of a group obtain identical returns. Success rates are similar across samplers (83.1–84.6%), whereas the fraction of zero-variance groups varies. The ODE sampler and deterministic map produce zero return variance in 62.5% and 56.2% of groups, respectively. Isotropic output noise reduces this fraction to 47.9%, and four-step Flow-SDE to 45.8%. The Itô map yields the lowest observed fraction, 35.4%, despite having a similar effective rank to the isotropic control. This suggests that its sample diversity translates more effectively into within-group return variation. Under binary rewards,

a lower zero-variance fraction corresponds to a larger proportion of groups providing a group-relative learning signal

Retention under training noise. Exploration noise is useful only if the n oisy policy still succeeds often enough for groups to contain both outcomes. On MetaWorld the Itô student attains 36.8% under the training noise σ=0.1 against 49.0% deterministically, a retention of 75%; after GRPO the figures are 55.4% and 72.0%, a retention of 77%. For comparison, π_{RL} reports that Flow-SDE at K=1 retains 14.7% of its deterministic success and 86% at K=4, which is why that method is trained at four steps.

	Effective rank		GRPO groups	
Sampler	Free a_0	Fixed a_0	SR (%)	Zero-var.
$\pi_{0.5}$ ODE, K=10	3.2	0.0	84.4	0.625
Flow-SDE, K=4	13.7	14.3	84.6	0.458
Deterministic map, K=1	10.3	0.0	84.4	0.562
+ $\mathcal{N}(0, 0.1^2 I)$ noise	18.5	18.5	84.4	0.479
Itô map, K=1, σ=0.1	18.2	18.5	83.1	**0.354**

Table 2: Sample diversity and GRPO signal. Effective rank measures action diversity with free or fixed initial latent a_0; zero-variance is the fraction of rollout groups with identical returns. SR denotes success rate. Multi-step samplers use the SFT checkpoint; one-step samplers use its distilled student.

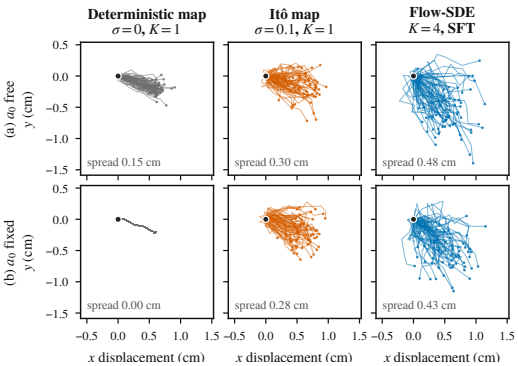

Figure 2: End-effector trajectories with (a) independent and (b) shared initial latents a_0. Fixing a_0 collapses the deterministic samples, while the stochastic samplers retain diversity. The distilled models shown use an earlier student checkpoint.

4.4 Ablation: Impact of Self-Distillation

Distillation loss. Distilled from the full-demonstration $\pi_{0.5}$ checkpoint, the one-step Itô student attains 96.55% on the LIBERO four-suite mean against the ten-step teacher's 96.85%, with the residual concentrated on the long-horizon suite. Evaluated at K=1 and K=5, the same checkpoint varies by less than $\pm$2 points, indicating that the student approximates the solution operator over the interval rather than a single endpoint. The loss is larger when the teacher is weaker: 8.5 points from the few-shot LIBERO checkpoint (77.1 $\rightarrow$ 68.6) and 2.8 points on MetaWorld (51.8 $\rightarrow$ 49.0), the two starting points of section 4.2. The order of training also matters: distilling from the converged SFT checkpoint rather than jointly from the $\pi_{0.5}$ base raised a matched-budget student from 12.2% to 42.7% at K=1.

Table 3: One-step distillation from π_0 on LIBERO, 500 episodes per suite, all students evaluated at K=1. The teacher is π_{RL}'s full-dataset SFT π_0 checkpoint at K=10; the deterministic map under ESD is MeanFlow (Geng et al., 2025). Forward is the action head's time per chunk on one A100. Bold marks the best objective for each map; the standard error on each mean is about 0.7 points.

Map	Objective	Forward (ms)	spatial	object	goal	long	mean
π_0 teacher, K=10	—	58.4	96.8	98.8	95.8	85.2	94.15
Deterministic flow map	LSD	7.6	96.2	**96.0**	**94.0**	**79.0**	**91.30**
	PSD	7.6	**96.4**	95.4	93.6	77.8	90.80
	ESD	7.6	90.2	76.8	77.0	26.0	67.50
Itô flow map, σ=0.1	LSD	7.6	94.2	**93.6**	**91.0**	**77.0**	**88.95**
	PSD	7.6	**96.6**	91.2	90.8	73.2	87.95
	ESD	7.6	85.4	76.8	62.6	37.2	65.50

Consistency objective and map. Table 3 compares deterministic and Itô maps under three consistency objectives using the π_0 teacher. The Lagrangian and progressive objectives yield one-step

performance reductions of 2.9–6.2 percentage points, with the largest reductions on the long-horizon suite. The Eulerian objective exhibits substantially larger reductions, consistent with reduced accuracy of first-order extrapolation from the diagonal as $|t - r|$ increases. LSD achieves the highest mean success for both maps and is used throughout. Under LSD, the Itô map achieves 88.95%, compared with 91.30% for the deterministic map, indicating a 2.35-percentage-point performance trade-off associated with the stochastic formulation. Its exploration properties are analyzed in section 4.3.

