One-Step Flow Policy: Self-Distillation for Fast Visuomotor Policies

Shaolong Li[1], Lichao Sun[2], and Yongchao Chen[3⋆]

[1] Computer Science and Engineering, University of Michigan, shaolonl@umich.edu
[2] Lehigh University, lis221@lehigh.edu
[3] College of AI, Tsinghua University, yongchaochen12@gmail.com

Abstract. Generative flow and diffusion models provide the continuous, multimodal action distributions needed for high-precision robotic policies. However, their reliance on iterative sampling introduces severe inference latency, degrading control frequency and harming performance in time-sensitive manipulation. To address this problem, we propose the One-Step Flow Policy (OFP), a from-scratch self-distillation framework for high-fidelity, single-step action generation without a pre-trained teacher. OFP unifies a self-consistency loss to enforce coherent transport across time intervals, and a self-guided regularization to sharpen predictions toward high-density expert modes. In addition, a warm-start mechanism leverages temporal action correlations to minimize the generative transport distance. Evaluations across 56 diverse simulated manipulation tasks demonstrate that a one-step OFP achieves state-of-the-art results, outperforming 100-step diffusion and flow policies while accelerating action generation by over 100×. We further integrate OFP into the $\pi_{0.5}$ model on RoboTwin 2.0, where one-step OFP surpasses the original 10-step policy. These results establish OFP as a practical, scalable solution for highly accurate and low-latency robot control.

Keywords: Flow-based generative policy · Fast visuomotor control

1 Introduction

Vision-Language-Action (VLA) models are progressing quickly in robot manipulation [7, 11, 14, 31], autonomous driving [54, 59], and long-horizon task execution [2, 27, 58]. Within this high-dimensional control setting, flow and diffusion models have emerged as a dominant paradigm for parameterizing conditional policies [5, 19, 20, 31]. These generative models offer two distinct advantages over discrete tokenization: they naturally represent the multimodal action distributions inherent in human demonstrations, and they output continuous control signals. This continuous output is essential for high-precision manipulation tasks where subtle action granularity determines success.

Despite these benefits, the current paradigm suffers from a critical inference latency bottleneck. Sampling from a flow or diffusion policy typically requires

⋆ Corresponding author.

iteratively solving an Ordinary Differential Equation (ODE) or a Stochastic Differential Equation (SDE), transporting samples from a noise prior to the target action distribution. This process requires tens to hundreds of forward passes through a large neural network for a single action [28, 44]. In time-sensitive robotics applications such as high-speed grasping or dynamic interaction, this latency is prohibitive. Action generation delays directly reduce control frequency and exacerbate compounding errors, frequently resulting in task failure. As VLA architectures scale in size and complexity, a central challenge arises:

How can we accelerate generative policies to output high-fidelity actions in a few steps, or even a single step, without compromising control precision?

A large body of work in image and video generation studies how to accelerate flow and diffusion sampling [15,23,45,51], aiming to produce high-fidelity outputs with a minimal Number of Function Evaluations (NFE). We focus on flow-based models due to their widespread industrial deployment [4,5,20]. Diffusion models can be viewed as a specific instance of the flow framework, allowing acceleration techniques to transfer across parameterizations. One common approach employs higher-order numerical solvers to reduce steps [33, 34]. However, in the extreme few-step regime, discretization errors become unmanageable and severely degrade generation quality.

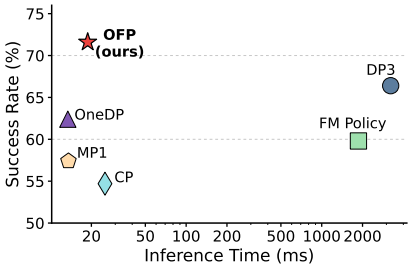

Fig. 1: Averaged across 56 tasks. Evaluated at NFE=1, OFP outperforms all other single-step baselines and accelerates generation by over 100× compared to DP3 and FM Policy (NFE=100).

Another line of work uses distillation [40], typically by training a student to compress the trajectory of a pre-trained teacher into fewer steps. Two primary families exist: Consistency Distillation (CD) [23,45] and Score Distillation (SD) [37,50,51]. CD enforces a self-consistency constraint where points along a trajectory map to the same endpoint, enabling one-step generation. SD instead treats the teacher score as a differentiable prior to optimize the generator. Recently, MeanFlow [15,16] attempted one-step generation by modeling an average velocity field, but it introduces Jacobian-vector products (JVPs) during training, which significantly increase memory costs and destabilizes optimization.

Prior works have directly adapted these vision acceleration ideas into robot policy learning to reduce inference latency, but their impact on control tasks has not been sufficiently analyzed. As a result, existing methods often struggle to achieve both low latency and high action precision. Consistency Policy (CP) [39] follows the consistency-distillation paradigm. However, due to the mode-covering nature of its trajectory-matching objectives [48], the policy tends to average over multimodal action distributions. While this aids few-step inference, the resulting one-step actions often lack the sharpness required for precise manipulation. In contrast, One-Step Diffusion Policy (OneDP) [47] employs a score-distillation

approach. These objectives are typically mode-seeking [36,46], causing the policy to collapse toward a single high-probability mode. While this produces high-quality one-step samples, it drastically reduces diversity. Due to their design, OneDP-like methods only support single-step inference, preventing them from trading computation time for higher control accuracy.

Motivated by these observations, we propose **One-Step Flow Policy (OFP)**, which combines the strengths of both directions without requiring an auxiliary network or a pre-trained teacher. OFP is a from-scratch self-distillation framework that unifies two complementary signals: self-consistency, which enforces cross-time coherence of the transport dynamics, and self-guidance, which injects a distribution-level correction that sharpens one-step predictions toward high-density expert behaviors. This combination enables high-fidelity one-step inference without performance degradation. Additionally, we introduce a warm-start action reuse mechanism that provides a strong prior for one-step generation, improving both temporal smoothness and control precision.

Our contributions are summarized as follows: 1) We present a unified self-distillation approach for flow-based policies that resolves the trade-off between inference speed and action precision without relying on teacher models. 2) We repurpose a warm-start action initialization strategy as a highly effective, training-free mechanism to reduce generative difficulty in few-step inference. 3) We achieve state-of-the-art success rates across 56 simulated tasks spanning Adroit [25], DexArt [3], and MetaWorld [52]. A 1-NFE OFP outperforms 100-NFE diffusion and flow baselines, delivering over a 100× acceleration. 4) We validate the scalability of OFP by integrating it into the $\pi_{0.5}$ model [20]. On RoboTwin 2.0 [8], our 1-step integration exceeds the baseline 10-step policy, proving its robustness for large-capacity systems and complex semantics.

2 Preliminaries

2.1 Flow Matching

Let the data space be $\mathbb{R}^d$. We aim to learn a time-dependent velocity field $\mathbf{v}_\theta \colon [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ defining an Ordinary Differential Equation (ODE). The flow $\{\boldsymbol{\xi}(t)\}_{t\in[0,1]}$ induced by this ODE transports a base distribution p_0 to a target data distribution p_1:

$$\frac{d}{dt}\boldsymbol{\xi}(t) = \mathbf{v}_\theta\big(\boldsymbol{\xi}(t), t\big), \quad \boldsymbol{\xi}(0) \sim p_0. \tag{1}$$

Consider a probability path $\{p_t\}_{t\in[0,1]}$ interpolating between the boundary conditions p_0 and p_1. If $\mathbf{v}_\theta$ matches the vector field generating this path, the solution $\boldsymbol{\xi}(1)$ will follow the target distribution p_1. Flow Matching trains $\mathbf{v}_\theta$ by regressing a target velocity field. However, the marginal velocity field is typically intractable.