5 Related Work

5.1 Reinforcement fine-tuning of generative policies

Building on pretrained VLAs (Brohan et al., 2023; Kim et al., 2024; Black et al., 2024; Physical Intelligence et al., 2025), online RL has been explored for autoregressive action generation (Lu et al., 2025; Li et al., 2025). For diffusion and flow policies, DPPO optimizes the denoising chain with PPO (Ren et al., 2025), while ReinFlow and π_{RL} introduce stochastic denoising transitions for likelihood-based fine-tuning, retaining multi-step generation (Zhang et al., 2025; Chen et al., 2025). In image generation, Flow-GRPO and DanceGRPO apply GRPO through ODE-to-SDE conversion (Liu et al., 2025; Xue et al., 2025); Flow-Map GRPO extends this approach to few-step generators through anchored stochastic composition (Li et al., 2026). Complementary critic-based approaches optimize a fixed policy's latent-noise distribution (DSRL) (Wagenmaker et al., 2025), update flow maps through critic-guided trust regions (Flow Map Q-Guidance) (Ziakas et al., 2026), or combine value calibration with action filtering (FORCE) (Zhang et al., 2026). Our work enables GRPO fine-tuning of a one-step stochastic VLA through its conditional action density, avoiding a multi-step denoising chain.

5.2 One-step generation and stochastic flow maps

Diffusion and flow policies model multimodal robot actions through iterative sampling (Chi et al., 2023; Black et al., 2024). Consistency models, shortcut models, MeanFlow, and flow-map methods replace iterative integration with learned finite-time transport (Song et al., 2023; Frans et al., 2025; Geng et al., 2025; Boffi et al., 2024; Albergo et al., 2023), including self-distillation from the map's own diagonal (Boffi et al., 2025). In robotics, Consistency Policy and SnapFlow distill policies into few-step or one-step samplers (Prasad et al., 2024; Luan et al., 2026), while Let It Be Simple enables one-step decoding through high-noise training (Chen et al., 2026). These maps are deterministic given the initial noise. Stochastic extensions include Meta Flow Maps for reward-conditioned posterior sampling (Potaptchik et al., 2026) and Itô maps for Brownian-path-conditioned SDE transitions (Pan et al., 2026). We build on Itô maps to combine one-step VLA generation with explicit exploration and conditional-density-based on-policy optimization.

6 Conclusion

We presented VLA-SSD, which distills a flow-matching VLA into a one-step stochastic policy based on Itô flow maps and fine-tunes it with GRPO. The Brownian path provides exploration with an explicit noise scale, and the derived conditional density makes the policy usable in likelihood-based on-policy updates. On LIBERO and MetaWorld MT50, the result matches multi-step GRPO fine-tuning with π_{RL} using just one action-expert evaluation per chunk instead of four. Regarding limitations, our approximate likelihood scoring relies on a first-order truncation of the log-determinant. In addition, while GRPO avoids the heavy memory overhead of a vision–language critic, it does not match the peak absolute sample efficiency of critic-based PPO on LIBERO. However, because physical robot rollouts operate in real-time and constitute the primary bottleneck of on-policy RL, the $3.2\times$ faster rollout generation provided by VLA-SSD is particularly valuable for real-world deployment. Translating this framework to real-robot fine-tuning is the natural next step. Finally, investigating adaptive step sizing—where the model defaults to the one-step Itô map but allocates additional sampling steps for complex, high-precision maneuvers—presents an exciting direction for bridging one-step and multi-step continuous control.

AI USE STATEMENT

We used generative AI tools to create and edit software code, assist in the writing of the proof of Proposition 1, modify scientific figures, format LaTeX tables and references, and edit the research paper to improve readability. We have not used generative AI tools to generate synthetic datasets, propose or refine hypotheses, design or provide feedback on research methodology or experiments, or interpret results. The remaining required disclosure tasks (e.g., developing theoretical models, formulating mathematical claims, assisting with translation, cleaning and reformatting datasets, and supporting qualitative and thematic data analysis) are not applicable to this work. We have carefully reviewed all AI-assisted work. All AI-assisted text, code, and mathematical content were reviewed and verified by the authors. We take full responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses existing robotic simulation benchmarks and does not involve human subjects or personally identifiable information. While the proposed method is evaluated only in simulation, deploying learned robotic policies in the physical world may introduce safety risks. Real-world deployment should therefore include appropriate safety constraints, human oversight, and testing under controlled conditions.

REPRODUCIBILITY STATEMENT

We provide the complete training objectives, theoretical derivations, implementation details, optimization hyperparameters, and evaluation protocols in the main text and appendices. The appendix additionally describes the Brownian-path representation, policy-scoring approximation, distillation and reinforcement-learning configurations, and benchmark evaluation settings.

REFERENCES

Michael S. Albergo, Nicholas M. Boffi, and Eric Vanden-Eijnden. Stochastic Interpolants: A Unifying Framework for Flows and Diffusions. *arXiv preprint arXiv:2303.08797*, 2023. URL https://arxiv.org/abs/2303.08797. (Cited on pages 3 and 9)

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A Vision-Language-Action Flow Model for General Robot Control. *arXiv preprint arXiv:2410.24164*, 2024. URL https://arxiv.org/abs/2410.24164. (Cited on pages 1 and 9)

Nicholas M. Boffi, Michael S. Albergo, and Eric Vanden-Eijnden. Flow map matching with stochastic interpolants: A mathematical framework for consistency models. *arXiv preprint arXiv:2406.07507*, 2024. URL https://arxiv.org/abs/2406.07507. (Cited on pages 1, 3, 4, 9, and 13)

Nicholas M. Boffi, Michael S. Albergo, and Eric Vanden-Eijnden. How to build a consistency model: Learning flow maps via self-distillation. *arXiv preprint arXiv:2505.18825*, 2025. URL https://arxiv.org/abs/2505.18825. (Cited on pages 1, 4, 9, and 13)

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-2: Vision-Language-Action

Models Transfer Web Knowledge to Robotic Control. *arXiv preprint arXiv:2307.15818*, 2023. URL https://arxiv.org/abs/2307.15818. (Cited on pages 1 and 9)

Kang Chen, Zhihao Liu, Tonghe Zhang, Zhen Guo, Si Xu, Hao Lin, Hongzhi Zang, Xiang Li, Quanlu Zhang, Zhaofei Yu, Guoliang Fan, Tiejun Huang, Yu Wang, and Chao Yu. π_{RL}: Online RL Fine-tuning for Flow-based Vision-Language-Action Models. *arXiv preprint arXiv:2510.25889*, 2025. URL https://arxiv.org/abs/2510.25889. (Cited on pages 1, 5, 6, and 9)

Yitong Chen, Shiduo Zhang, Jingjing Gong, and Xipeng Qiu. Let It Be Simple: One-Step Action Generation for Vision-Language-Action Models. *arXiv preprint arXiv:2606.05737*, 2026. URL https://arxiv.org/abs/2606.05737. (Cited on page 9)

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin CM Burchfiel, and Shuran Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. In *Proceedings of Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.026. URL https://www.roboticsproceedings.org/rss19/p026.html. (Cited on page 9)

Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One Step Diffusion via Shortcut Models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=OlzB6LnXcS. (Cited on pages 3 and 9)

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J. Zico Kolter, and Kaiming He. Mean Flows for One-step Generative Modeling. *arXiv preprint arXiv:2505.13447*, 2025. URL https://arxiv.org/abs/2505.13447. (Cited on pages 1, 3, 8, 9, and 14)

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag 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. *arXiv preprint arXiv:2406.09246*, 2024. URL https://arxiv.org/abs/2406.09246. (Cited on pages 1 and 9)

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-Tuning Vision-Language-Action Models: Optimizing Speed and Success. *arXiv preprint arXiv:2502.19645*, 2025. URL https://arxiv.org/abs/2502.19645. (Cited on page 1)

Haozhan Li, Yuxin Zuo, Jiale Yu, Yuhao Zhang, Zhaohui Yang, Kaiyan Zhang, Xuekai Zhu, Yuchen Zhang, Tianxing Chen, Ganqu Cui, Dehui Wang, Dingxiang Luo, Yuchen Fan, Youbang Sun, Jia Zeng, Jiangmiao Pang, Shanghang Zhang, Yu Wang, Yao Mu, Bowen Zhou, and Ning Ding. SimpleVLA-RL: Scaling VLA Training via Reinforcement Learning. *arXiv preprint arXiv:2509.09674*, 2025. URL https://arxiv.org/abs/2509.09674. (Cited on page 9)

Zhiqi Li, Wen Zhang, and Bo Zhu. Flow-Map GRPO: Reinforcement Learning for Few-Step Flow-Map Generators via Anchored Stochastic Composition. *arXiv preprint arXiv:2607.00535*, 2026. URL https://arxiv.org/abs/2607.00535. (Cited on pages 1 and 9)

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for Generative Modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t. (Cited on page 3)

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking Knowledge Transfer for Lifelong Robot Learning. In *Advances in Neural Information Processing Systems 36, Datasets and Benchmarks Track*, 2023. URL https://arxiv.org/abs/2306.03310. (Cited on page 7)

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-GRPO: Training Flow Matching Models via Online RL. *arXiv preprint arXiv:2505.05470*, 2025. URL https://arxiv.org/abs/2505.05470. (Cited on page 9)

Guanxing Lu, Wenkai Guo, Chubin Zhang, Yuheng Zhou, Haonan Jiang, Zifeng Gao, Yansong Tang, and Ziwei Wang. VLA-RL: Towards Masterful and General Robotic Manipulation with Scalable Reinforcement Learning. *arXiv preprint arXiv:2505.18719*, 2025. URL https://arxiv.org/abs/2505.18719. (Cited on page 9)

Wuyang Luan, Junhui Li, Weiguang Zhao, Wenjian Zhang, Tieru Wu, and Rui Ma. SnapFlow: One-Step Action Generation for Flow-Matching VLAs via Progressive Self-Distillation. *arXiv preprint arXiv:2604.05656*, 2026. URL https://arxiv.org/abs/2604.05656. (Cited on page 9)

NVIDIA, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. GR00T N1: An Open Foundation Model for Generalist Humanoid Robots. *arXiv preprint arXiv:2503.14734*, 2025. URL https://arxiv.org/abs/2503.14734. (Cited on page 1)

Zhengkai Pan, Peter Potaptchik, Wenxi Yao, Michael S. Albergo, and Jakiw Pidstrigach. Itô maps for any-step SDEs. *arXiv preprint arXiv:2606.11156*, 2026. URL https://arxiv.org/abs/2606.11156. (Cited on pages 1, 4, 5, and 9)

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a Vision-Language-Action Model with Open-World Generalization. *arXiv preprint arXiv:2504.16054*, 2025. URL https://arxiv.org/abs/2504.16054. (Cited on pages 1, 6, and 9)

Peter Potaptchik, Adhi Saravanan, Abbas Mammadov, Alvaro Prat, Michael S. Albergo, and Yee Whye Teh. Meta Flow Maps enable scalable reward alignment. *arXiv preprint arXiv:2601.14430*, 2026. URL https://arxiv.org/abs/2601.14430. (Cited on page 9)