To make training tractable, Conditional Flow Matching (CFM) [28] introduces a conditional probability path. Given a data sample $x_1 \sim p_1$ and noise

$\epsilon \sim p_0$, we define a conditional path $\mathbf{z}_t = a(t)x_1 + b(t)\epsilon,\ t \in [0,1]$, where $a(0) = 0,\ b(0) = 1$ and $a(1) = 1,\ b(1) = 0$. The time derivative is given by: $\dot{\mathbf{z}}_t(x_1, \epsilon) = a'(t)\, x_1 + b'(t)\, \epsilon$. The marginal velocity field corresponds to the expectation: $\bar{\mathbf{v}}(x,t) = \mathbb{E}[\dot{\mathbf{z}}_t(x_1,\epsilon) \mid \mathbf{z}_t = x]$. CFM learns $\mathbf{v}_\theta$ by directly regressing the derivative of the conditional path:

$$\mathcal{L}_{\mathrm{CFM}}(\theta) = \mathbb{E}_{t,x_1,\epsilon}\big\|\mathbf{v}_\theta(\mathbf{z}_t,t) - \dot{\mathbf{z}}_t(x_1,\epsilon)\big\|_2^2. \tag{2}$$

Minimizing Eq. (2) yields the same optimal solution as regressing the intractable marginal velocity field $\bar{\mathbf{v}}(x,t)$.

Sampling involves integrating Eq. (1) from $\mathbf{z}_0 \sim p_0$ using the learned field $\mathbf{v}_\theta$. In practice, this requires time discretization (e.g., Euler or RK4 solvers), resulting in multiple network forward passes. This iterative process is the main inference bottleneck. Since the marginal velocity field exhibits high curvature due to the averaging of overlapping trajectories, standard solvers incur large truncation errors in low-NFE regimes, leading to poor sample quality.

2.2 Flow-Based Generative Policy

We consider a visuomotor imitation learning setting. Let $\mathcal{D} = \{(o^{(i)}, a^{(i)})\}_{i=1}^N$ denote a dataset of expert demonstrations. The observation $o \in \mathcal{O}$ may include proprioception, multi-view RGB images, pointclouds, and language instructions. The action $a \in \mathbb{R}^{d_a H}$ represents a trajectory chunk of horizon H, parameterized typically as end-effector SE(3) poses or joint angles. The objective is to learn a conditional generative policy $\pi_\theta(a \mid o)$ that samples actions consistent with the expert distribution.

We extend the framework in Sec. 2.1 to the conditional setting by defining an observation-conditioned velocity field $\mathbf{v}_\theta(x, t \mid o)$. At deployment, the policy generates an action $\hat{a}$ by integrating the conditional ODE from a noise prior $\mathbf{z}_0 \sim p_0$ to the terminal state $\mathbf{z}_1$. The inference cost scales linearly with NFE.

During training, we employ an Optimal Transport (OT) [30,32] formulation to construct a straight-line conditional path interpolating between noise and data: $\mathbf{z}_t = (1-t)\,\epsilon + t\,a,\ \epsilon \sim p_0$. The derivative is constant: $\dot{\mathbf{z}}_t = a - \epsilon$. We train the conditional velocity field via the following regression objective:

$$\mathcal{L}_{\mathrm{policy}}(\theta) = \mathbb{E}_{(o,a)\sim\mathcal{D}}\ \mathbb{E}_{t\sim p_T,\epsilon\sim p_0}\big\|\mathbf{v}_\theta(\mathbf{z}_t,t \mid o) - (a-\epsilon)\big\|_2^2. \tag{3}$$

Here p_T is a chosen sampling distribution over $t \in [0,1]$.

3 One-Step Flow Policy

In this section, we introduce the One-Step Flow Policy (OFP), a from-scratch self-distillation framework designed for fast and high-precision visuomotor control. Sec. 3.1 details Self-Consistency Training, which learns an interval-averaged velocity field to enforce temporal coherence across trajectories. Sec. 3.2 proposes Self-Guided Regularization, a mechanism that explicitly sharpens single-step action predictions toward high-density expert modes. Sec. 3.3 introduces a

Warm-Start mechanism leveraging the temporal correlation of consecutive action chunks to provide a strong initialization prior.

3.1 Self-Consistency Training

To eliminate the reliance on iterative ODE integration during inference, we learn an interval-averaged velocity field rather than the instantaneous velocity used in standard Flow Matching [28, 30]. Concretely, we parameterize $\mathbf{u}_\theta(\mathbf{z}_t, t, r \mid o)$ for $0 \le t \le r \le 1$, representing the average velocity along the trajectory from t to r. For brevity, we omit the condition o when the context is clear.

Given $\mathbf{u}$, the update rule for any interval is defined as:

$$\mathbf{z}_r = \mathbf{z}_t + (r-t)\mathbf{u}(\mathbf{z}_t, t, r). \tag{4}$$

When $r \to t^+$, $\mathbf{u}(\mathbf{z}_t, t, r)$ reduces to the instantaneous velocity, maintaining theoretical consistency with Flow Matching on the diagonal.

Given a data-noise pair $((o,a),\epsilon) \sim \mathcal{D} \times p_0$, we sample two times $(t,r) \sim p_{T,R}$ subject to $t<r$, and draw an intermediate time $m \in [t,r]$. Using the OT straight-line interpolation, we obtain $\mathbf{z}_m = (1-m)\epsilon + ma$. An Exponential Moving Average (EMA) copy of the model, denoted $\mathbf{u}_{\theta^-}$, serves as the teacher to predict the trajectory endpoint $\hat{\mathbf{z}}_r = \mathbf{z}_m + (r-m)\mathbf{u}_{\theta^-}(\mathbf{z}_m, m, r)$. Based on the definition of average velocity over the interval $[t,r]$, we define the training target:

$$\mathbf{u}_{\text{target}} = \frac{\hat{\mathbf{z}}_r - \mathbf{z}_t}{r-t}. \tag{5}$$

We train $\mathbf{u}_\theta$ by minimizing the self-consistency loss:

$$\mathcal{L}_{\text{self-consistency}}(\theta) = \mathbb{E}_{(o,a)\sim\mathcal{D}}\ \mathbb{E}_{(t,r,m)\sim p_{T,R,M},\,\epsilon\sim p_0} \left\| \mathbf{u}_\theta(\mathbf{z}_t, t, r \mid o) - \mathbf{u}_{\text{target}} \right\|_2^2. \tag{6}$$

This objective functions as a self-distillation mechanism where the teacher is the slowly evolving EMA copy θ^-.

Time-Contracting Schedule. We sample m from a range that gradually contracts toward t as training proceeds:

$$m \sim \mathcal{U}\Big[t, t + (r-t)\rho(s)\Big]. \tag{7}$$

Here, $\rho(s)$ is a monotonically decreasing function of the training step s. In the early stages of training ($\rho(s) \approx 1$), m is sampled across the entire interval $[t,r]$. This allows the target $\mathbf{u}_{\text{target}}$ to rely more heavily on the interpolated state $\mathbf{z}_m$, rather than an untrained teacher $\mathbf{u}_{\theta^-}$. This effectively reduces bootstrapping error caused by an untrained teacher. As training progresses ($\rho(s) \to 0$), the sampling range collapses such that $m \to t$. This shifts the objective to enforce strict local self-consistency along the true trajectory.

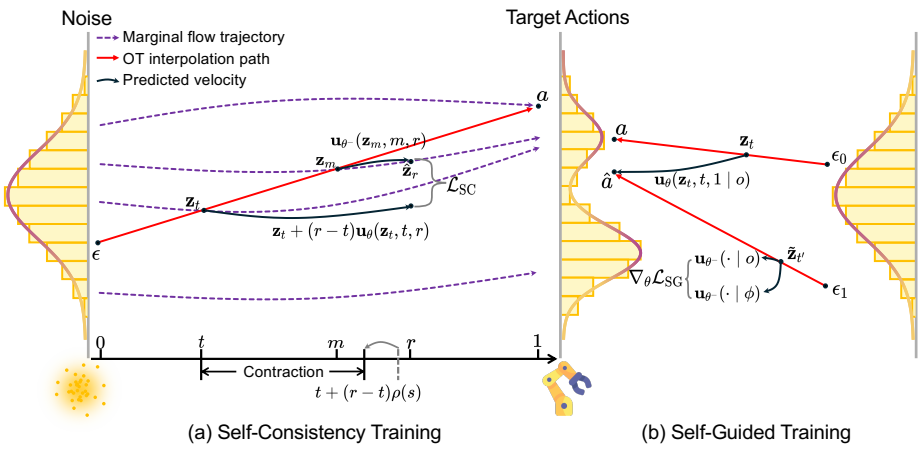

Fig. 2: Self-Distillation of One-Step Flow Policies. (a) **Self-Consistency Training:** The model learns an interval-averaged velocity field by matching predictions across nested sub-intervals, enforcing temporal coherence along the marginal flow trajectory. (b) **Self-Guided Training:** By leveraging Classifier-Free Guidance on the model's own predictions, we extract a distribution-level correction signal. The regularization repels single-step predictions from the unconditional prior and sharpens the generated actions toward the high-density modes of the expert data.

Proposition 1. *Under the assumption that the interval-velocity field is continuously differentiable and globally Lipschitz continuous in state, when the contracting factor* $\rho(s) \to 0$ *and the EMA teacher* $\mathbf{u}_{\theta^-}$ *is accurate,* $\mathbf{u}_{target}$ *becomes a consistent supervision signal for the ground truth average velocity* $\mathbf{u}(\mathbf{z}_t, t, r)$*, and minimizing* $\mathcal{L}_{self\text{-}consistency}$ *satisfies* $\mathbf{u}_\theta \approx \mathbf{u}$*.*

A detailed derivation is provided in the appendix. Proposition 1 establishes that the time-contracting schedule provides a smooth curriculum: it transitions from stabilizing initialization to refining trajectory precision.

Comparison with MeanFlow. Crucially, constructing the target $\mathbf{u}_{\text{target}}$ circumvents the costly Jacobian-Vector Product (JVP) computations required by MeanFlow [15, 16]. MeanFlow relies on a differential objective, which significantly increases memory overhead and wall-clock training time. In contrast, our approach relies solely on forward passes. We provide a formal proof in the appendix demonstrating that as the interval $(m - t) \to 0$, our consistency target converges to a finite-difference approximation of the derivative term in the MeanFlow objective.

Boundary Anchoring. To constrain the solution space and ensure the policy remains grounded in the expert data distribution, we introduce a boundary anchoring loss. This term applies standard Flow Matching supervision on the instantaneous velocity:

$$\mathcal{L}_{\text{flow}}(\theta) = \mathbb{E}_{(o,a)\sim\mathcal{D}}\, \mathbb{E}_{t\sim\mathcal{U}[0,1],\epsilon\sim p_0} \left\| \mathbf{u}_\theta(\mathbf{z}_t, t, t \mid o) - (a - \epsilon) \right\|_2^2. \tag{8}$$

This anchoring term ensures that OFP retains the capability for high-quality multi-step generation.

The self-consistency objective functions as a trajectory-matching mechanism. This formulation yields strong performance in few-step regimes (e.g., 2 to 4 steps) and preserves the diversity of the action distribution. However, we find that self-consistency alone often yields insufficient precision for complex manipulation in the single-step regime. To address this limitation, we introduce self-guided regularization in Sec. 3.2.

3.2 Self-Guided Regularization

While self-consistency (Sec. 3.1) constrains trajectory coherence across time intervals, it does not explicitly enforce that single-step predictions align with the high-density modes of the expert distribution.

To impose the distribution-level constraint, we introduce a score-based regularizer. The score function $\nabla_x \log p(x)$ points toward higher-density regions. We use it as a guidance signal to steer generated actions toward the expert data manifold.

One-step Sample and Re-noising. Let $\epsilon_0, \epsilon_1 \sim \mathcal{N}(0, I)$ be independent noise, and $(o, a) \sim \mathcal{D}$. For a time $t \in [0, 1]$, define the linear mixture $\mathbf{z}_t = (1-t)\epsilon_0 + ta$. The policy generates a one-step prediction by a single interval jump: $\hat{a} \triangleq \hat{\mathbf{z}}_1 = \mathbf{z}_t + (1-t)\mathbf{u}_\theta(\mathbf{z}_t, t, 1 \mid o)$. To compare the marginal distribution of the policy with the expert, we re-noise $\hat{a}$ at a new time $t' \sim \mathcal{U}[0, 1]$: $\tilde{\mathbf{z}}_{t'} = (1-t')\epsilon_1 + t'\hat{a}$.

Let $\pi_\theta(\tilde{\mathbf{z}}_{t'} \mid o)$ denote the policy-induced marginal distribution of $\tilde{\mathbf{z}}_{t'}$ conditioned on o, and let $\pi_\star(\tilde{\mathbf{z}}_{t'} \mid o)$ denote the corresponding expert distribution. We aim to minimize the reverse Kullback-Leibler (KL) divergence between $\pi_\theta(\cdot \mid o)$ and $\pi_\star(\cdot \mid o)$:

$$D_{\mathrm{KL}}\big(\pi_\theta(\tilde{\mathbf{z}}_{t'} \mid o) \,\|\, \pi_\star(\tilde{\mathbf{z}}_{t'} \mid o)\big) = \mathbb{E}_{\tilde{\mathbf{z}}_{t'} \sim \pi_\theta(\cdot|o)}\big[\log \pi_\theta(\tilde{\mathbf{z}}_{t'} \mid o) - \log \pi_\star(\tilde{\mathbf{z}}_{t'} \mid o)\big]. \quad (9)$$

The gradient of Eq. (9) with respect to θ can be written as a score-difference term:

$$\begin{aligned} &\nabla_\theta D_{\mathrm{KL}}(\pi_\theta \,\|\, \pi_\star) \\ &= \mathbb{E}\Big[\big(\nabla_{\tilde{\mathbf{z}}_{t'}} \log \pi_\theta(\tilde{\mathbf{z}}_{t'} \mid o) - \nabla_{\tilde{\mathbf{z}}_{t'}} \log \pi_\star(\tilde{\mathbf{z}}_{t'} \mid o)\big)\frac{\partial \tilde{\mathbf{z}}_{t'}}{\partial \theta}\Big] \\ &= \mathbb{E}\Big[\big(s_\theta(\tilde{\mathbf{z}}_{t'} \mid o) - s_\star(\tilde{\mathbf{z}}_{t'} \mid o)\big)\frac{\partial \tilde{\mathbf{z}}_{t'}}{\partial \theta}\Big], \end{aligned} \quad (10)$$

where $s(\cdot) = \nabla_{\tilde{\mathbf{z}}_{t'}} \log p(\cdot)$ denotes the score function.