Aaditya Prasad, Kevin Lin, Jimmy Wu, Linqi Zhou, and Jeannette Bohg. Consistency Policy: Accelerated Visuomotor Policies via Consistency Distillation. In *Proceedings of Robotics: Science and Systems*, 2024. URL https://arxiv.org/abs/2405.07503. (Cited on page 9)

Allen Z. Ren, Justin Lidard, Lars L. Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion Policy Policy Optimization. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2409.00588. (Cited on page 9)

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. *arXiv preprint arXiv:1707.06347*, 2017. URL https://arxiv.org/abs/1707.06347. (Cited on page 3)

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. *arXiv preprint arXiv:2402.03300*, 2024. URL https://arxiv.org/abs/2402.03300. (Cited on pages 1 and 3)

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Remi Cadene. SmolVLA: A Vision-Language-Action Model for Affordable and Efficient Robotics. *arXiv preprint arXiv:2506.01844*, 2025. URL https://arxiv.org/abs/2506.01844. (Cited on page 1)

Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency Models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 32211–32252. PMLR, 2023. URL https://proceedings.mlr.press/v202/song23a.html. (Cited on pages 1, 3, and 9)

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998. (Cited on page 2)

Andrew Wagenmaker, Mitsuhiko Nakamoto, Yunchu Zhang, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering Your Diffusion Policy with Latent Space Reinforcement Learning. *arXiv preprint arXiv:2506.15799*, 2025. URL https://arxiv.org/abs/2506.15799. (Cited on page 9)

Zeyue Xue, Jie Wu, Yu Gao, Fangyuan Kong, Lingting Zhu, Mengzhao Chen, Zhiheng Liu, Wei Liu, Qiushan Guo, Weilin Huang, and Ping Luo. DanceGRPO: Unleashing GRPO on Visual Generation. *arXiv preprint arXiv:2505.07818*, 2025. URL https://arxiv.org/abs/2505.07818. (Cited on page 9)

Chao Yu, Yuanqing Wang, Zhen Guo, Hao Lin, Si Xu, Hongzhi Zang, Quanlu Zhang, Yongji Wu, Chunyang Zhu, Junhao Hu, Zixiao Huang, Mingjie Wei, Yuqing Xie, Ke Yang, Bo Dai, Zhexuan Xu, Jiakun Du, Xiangyuan Wang, Xu Fu, Letong Shi, Zhihao Liu, Kang Chen, Weilin Liu, Gang Liu, Boxun Li, Jianlei Yang, Zhi Yang, Guohao Dai, and Yu Wang. RLinf: Flexible and Efficient Large-scale Reinforcement Learning via Macro-to-Micro Flow Transformation. *arXiv preprint arXiv:2509.15965*, 2025. URL https://arxiv.org/abs/2509.15965. (Cited on page 7)

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-World: A Benchmark and Evaluation for Multi-Task and Meta Reinforcement Learning. In *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 1094–1100. PMLR, 2020. URL https://proceedings.mlr.press/v100/yu20a.html. (Cited on page 7)

Guojian Zhan, Letian Tao, Pengcheng Wang, Yixiao Wang, Yiheng Li, Yuxin Chen, Hongyang Li, Masayoshi Tomizuka, and Shengbo Eben Li. Mean Flow Policy with Instantaneous Velocity Constraint for One-step Action Generation. *arXiv preprint arXiv:2602.13810*, 2026. URL https://arxiv.org/abs/2602.13810. (Cited on page 1)

Shuyi Zhang, Yunfan Lou, Hongyang Cheng, Yichen Guo, Chuyao Fu, Yaoxu Lyu, Xiaojie Zhang, Haoran Li, Pengwei Wang, Zhongyuan Wang, and Shanghang Zhang. FORCE: Efficient VLA Reinforcement Fine-Tuning via Value-Calibrated Warm-up and Self-Distillation. *arXiv preprint arXiv:2606.26006*, 2026. URL https://arxiv.org/abs/2606.26006. (Cited on page 9)

Tonghe Zhang, Chao Yu, Sichang Su, and Yu Wang. ReinFlow: Fine-tuning Flow Matching Policy with Online Reinforcement Learning. *arXiv preprint arXiv:2505.22094*, 2025. URL https://arxiv.org/abs/2505.22094. (Cited on pages 1, 5, and 9)

Christos Ziakas, Alessandra Russo, and Avishek Joey Bose. Aligning Flow Map Policies with Optimal Q-Guidance. *arXiv preprint arXiv:2605.12416*, 2026. URL https://arxiv.org/abs/2605.12416. (Cited on pages 1, 3, 4, 9, 13, and 14)

A Consistency Objectives for Flow-Map Distillation

This appendix states the three self-distillation objectives of section 3.1 in full, in the notation of section 2, and records the properties that motivate the choice made in the main text. Throughout, the flow map is parameterized as $X^{\theta}_{r,t}(a_r \mid s) = a_r + (t-r)\, u^{\theta}_{r,t}(a_r \mid s)$ for $0 \le r \le t \le 1$, the diagonal $u^{\theta}_{t,t}$ is trained by $\mathcal{L}_{\text{Diag}}$ in eq. (13), and $\mathtt{sg}$ denotes a stop-gradient; conditioning on s and the superscript θ are suppressed where unambiguous.