Self-Guidance via CFG. To amplify the conditional signal, we incorporate Classifier-Free Guidance (CFG) [17]. Given a guidance scale $\alpha \geq 1$ and a null condition ϕ, the CFG-adjusted expert score is defined as: $s_\star^{\mathrm{CFG}}(\tilde{\mathbf{z}} \mid o) = s_\star(\tilde{\mathbf{z}} \mid o) + (\alpha - 1)\big(s_\star(\tilde{\mathbf{z}} \mid o) - s_\star(\tilde{\mathbf{z}} \mid \phi)\big)$. Substituting this form into Eq. (10) reveals a

decomposition into two components:

$$s_\theta(\tilde{\mathbf{z}}_{t'} \mid o) - s_\star^{\mathrm{CFG}}(\tilde{\mathbf{z}}_{t'} \mid o) = \\ (\alpha-1)\underbrace{\Big(s_\star(\tilde{\mathbf{z}}_{t'} \mid \phi) - s_\star(\tilde{\mathbf{z}}_{t'} \mid o)\Big)}_{\text{CFG Augmentation}} + \underbrace{\Big(s_\theta(\tilde{\mathbf{z}}_{t'} \mid o) - s_\star(\tilde{\mathbf{z}}_{t'} \mid o)\Big)}_{\text{Distribution Matching}}. \tag{11}$$

Recent analyses [29,53] of DMD-style training suggest that the CFG-Augmentation (CA) term is the primary driver of sample quality. Motivated by this, we use only the CA term as our guidance signal. Intuitively, the CA term steers the model toward the conditional distribution. Unlike prior works [22, 47] that rely on a pre-trained teacher to compute $s_\star$, we estimate the score using the EMA teacher $\mathbf{u}_{\theta^-}$, which yields a self-guided training signal.

For the OT probability path $\tilde{\mathbf{z}}_t = (1-t)\epsilon + ta$, the marginal score at time t is analytically related to the instantaneous velocity by:

$$s(\tilde{\mathbf{z}}_t \mid o) = \frac{t\mathbf{u}(\tilde{\mathbf{z}}_t, t, t \mid o) - \tilde{\mathbf{z}}_t}{1-t}. \tag{12}$$

We then construct the self-guidance target:

$$\mathbf{s}_{\text{target}} = \mathrm{sg}\Big[\mathbf{u}_\theta(\mathbf{z}_t, t, 1 \mid o) - \big(\mathbf{u}_{\theta^-}(\tilde{\mathbf{z}}_{t'}, t', t' \mid \phi) - \mathbf{u}_{\theta^-}(\tilde{\mathbf{z}}_{t'}, t', t' \mid o)\big)\Big], \tag{13}$$

where $\mathrm{sg}[\cdot]$ denotes the stop-gradient operator.

We formulate the self-guidance loss as the squared error between the one-step prediction and the guided target:

$$\mathcal{L}_{\text{self-guidance}}(\theta) = \mathbb{E}_{(o,a)\sim\mathcal{D}}\, \mathbb{E}_{t,t'\sim\mathcal{U}[0,1],\, \epsilon\sim p_0}\, \Big[\big\|\mathbf{u}_\theta(\mathbf{z}_t, t, 1 \mid o) - \mathbf{s}_{\text{target}}\big\|^2\Big]. \tag{14}$$

Minimizing Eq. (14) yields a gradient for $\mathbf{u}_\theta$ that aligns the update direction with the CA term (proof provided in Appendix), which effectively repels the generation from the unconditional mode. Since the guidance signal is induced by the model itself, this objective can also be viewed as a form of self-distillation.

The self-guidance term requires an unconditional branch. We implement this using condition dropout: during training, we randomly replace the observation o with a null token ϕ with probability p_{drop}, enabling a single network to capture both conditional and unconditional dynamics.

Unified Objective. We combine the flow anchoring loss (Eq. (8)), self-consistency loss (Eq. (6)), and self-guidance loss (Eq. (14)) into a unified training objective:

$$\mathcal{L}_{\text{self-distill}}(\theta) = \mathcal{L}_{\text{flow}}(\theta) + \lambda_c \mathcal{L}_{\text{self-consistency}}(\theta) + \lambda_g \mathcal{L}_{\text{self-guidance}}(\theta). \tag{15}$$

Empirically, adding self-guidance significantly sharpens the precision of one-step outputs. The combination of these signals enables OFP to achieve state-of-the-art performance in both few-step and one-step generation. The complete training procedure is outlined in Algorithm 1.

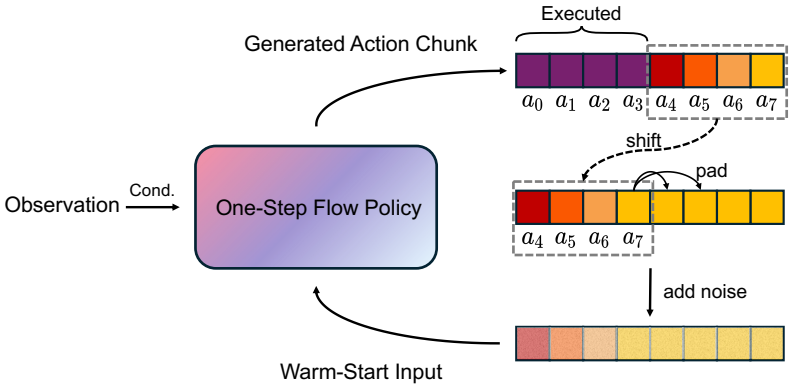

Fig. 3: Warm-Start Action Prior for One-Step Inference. The unexecuted suffix of the previously generated action chunk is shifted and padded with the terminal action to form a full-length temporal prior. By starting closer to the target data manifold rather than from pure Gaussian noise, this initialization reduces the required transport distance.

Inference. OFP supports flexible sampling strategies. For minimal latency, we generate a one-step action prediction $\hat{a}$ via a single global jump from noise: $\hat{a} = \epsilon + \mathbf{u}_\theta(\epsilon, 0, 1 \mid o), \epsilon \sim \mathcal{N}(0, I)$. Alternatively, to achieve higher precision, the model can function as a multi-step solver by chaining interval predictions. Given a time discretization schedule $0 = \tau_0 < \tau_1 < \cdots < \tau_K = 1$, we recursively update the state by querying the average velocity over each sub-interval $[\tau_k, \tau_{k+1}]$: $\hat{\mathbf{z}}_{\tau_{k+1}} = \hat{\mathbf{z}}_{\tau_k} + (\tau_{k+1} - \tau_k)\mathbf{u}_\theta(\hat{\mathbf{z}}_{\tau_k}, \tau_k, \tau_{k+1} \mid o)$. This allows the policy to trade increased inference time for finer control.

3.3 Warm-Start for One-Step Inference

Warm-start is widely used [9, 21] in action-chunking generative policies to improve smoothness and execution efficiency in receding-horizon rollouts. Here, we repurpose warm-start for reducing the generative burden of single-step inference. By leveraging the high temporal correlation between consecutive chunks, we construct a strong prior that lowers the transport cost required to reach the target action distribution.

In a receding-horizon control loop with execution horizon $h < H$, the robot executes the first h actions. The unexecuted suffix represents a valid plan for the current step, $a_{\text{left}} = [a_{h+1}, \ldots, a_H] \in \mathbb{R}^{d_a(H-h)}$. We construct a full-length warm-start prior a_{warm} by shifting the unexecuted suffix and padding the terminal action:

$$a_{\text{warm}} = \big[a_{h+1}, \ldots, a_H, \underbrace{a_H, \ldots, a_H}_{h \text{ times}}\big] \in \mathbb{R}^{d_a H}. \tag{16}$$

Warm-Started Generation. Rather than sampling from non-informative Gaussian noise, we initialize the generator from a noised projection of the warm-start prior. We select a noise level $t_w \in (0, 1]$ and sample $\epsilon \sim \mathcal{N}(0, I)$ to construct the initial state: $\mathbf{z}_{t_w} = (1 - t_w)\epsilon + t_w a_{\text{warm}}$. The policy can then generate the next action chunk $\hat{a}$ via a single interval update: $\hat{a} = \mathbf{z}_{t_w} + (1 - t_w)\mathbf{u}_\theta(\mathbf{z}_{t_w}, t_w, 1 \mid o)$. This formulation aligns structurally with the renoising form used in Sec. 3.2, making the inference-time initialization consistent with the training-time guidance construction. We illustrate the construction of this warm-start prior and its integration into the receding-horizon control loop in Fig. 3.

Algorithm 1 OFP Training

Inputs: expert dataset $\mathcal{D}$, model $\mathbf{u}_\theta$, condition dropout p_{drop}.
Initialization: EMA teacher $\theta^- \leftarrow \theta$.
while *not converged* **do**
  Sample batch $(o, a) \sim \mathcal{D}$, time variables, and noise.
  Apply condition dropout: $o \leftarrow \phi$ with probability p_{drop}.
  Compute $\mathcal{L}_{\text{flow}}$ via Eq. (8)
  Compute $\mathcal{L}_{\text{self-consistency}}$ via Eq. (6)
  Compute $\mathcal{L}_{\text{self-guidance}}$ via Eq. (14)
  Update θ via Eq. (15)
  Update EMA teacher.
end

Initializing with a_{warm} significantly reduces the transport distance the flow must bridge in a single step. By starting closer to the data manifold, the policy yields higher precision and improved temporal smoothness without requiring additional training or computational overhead.

4 Experiments

To evaluate the effectiveness of OFP, we conduct extensive simulation experiments across 4 manipulation benchmarks: Adroit [25], DexArt [3], MetaWorld [52], and RoboTwin 2.0 [8]. Our evaluation answers three core questions: 1) Can OFP match or exceed the performance of standard multi-step diffusion and flow policies while requiring only a single network evaluation? 2) How does OFP compare against existing single-step acceleration methods in terms of stability, accuracy, and scaling capability? 3) Does OFP scale effectively to modern VLA architectures without experiencing performance collapse?

4.1 Experimental Setup

We divide our evaluation into 2D image-based and 3D pointcloud-based control settings. For the 2D evaluation, we test on 3 tasks from Adroit [25] and 4 tasks from DexArt [3]. For the 3D evaluation, we test on these 7 tasks alongside 49 tasks from MetaWorld [52]. The Adroit and DexArt evaluate single-task learning, while MetaWorld evaluates large-scale multi-task learning. Following prior work [41], we group the 49 MetaWorld tasks into 4 empirical difficulty levels: Easy, Medium, Hard, and Very Hard.

Table 1: Performance on 2D image-based manipulation. Results are averaged over 3 random seeds. The best performance in each column is highlighted in **bold**.

Method	NFE	Door	Hammer	Pen	Bucket	Faucet	Laptop	Toilet	Average
DP	100	51.7±4.0	100.0±0.0	67.7±1.2	**34.7±4.7**	38.7±1.5	80.3±2.9	76.0±1.7	64.2±2.3
DP	10	44.7±1.5	100.0±0.0	66.7±1.5	32.7±4.0	35.0±2.6	81.0±2.6	66.0±3.6	60.9±2.3
FM Policy	100	63.7±2.1	100.0±0.0	**69.7±2.1**	31.7±1.2	40.7±2.3	84.7±2.5	79.7±1.5	67.2±1.7
FM Policy	10	61.3±3.2	97.3±3.1	64.7±3.8	27.7±2.3	32.7±3.8	**87.0±1.7**	77.3±1.5	64.0±2.8
CP	1	46.3±3.5	100.0±0.0	58.7±5.7	30.7±2.5	33.3±0.6	79.0±2.6	69.7±3.1	59.7±2.6
OneDP	1	53.3±5.1	100.0±0.0	64.7±1.5	32.7±1.2	40.0±1.7	76.3±2.9	76.3±2.3	63.3±2.1
MP1	1	49.3±6.0	96.3±3.5	57.3±4.9	31.7±6.7	33.0±7.5	83.7±1.5	72.3±4.6	60.5±5.0
OFP (Ours)	1	**68.3±3.1**	**100.0±0.0**	69.3±1.5	33.3±2.5	**41.0±1.7**	85.7±3.5	**80.7±2.5**	**68.3±2.1**

Table 2: Performance on 3D pointcloud manipulation. We assess performance on 56 challenging tasks with 3 random seeds.

Method	NFE	Adroit (3)	DexArt (4)	MetaWorld Easy (28)	MetaWorld Medium (11)	MetaWorld Hard (5)	MetaWorld Very Hard (5)	Average
DP3	100	79.0±0.6	63.7±0.6	82.8±8.0	46.5±3.8	38.3±9.6	41.0±0.6	66.4±5.7
DP3	10	76.0±3.3	61.5±0.9	80.8±2.1	46.2±1.4	36.3±8.2	45.3±5.7	65.2±2.8
FM Policy	100	83.3±0.7	61.7±4.2	68.9±5.7	42.3±3.8	**43.6±3.4**	47.7±4.5	59.8±4.6
FM Policy	10	80.3±1.8	60.5±4.5	66.7±7.7	41.3±7.4	40.3±7.2	46.7±7.6	57.9±7.0
CP	1	76.3±4.6	60.7±3.4	68.3±1.8	32.1±4.3	28.7±5.2	36.0±2.3	54.7±2.9
OneDP	1	77.3±1.8	62.5±4.1	77.7±0.5	39.6±9.2	38.7±4.4	41.7±1.6	62.4±3.0
MP1	1	82.0±2.1	61.0±2.0	70.5±6.1	37.9±5.9	34.3±9.5	32.3±5.5	57.4±5.8
OFP (Ours)	1	**85.0±4.5**	**64.3±2.5**	**87.9±4.6**	**52.4±5.7**	43.3±2.4	**49.2±0.7**	**71.6±4.1**

Baselines. We compare OFP against five strong generative policy baselines: Diffusion Policy (DP) [9] and Flow Matching Policy (FM Policy) [56] as multi-step controls; Consistency Policy (CP) [39] and One-Step Diffusion Policy (OneDP) [47] representing distillation approaches; and MP1 [42] adapting MeanFlow [15]. For 3D settings, we utilize DP3 [55] and adapt FM Policy with an identical point encoder. CP and OneDP are distilled from a pre-trained DP3 teacher, whereas OFP and MP1 are trained from scratch. All evaluations share identical settings and are averaged over 3 random seeds (reported as mean ± standard deviation). Detailed experimental settings and evaluation metrics can be found in the appendix.