Origin of the conditions.. The exact solution operator $X_{r,t}$ of the ODE $\frac{\mathrm{d}}{\mathrm{d}t} a_t = v_t(a_t)$ satisfies the jump condition $X_{r,t}(a_r) = a_t$ and the semigroup property $X_{w,t} \circ X_{r,w} = X_{r,t}$ for $r \le w \le t$. Differentiating the jump condition with respect to the endpoint t gives $\partial_t X_{r,t}(a_r) = v_t(X_{r,t}(a_r))$; differentiating with respect to the start time r and using $\frac{\mathrm{d}}{\mathrm{d}r} X_{r,t}(a_r) = 0$ gives $\partial_r X_{r,t}(a_r) + \nabla X_{r,t}(a_r)\, v_r(a_r) = 0$. Each of these identities, together with the semigroup property, yields a residual that vanishes for the exact map and can be minimized for the parameterized map, with the diagonal $u_{t,t} = v_t$ serving as the teacher (Boffi et al., 2024; 2025; Ziakas et al., 2026).

Lagrangian self-distillation (LSD). Writing $y = X_{r,t}(a_r)$ for the transported state,

$$\mathcal{L}_{\text{LSD}} = \mathbb{E}_{r \le t,\, a_r}\left[\left\| \partial_t X_{r,t}(a_r) - \mathtt{sg}\big(u_{t,t}(y)\big) \right\|_2^2\right], \qquad \partial_t X_{r,t} = u_{r,t} + (t-r)\, \partial_t u_{r,t}. \tag{22}$$

The condition matches the endpoint derivative of the map to the instantaneous velocity evaluated at the transported state, so the map is supervised along the trajectory it generates. The time derivative $\partial_t u_{r,t}$ is obtained by a Jacobian-vector product with respect to the scalar t, and the target requires one additional evaluation of the diagonal at y.

Eulerian self-distillation (ESD).

$$\mathcal{L}_{\mathrm{ESD}} = \mathbb{E}_{r\le t,\, a_r}\left[\left\|\partial_r X_{r,t}(a_r) + \mathrm{sg}\big(\nabla X_{r,t}(a_r)\, u_{r,r}(a_r)\big)\right\|_2^2\right], \qquad \partial_r X_{r,t} = -u_{r,t} + (t-r)\,\partial_r u_{r,t}. \tag{23}$$

The condition transports the map along the characteristics of the diagonal field at the start point, so both the residual and its target are evaluated at a_r rather than at the transported state. The spatial term $\nabla X_{r,t}(a_r)\, u_{r,r}(a_r)$ is a Jacobian-vector product in the action argument. For the average-velocity parameterization, removing the stop-gradient and expanding $\nabla X_{r,t}\, u_{r,r} = u_{r,r} + (t-r)\,\nabla u_{r,t}\, u_{r,r}$ shows that the ESD gradient coincides with that of MeanFlow (Geng et al., 2025) up to the choice of teacher for the instantaneous velocity; see Ziakas et al. (2026, App. B).

Progressive self-distillation (PSD).

$$\mathcal{L}_{\mathrm{PSD}} = \mathbb{E}_{r\le t,\, a_r,\, \gamma}\left[\left\|X_{w,t}\big(X_{r,w}(a_r)\big) - \mathrm{sg}\big(X_{r,t}(a_r)\big)\right\|_2^2\right], \qquad w = (1-\gamma)\, r + \gamma\, t, \quad \gamma \sim \mathcal{U}[0,1]. \tag{24}$$

The condition enforces the semigroup property directly: a jump from r to t must agree with the composition of two shorter jumps through an intermediate time w. It requires no derivative of the network but two sequential forward passes on the left-hand side, and it constrains the off-diagonal map only through other off-diagonal evaluations, so the diagonal enters indirectly.

B Diagonal Target of the Itô Map

The diagonal loss in eq. (13) regresses the zero-path diagonal drift onto the drift of the SDE that shares marginals with the flow-matching ODE. For the interpolant $a_t = (1-t)\, a_0 + t\, a_1$, the marginal path p_t of the ODE $\frac{\mathrm{d}}{\mathrm{d}t} a_t = v_t(a_t)$ is also the marginal path of the SDE

$$\mathrm{d}a_t = \Big[v_t(a_t) + \tfrac{1}{2}\,\sigma^2 \beta(t)^2\, \nabla \log p_t(a_t)\Big]\mathrm{d}t + \sigma\beta(t)\,\mathrm{d}W_t, \tag{25}$$

for any diffusion profile $\sigma\beta(t)$, since the added diffusion and the score correction cancel in the Fokker–Planck equation. Conditioned on the endpoint a_1, the interpolant is Gaussian with mean $t\, a_1$ and covariance $(1-t)^2 I_d$, so the conditional score is $\nabla \log p_t(a_t \mid a_1) = -(a_t - t\, a_1)/(1-t)^2 = -a_0/(1-t)$. Substituting the conditional velocity $a_1 - a_0$ and the conditional score gives the regression target

$$a_1 - a_0 - \frac{\sigma^2 \beta(t)^2}{2(1-t)}\, a_0 = a_1 - \Big(1 + \frac{\sigma^2 \beta(t)^2}{2(1-t)}\Big) a_0, \tag{26}$$

which is the target in eq. (13); as for conditional flow matching, regressing onto the conditional target recovers the marginal drift in expectation. The target is singular at $t = 1$ unless $\beta(1) = 0$. With the profile $\beta(t) = \sqrt{2(1-t)}$ used in our experiments, the coefficient reduces to the constant $1 + \sigma^2$, and the Brownian variance accumulated over $[0,1]$ is $\sigma^2 \int_0^1 \beta(t)^2\, \mathrm{d}t = \sigma^2$ in the continuum; section D gives the truncated value used in practice.