4.2 Experimental Results

2D Image-Conditioned Evaluation. Tab. 1 reports the results on the 2D tasks. OFP achieves the best average success rate among all one-step methods. It also surpasses the strong multi-step baselines. At NFE= 1, OFP reaches 68.3%, which is higher than DP at NFE= 100 with 64.2% and FM Policy at NFE= 100 with 67.2%. This shows that the proposed self-distillation objective does more than reduce latency. It also significantly improves final action quality. During training, MP1 showed high variance and frequent loss spikes. This is consistent with the extra numerical sensitivity introduced by the JVP operations in its

Table 3: Few-Step vs. One-Step Generation. OFP supports flexible sampling, maintaining high performance at 1 step while reliably improving accuracy given more generation steps.

Method	NFE	Bucket	Faucet	Laptop	Toilet	Average
OneDP	1	32.7±5.8	44.7±3.1	89.3±3.8	**83.3±3.8**	62.5±4.1
CP	1	29.3±3.5	44.3±3.8	86.7±2.5	82.3±3.8	60.7±3.4
	4	38.3±4.2	43.7±1.2	90.0±3.5	84.7±2.9	64.2±2.9
OFP	1	39.3±0.6	45.7±3.8	91.7±3.2	81.3±3.2	64.5±2.7
	4	**42.7±2.1**	**47.0±1.0**	**92.3±4.2**	83.0±1.7	**66.2±2.2**

objective. OFP avoids JVPs by training on an interval-averaged target, which leads to more stable optimization and better control accuracy.

3D Pointcloud-Conditioned Evaluation. Tab. 2 shows an even larger gain in the 3D setting. OFP achieves the highest success rates in the vast majority of tasks. At NFE= 1, it outperforms DP3 at NFE= 100 by 8% on the average, and it exceeds the 3D FM Policy at NFE= 100 by 19.7%. The gain is especially clear in the MetaWorld multi-task setting, where OFP improves both average performance and robustness across difficulty levels. These results support the role of self-guided regularization, which sharpens the action distribution and reduces the over-smoothing often seen in standard flow objectives.

Fig. 1 shows the trade-off between inference time and success rate across all 56 tasks. OFP reaches the best average success rate while staying in the one-step low-latency regime. In wall-clock time, OFP requires 17.58 ms per action chunk, compared with 3225.67 ms for DP3 at NFE= 100 and 1865.72 ms for 3D FM Policy at NFE= 100. This corresponds to about 183× and 106× speedups, respectively. OFP is slightly slower than OneDP and MP1 because warm-start adds a small amount of computation, but the accuracy gain is large enough to justify that cost.

Latency and Accuracy Trade-off. A basic limitation of OneDP-like methods is that they only support one-step inference. By contrast, trajectory-matching methods such as CP support multi-step generation but often suffer less precise single-step prediction. OFP combines the strengths of both. As shown in Tab. 3, OneDP is stronger than CP at NFE= 1 on several tasks. However, CP improves once more steps are allowed. OFP remains strong at NFE= 1 and improves further at NFE= 4, which shows that the same model can provide both low-latency one-step control and better accuracy when a slightly larger compute budget is available.

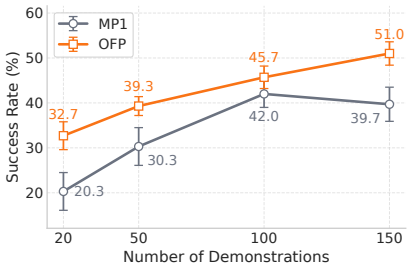

Fig. 4: Data Scaling Behavior. OFP extracts higher utility from sparse data (20 demos) and continues to scale cleanly as data increases, avoiding the performance degradation seen in MP1 at 150 demos.

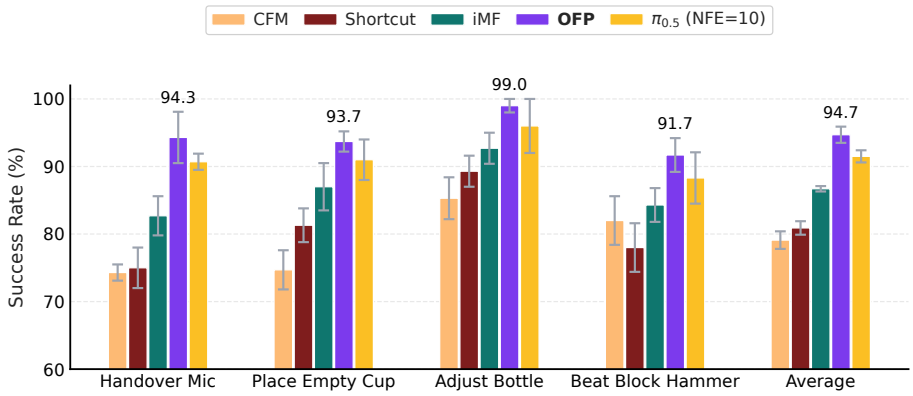

Fig. 5: Transfer to $\pi_{0.5}$ on RoboTwin 2.0. All acceleration methods are evaluated at NFE=1, and the $\pi_{0.5}$ baseline operates at NFE=10. OFP achieves the best average success rate across four tasks, showing that OFP remains effective even for large-scale VLA models with richer multi-modal inputs.

Fig. 6: Visualization of RoboTwin 2.0 tasks. Evaluations use domain randomization to test robustness under visual variation.

Ablation Studies. To isolate the contributions of our core components, we conducted targeted ablation studies. Specifically, we independently ablate each module under fixed training conditions to evaluate its impact on single-step and few-step success rates. Our results show that self-consistency training is necessary for reliable few-step inference, while self-guided regularization drives the performance gains in the single-step regime. Empirically, self-consistency and self-guided training complement each other. Additionally, the warm-start mechanism acts as a training-free inference prior that consistently improves generation quality across any step count. Unifying these mechanisms allows OFP to deliver high accuracy in both single-step and few-step control. We report the detailed results of these ablations in the appendix.

Data Efficiency and Scaling. We investigate how performance scales with data availability on the DexArt Faucet task (Fig. 4). Under very limited data, with only 20 demonstrations, MP1 degrades sharply. In contrast, OFP maintains a robust success rate, indicating that self-guided regularization provides stronger learning signals in the low-data regime. As the number of demonstra-

tions increases, OFP improves steadily from 32.7% to 51.0%. MP1 improves at first, but then drops after 100 demonstrations.

4.3 Integration with VLA Models

Modern VLA systems often require large over-parameterized models and condition action generation on rich multi-modal semantics. However, bootstrapping-based methods may suffer from rank-collapse when model capacity increases [12, 24]. To verify whether OFP remains effective in this regime, we integrate it into the $\pi_{0.5}$ model and evaluate it in RoboTwin 2.0. Fig. 6 visualizes the 4 RoboTwin 2.0 tasks used in our $\pi_{0.5}$ integration study. We evaluate OFP under domain randomization (clutter/distractors, background textures, lighting, and tabletop height). We compare with three representative from-scratch acceleration methods: Consistency Flow Matching (CFM) [49], Shortcut Models [12], and Improved MeanFlow (iMF) [16].