Density diagnostics. At the end of distillation the Brownian pathway's output projection is of order 3×10^{-4}; consequently the variance $2\|J_\theta\|_F^2$ of the single-probe trace estimator and the truncation error $\le \|J_\theta\|_F^2/2$ of the first-order log-determinant in eq. (30) are both below 10^{-6} nats per chunk. At the first minibatch of every run the ratio $\widetilde{\rho}_i^{\tau}$ equals one to floating-point precision and the mean Jacobian correction is indistinguishable from zero. These values are measured at the start of training; the pathway norm was not logged during reinforcement learning.

C Proof of the Conditional-Density Proposition

Proof of proposition 1. For each action coordinate, $e^{\mathsf{T}}\xi_j$ and $(I_L - ee^{\mathsf{T}})\xi_j$ are independent Gaussian projections. Since $m_j = \sigma\|c\|_2 e^{\mathsf{T}}\xi_j$, the coordinates of m are independent and $m \sim \mathcal{N}(0, \Sigma I_d)$,

where $\Sigma = \sigma^2\|c\|_2^2$. Because a_0 is sampled independently of the Brownian coefficients, m is independent of $h = (a_0, \xi_\perp)$.

Fix (s, h) and an action a. A preimage of a under T_θ satisfies

$$m = a - a_0 - g_\theta(m; s, h).$$

The right-hand side is a contraction under the assumption of the theorem, so the Banach fixed-point theorem guarantees a unique solution $m_\theta^\star$. Therefore, $T_\theta(\cdot; s, h)$ is invertible. Its Jacobian is

$$\frac{\partial T_\theta}{\partial m} = I_d + J_\theta(m; s, h),$$

which is nonsingular because $\|J_\theta(m; s, h)\|_2 < 1$. Applying the change-of-variables formula to $m \sim \mathcal{N}(0, \Sigma I_d)$ gives eq. (17). □

The proposition provides the exact density conditioned on the auxiliary variable h; the marginal action density $\pi_\theta(a \mid s)$ remains intractable. We next approximate the inverse and log-determinant required to evaluate the conditional density.

Approximating policy scoring. After a policy update, the stored increment $m^{(0)}$ need not generate the recorded action under the updated parameters. The updated preimage satisfies the fixed-point equation

$$m = a - a_0 - g_\theta(m; s, h). \tag{27}$$

Starting from the stored increment $m^{(0)}$, we approximate the updated preimage using one fixed-point iteration:

$$m_\theta^{(1)} = a - a_0 - g_\theta(m^{(0)}; s, h). \tag{28}$$

Under the contraction assumption of proposition 1, the Jacobian correction to the conditional log-density admits the convergent trace expansion

$$\mathcal{C}(J) = \sum_{k=1}^{\infty} \frac{(-1)^k}{k} \operatorname{tr}(J^k), \qquad J = J_\theta(m_\theta^\star; s, h). \tag{29}$$

We retain its first-order term and estimate the trace using a Hutchinson probe $\omega \sim \mathcal{N}(0, I_d)$:

$$\begin{aligned} \widetilde{\ell}_\theta(a \mid s, h) = & -\frac{\|m_\theta^{(1)}\|_2^2}{2\Sigma} - \frac{d}{2}\log(2\pi\Sigma) \\ & - \omega^\mathsf{T} J_\theta(m^{(0)}; s, h)\omega. \end{aligned} \tag{30}$$

The final term requires one Jacobian-vector product. At $\theta = \theta_{\text{old}}$, the fixed-point update recovers the stored increment exactly, $m_{\theta_{\text{old}}}^{(1)} = m^{(0)}$. We use the same probe ω for the old and updated policies to reduce variance in the likelihood ratio. Thus, eq. (30) is used as an approximate policy score rather than an exact likelihood.

D Implementation Details

Model. The action expert of $\pi_{0.5}$ is extended with two additive pathways, each a two-layer MLP of width 1024 with swish activation and a zero-initialised output layer, so that the extended model reproduces the pretrained velocity at initialisation. The interval pathway embeds $t-r$ with a sinusoidal positional encoding (periods 4×10^{-3} to 4); the path pathway projects the martingale increment $\Delta M_{r,t}$ through the action input projection, mean-pools it over the horizon, adds the interval encoding, and passes the sum through the MLP. Both outputs enter the expert through its adaptive RMS-norm conditioning, not as additional tokens. The vision–language backbone (SigLIP and Gemma-2B) is frozen; the action expert, its projections, and the two pathways are trained, all in bfloat16 under FSDP without parameter sharding. The expert predicts a chunk of 10 steps in a 32-dimensional padded action space; the first 5 steps of the first 7 dimensions are executed, and the conditional density is evaluated on these $d = 5 \times 7 = 35$ coordinates only.

Brownian path. Each of the d coordinates carries an independent standard Brownian motion B on $[0, 1]$, represented by its first $L = 5$ Karhunen–Loève modes, $B(q) = \sum_{k=1}^{L} \phi_k(q)\,\xi_k$ with $\phi_k(q) =$

$\sqrt{2}\,\sin(\omega_k q)/\omega_k$, $\omega_k = (k-\frac{1}{2})\pi$, and $\xi \sim \mathcal{N}(0, I_L)$. The diffusion profile $\beta(t) = \sqrt{2(1-t)}$ is implemented as the time change $M_t = \sigma B\big((1-t)^2\big)$, so that the increment of eq. (10) is the closed-form difference $\Delta M_{r,t} = \sigma\big[B((1-r)^2) - B((1-t)^2)\big]$ with $c_k(r,t) = \phi_k((1-r)^2) - \phi_k((1-t)^2)$; no quadrature is used. For the full jump, $\|c(0,1)\|_2^2 = \sum_{k=1}^{5} 2/\omega_k^2 = 0.9596$, so the realized variance in eq. (17) is $\Sigma = 0.9596\,\sigma^2$ rather than its continuum limit σ^2; the same truncation is used at training and at scoring, since a mismatch would bias every importance ratio.