Fig. 5 shows that OFP achieves the best average success rate, reaching 94.7% across the four tasks. More importantly, OFP at NFE= 1 surpasses the original $\pi_{0.5}$ policy at NFE= 10. This result shows that OFP does not collapse in a larger VLA setting with richer semantics. Instead, it preserves the benefits of single-step generation while still improving control quality. This confirms that OFP is not tied to a narrow policy class. The self-distillation design continues to work when the backbone is larger, the conditioning is richer, and the task semantics are more complex.

5 Related Work

5.1 Generative Policies for Robot Control

The integration of diffusion models into robotics initially emerged through the planning-as-generation paradigm [1, 21]. Subsequently, the focus shifted toward visuomotor imitation: Diffusion Policy [9] established a robust baseline by parameterizing conditional policies as iterative denoising processes over action chunks. While this approach excels at modeling multimodal distributions and ensures training stability, it incurs high inference latency that scales linearly with the number of denoising steps. Subsequent research focuses on stronger conditional representations [26, 55] and more structured generation [38]. In parallel, Flow Matching policies [6, 10, 13, 18, 56, 57] leverage an ODE-based velocity field formulation. The flow-based paradigm is gradually converging with modern Vision-Language-Action (VLA) architectures [4, 5, 19, 20, 31, 43].

5.2 Distillation and Acceleration

Few-step acceleration of generative policies mainly follows two routes.

Consistency Distillation. Building on Consistency Models [45], approaches like Consistency Policy [39] and ManiCM [35] distill a pre-trained diffusion

teacher into a student. However, they are more likely to exhibit averaging under highly multimodal action distributions, resulting in insufficient one-step action precision.

Score Distillation. Inspired by Distribution Matching Distillation (DMD) [51], methods such as One-Step Diffusion Policy [47] and SDM Policy [22] utilize score-based gradients to optimize a one-step generator. While this approach produces sharper, mode-seeking samples ideal for precision tasks, it typically sacrifices distribution diversity and relies heavily on pre-trained teacher models.

Our approach unifies these two paradigms within a from-scratch self-distillation framework, achieving high-fidelity few-step and single-step generation without an external teacher. See Appendix for a detailed version.

6 Conclusion & Limitation

We introduce OFP, a self-distillation framework that accelerates flow-based visuomotor policies to the one-step regime while maintaining high control precision. By unifying self-consistency and self-guided training, OFP achieves over a 100× speedup against multi-step baselines, setting state-of-the-art success rates across diverse manipulation benchmarks and transferring effectively to larger VLA backbones like $\pi_{0.5}$. While OFP demonstrates robust performance across diverse tasks, our current evaluations are conducted in simulation. A direct next step is to evaluate OFP on physical robot systems. OFP is orthogonal to system-level acceleration techniques. Future work can seamlessly integrate OFP with model quantization and structural pruning to further decrease inference latency.

Acknowledgements

The computations in this work were run on the FASRC cluster supported by the FAS Division of Science Research Computing Group at Harvard University.

References

1. Ajay, A., Du, Y., Gupta, A., Tenenbaum, J., Jaakkola, T., Agrawal, P.: Is conditional generative modeling all you need for decision-making? arXiv preprint arXiv:2211.15657 (2022)
2. Ajay, A., Han, S., Du, Y., Li, S., Gupta, A., Jaakkola, T., Tenenbaum, J., Kaelbling, L., Srivastava, A., Agrawal, P.: Compositional foundation models for hierarchical planning. Advances in Neural Information Processing Systems **36**, 22304–22325 (2023)
3. Bao, C., Xu, H., Qin, Y., Wang, X.: Dexart: Benchmarking generalizable dexterous manipulation with articulated objects. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21190–21200 (2023)
4. Bjorck, J., Castañeda, F., Cherniadev, N., Da, X., Ding, R., Fan, L., Fang, Y., Fox, D., Hu, F., Huang, S., et al.: Gr00t n1: An open foundation model for generalist humanoid robots. arXiv preprint arXiv:2503.14734 (2025)

5. Black, K., Brown, N., Driess, D., Esmail, A., Equi, M., Finn, C., Fusai, N., Groom, L., Hausman, K., Ichter, B., et al.: π_0: A vision-language-action flow model for general robot control. arXiv preprint arXiv:2410.24164 (2024)
6. Braun, M., Jaquier, N., Rozo, L., Asfour, T.: Riemannian flow matching policy for robot motion learning. In: 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 5144–5151. IEEE (2024)
7. Cheang, C.L., Chen, G., Jing, Y., Kong, T., Li, H., Li, Y., Liu, Y., Wu, H., Xu, J., Yang, Y., et al.: Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation. arXiv preprint arXiv:2410.06158 (2024)
8. Chen, T., Chen, Z., Chen, B., Cai, Z., Liu, Y., Li, Z., Liang, Q., Lin, X., Ge, Y., Gu, Z., et al.: Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. arXiv preprint arXiv:2506.18088 (2025)
9. Chi, C., Xu, Z., Feng, S., Cousineau, E., Du, Y., Burchfiel, B., Tedrake, R., Song, S.: Diffusion policy: Visuomotor policy learning via action diffusion. The International Journal of Robotics Research **44**(10-11), 1684–1704 (2025)
10. Chisari, E., Heppert, N., Argus, M., Welschehold, T., Brox, T., Valada, A.: Learning robotic manipulation policies from point clouds with conditional flow matching. arXiv preprint arXiv:2409.07343 (2024)
11. Fan, C., Jia, X., Sun, Y., Wang, Y., Wei, J., Gong, Z., Zhao, X., Tomizuka, M., Yang, X., Yan, J., et al.: Interleave-vla: Enhancing robot manipulation with interleaved image-text instructions. arXiv preprint arXiv:2505.02152 (2025)
12. Frans, K., Hafner, D., Levine, S., Abbeel, P.: One step diffusion via shortcut models. arXiv preprint arXiv:2410.12557 (2024)
13. Funk, N., Urain, J., Carvalho, J., Prasad, V., Chalvatzaki, G., Peters, J.: Actionflow: Equivariant, accurate, and efficient policies with spatially symmetric flow matching. arXiv preprint arXiv:2409.04576 (2024)
14. Gbagbe, K.F., Cabrera, M.A., Alabbas, A., Alyunes, O., Lykov, A., Tsetserukou, D.: Bi-vla: Vision-language-action model-based system for bimanual robotic dexterous manipulations. In: 2024 IEEE International Conference on Systems, Man, and Cybernetics (SMC). pp. 2864–2869. IEEE (2024)
15. Geng, Z., Deng, M., Bai, X., Kolter, J.Z., He, K.: Mean flows for one-step generative modeling. arXiv preprint arXiv:2505.13447 (2025)
16. Geng, Z., Lu, Y., Wu, Z., Shechtman, E., Kolter, J.Z., He, K.: Improved mean flows: On the challenges of fastforward generative models. arXiv preprint arXiv:2512.02012 (2025)
17. Ho, J., Salimans, T.: Classifier-free diffusion guidance. arXiv preprint arXiv:2207.12598 (2022)
18. Hu, X., Liu, Q., Liu, X., Liu, B.: Adaflow: Imitation learning with variance-adaptive flow-based policies. Advances in Neural Information Processing Systems **37**, 138836–138858 (2024)
19. Intelligence, P., Amin, A., Aniceto, R., Balakrishna, A., Black, K., Conley, K., Connors, G., Darpinian, J., Dhabalia, K., DiCarlo, J., et al.: $\pi^*_{0.6}$: a vla that learns from experience. arXiv preprint arXiv:2511.14759 (2025)
20. Intelligence, P., Black, K., Brown, N., Darpinian, J., Dhabalia, K., Driess, D., Esmail, A., Equi, M., Finn, C., Fusai, N., et al.: $\pi^*_{0.5}$: a vision-language-action model with open-world generalization. arXiv preprint arXiv:2504.16054 (2025)
21. Janner, M., Du, Y., Tenenbaum, J.B., Levine, S.: Planning with diffusion for flexible behavior synthesis. arXiv preprint arXiv:2205.09991 (2022)