Distillation. The student is initialised from the $\pi_{0.5}$ SFT checkpoint and trained for 1,000 optimizer steps on 4 A100 GPUs with global batch 128 (micro-batch 4), AdamW with $\beta = (0.9, 0.95)$, $\epsilon = 10^{-8}$, weight decay 10^{-10}, and gradient clipping at 1.0. The learning rate is warmed up linearly to 5×10^{-6} over the run. The objective is $\mathcal{L}_{\text{Diag}} + \lambda \mathcal{L}_{\text{LSD}}$ with $\lambda = 1$, where the consistency term is switched on after the first 5% of steps; the consistency residual is not normalised and is typically one to two orders of magnitude larger than the diagonal residual, so $\lambda = 1$ does not correspond to an equal weighting of the two terms. Interval endpoints are drawn as $r = \text{mid} - p\,\delta/2$, $t = \text{mid} + p\,\delta/2$, where mid and δ are the midpoint and length of the interval between two independent $\mathcal{U}[0,1)$ draws and p increases linearly from 0 at 5% of the run to 1 at 25%; a quarter of each batch is forced to the full interval $(r,t) = (0,1)$. The noise scale is ramped linearly from 0 to $\sigma = 0.1$ over the first 30% of the run and held fixed thereafter. The checkpoint at step 600 is used.

Reinforcement learning. GRPO runs in RLinf on 4 A100 GPUs. Each iteration collects 8 groups of $N = 8$ rollouts on LIBERO (32 parallel environments, two rollout epochs) and 32 groups on MetaWorld, with the same task and initial state within a group, $K = 1$, and $\sigma = 0.1$ applied in full from the first iteration. Episodes run for at most 500 environment steps on LIBERO and 100 on MetaWorld; the return is 1 at the step of first success and 0 otherwise. Groups whose mean return lies outside $[0.1, 0.9]$ are discarded, which removes degenerate groups before the update. Advantages use $\kappa = 10^{-6}$ and are normalised across the batch; the clipped objective uses $\epsilon = 0.2$ with a dual clip at 3.0, token-mean aggregation, no entropy bonus, and no KL penalty. Each iteration performs one pass over the collected chunks in minibatches of 256 with micro-batch 64, i.e. 25 parameter updates per iteration, using AdamW with learning rate 10^{-6}, $\beta = (0.9, 0.95)$, weight decay 0.01, and gradient clipping at 1.0, without warmup or decay. The Jacobian-vector products in eqs. (22) and (30) are computed by forward-mode automatic differentiation.

Evaluation. LIBERO is evaluated on the four ten-task suites with 500 episodes per suite in 500 parallel environments; MetaWorld MT50 on 50 tasks with 10 fixed reset states each, following π_{RL}. Forward times in table 1 are wall-clock times of the action head per chunk on one A100.

Successful Rollouts. We visualize successful rollouts of the one-step VLA-SSD policy (Itô map, K=1, σ=0 at evaluation) on the four LIBERO suites and on MetaWorld MT50. Each figure shows uniformly spaced frames from a single successful episode, from the initial state to the step at which success is registered.

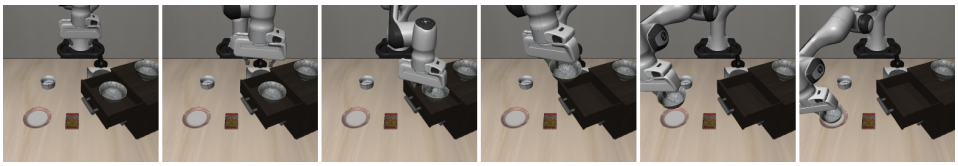

Figure 3: `LIBERO-Spatial`, task 4. Pick up the black bowl in the top drawer of the wooden cabinet and place it on the plate.

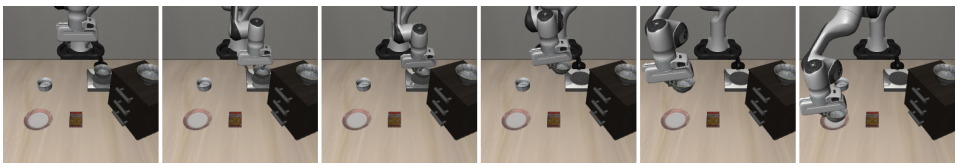

Figure 4: `LIBERO-Spatial`, task 7. Pick up the black bowl on the stove and place it on the plate.



Figure 5: `LIBERO-Object`, task 0. Pick up the alphabet soup and place it in the basket.

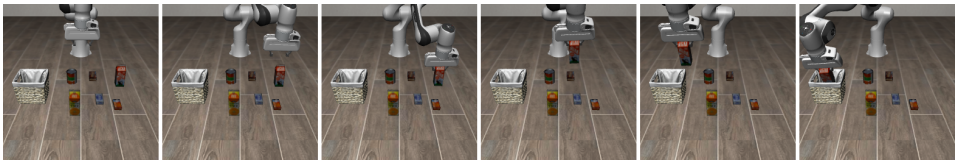

Figure 6: `LIBERO-Object`, task 7. Pick up the milk and place it in the basket.

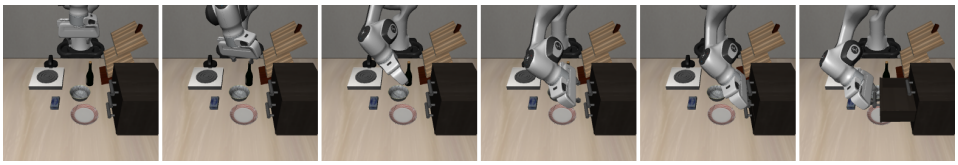

Figure 7: `LIBERO-Goal`, task 0. Open the middle drawer of the cabinet.

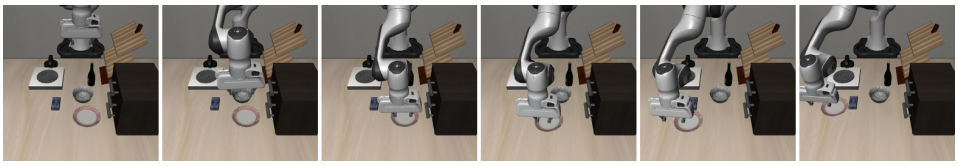

Figure 8: `LIBERO-Goal`, task 5. Push the plate to the front of the stove.



Figure 9: `LIBERO-Long`, task 2. Turn on the stove and put the moka pot on it.

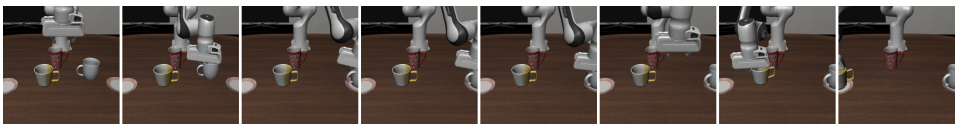

Figure 10: `LIBERO-Long`, task 4. Put the white mug on the left plate and put the yellow and white mug on the right plate.

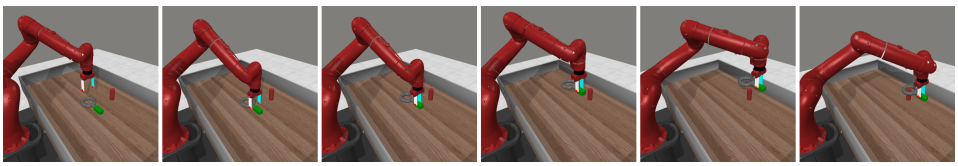

Figure 11: `assembly-v3` (MetaWorld MT50). Pick up a nut and place it onto a peg.

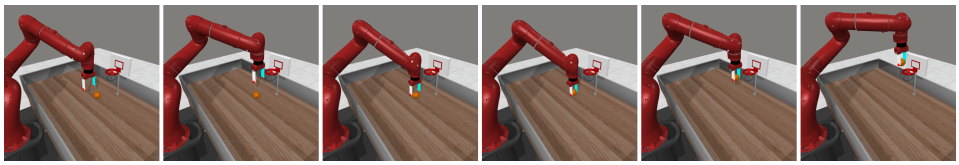

Figure 12: `basketball-v3` (MetaWorld MT50). Dunk the basketball into the basket.

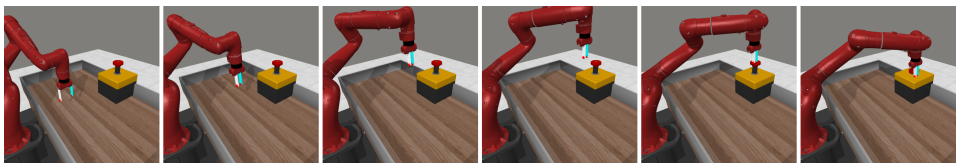

Figure 13: `button-press-topdown-v3` (MetaWorld MT50). Press a button from the top.

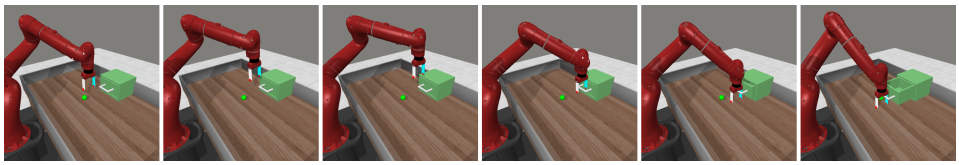

Figure 14: `drawer-open-v3` (MetaWorld MT50). Open a drawer.

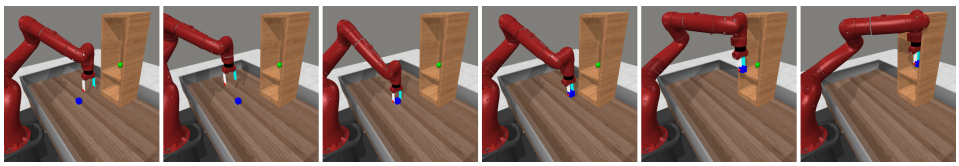

Figure 15: `shelf-place-v3` (MetaWorld MT50). Pick and place a puck onto a shelf.

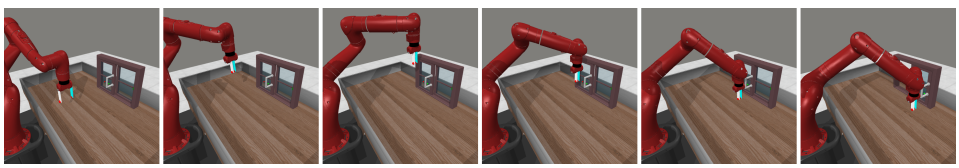

Figure 16: `window-open-v3` (MetaWorld MT50). Push and open a window.