22. Jia, B., Ding, P., Cui, C., Sun, M., Qian, P., Huang, S., Fan, Z., Wang, D.: Score and distribution matching policy: Advanced accelerated visuomotor policies via matched distillation. arXiv preprint arXiv:2412.09265 (2024)
23. Kim, D., Lai, C.H., Liao, W.H., Murata, N., Takida, Y., Uesaka, T., He, Y., Mitsufuji, Y., Ermon, S.: Consistency trajectory models: Learning probability flow ode trajectory of diffusion. arXiv preprint arXiv:2310.02279 (2023)
24. Kumar, A., Agarwal, R., Ghosh, D., Levine, S.: Implicit under-parameterization inhibits data-efficient deep reinforcement learning. arXiv preprint arXiv:2010.14498 (2020)
25. Kumar, V.: Manipulators and Manipulation in high dimensional spaces. Ph.D. thesis (2016)
26. Li, X., Belagali, V., Shang, J., Ryoo, M.S.: Crossway diffusion: Improving diffusion-based visuomotor policy via self-supervised learning. In: 2024 IEEE International Conference on Robotics and Automation (ICRA). pp. 16841–16849. IEEE (2024)
27. Lin, F., Nai, R., Hu, Y., You, J., Zhao, J., Gao, Y.: Onetwovla: A unified vision-language-action model with adaptive reasoning. arXiv preprint arXiv:2505.11917 (2025)
28. Lipman, Y., Chen, R.T., Ben-Hamu, H., Nickel, M., Le, M.: Flow matching for generative modeling. arXiv preprint arXiv:2210.02747 (2022)
29. Liu, D., Gao, P., Liu, D., Du, R., Li, Z., Wu, Q., Jin, X., Cao, S., Zhang, S., Li, H., et al.: Decoupled dmd: Cfg augmentation as the spear, distribution matching as the shield. arXiv preprint arXiv:2511.22677 (2025)
30. Liu, Q.: Rectified flow: A marginal preserving approach to optimal transport. arXiv preprint arXiv:2209.14577 (2022)
31. Liu, S., Wu, L., Li, B., Tan, H., Chen, H., Wang, Z., Xu, K., Su, H., Zhu, J.: Rdt-1b: a diffusion foundation model for bimanual manipulation. arXiv preprint arXiv:2410.07864 (2024)
32. Liu, X., Gong, C., Liu, Q.: Flow straight and fast: Learning to generate and transfer data with rectified flow. arXiv preprint arXiv:2209.03003 (2022)
33. Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., Zhu, J.: Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. Advances in neural information processing systems **35**, 5775–5787 (2022)
34. Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., Zhu, J.: Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. Machine Intelligence Research pp. 1–22 (2025)
35. Lu, G., Gao, Z., Chen, T., Dai, W., Wang, Z., Ding, W., Tang, Y.: Manicm: Real-time 3d diffusion policy via consistency model for robotic manipulation. arXiv preprint arXiv:2406.01586 (2024)
36. Lu, Y., Ren, Y., Xia, X., Lin, S., Wang, X., Xiao, X., Ma, A.J., Xie, X., Lai, J.H.: Adversarial distribution matching for diffusion distillation towards efficient image and video synthesis. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 16818–16829 (2025)
37. Lukoianov, A., Sáez de Ocáriz Borde, H., Greenewald, K., Guizilini, V., Bagautdinov, T., Sitzmann, V., Solomon, J.M.: Score distillation via reparametrized ddim. Advances in Neural Information Processing Systems **37**, 26011–26044 (2024)
38. Ma, X., Patidar, S., Haughton, I., James, S.: Hierarchical diffusion policy for kinematics-aware multi-task robotic manipulation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18081–18090 (2024)

39. Prasad, A., Lin, K., Wu, J., Zhou, L., Bohg, J.: Consistency policy: Accelerated visuomotor policies via consistency distillation. arXiv preprint arXiv:2405.07503 (2024)
40. Salimans, T., Ho, J.: Progressive distillation for fast sampling of diffusion models. arXiv preprint arXiv:2202.00512 (2022)
41. Seo, Y., Hafner, D., Liu, H., Liu, F., James, S., Lee, K., Abbeel, P.: Masked world models for visual control. In: Conference on Robot Learning. pp. 1332–1344. PMLR (2023)
42. Sheng, J., Wang, Z., Li, P., Liu, M.: Mp1: Meanflow tames policy learning in 1-step for robotic manipulation. arXiv preprint arXiv:2507.10543 (2025)
43. Shi, L.X., Ichter, B., Equi, M., Ke, L., Pertsch, K., Vuong, Q., Tanner, J., Walling, A., Wang, H., Fusai, N., et al.: Hi robot: Open-ended instruction following with hierarchical vision-language-action models. arXiv preprint arXiv:2502.19417 (2025)
44. Song, J., Meng, C., Ermon, S.: Denoising diffusion implicit models. arXiv preprint arXiv:2010.02502 (2020)
45. Song, Y., Dhariwal, P., Chen, M., Sutskever, I.: Consistency models (2023)
46. Wang, P., Xu, D., Fan, Z., Wang, D., Mohan, S., Iandola, F., Ranjan, R., Li, Y., Liu, Q., Wang, Z., et al.: Taming mode collapse in score distillation for text-to-3d generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9037–9047 (2024)
47. Wang, Z., Li, Z., Mandlekar, A., Xu, Z., Fan, J., Narang, Y., Fan, L., Zhu, Y., Balaji, Y., Zhou, M., et al.: One-step diffusion policy: Fast visuomotor policies via diffusion distillation. arXiv preprint arXiv:2410.21257 (2024)
48. Xu, Y., Nie, W., Vahdat, A.: One-step diffusion models with f-divergence distribution matching. arXiv preprint arXiv:2502.15681 (2025)
49. Yang, L., Zhang, Z., Zhang, Z., Liu, X., Xu, M., Zhang, W., Meng, C., Ermon, S., Cui, B.: Consistency flow matching: Defining straight flows with velocity consistency. arXiv preprint arXiv:2407.02398 (2024)
50. Yin, T., Gharbi, M., Park, T., Zhang, R., Shechtman, E., Durand, F., Freeman, B.: Improved distribution matching distillation for fast image synthesis. Advances in neural information processing systems **37**, 47455–47487 (2024)
51. Yin, T., Gharbi, M., Zhang, R., Shechtman, E., Durand, F., Freeman, W.T., Park, T.: One-step diffusion with distribution matching distillation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 6613–6623 (2024)
52. Yu, T., Quillen, D., He, Z., Julian, R., Hausman, K., Finn, C., Levine, S.: Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In: Conference on robot learning. pp. 1094–1100. PMLR (2020)
53. Yu, X., Qi, X., Li, Z., Zhang, K., Zhang, R., Lin, Z., Shechtman, E., Wang, T., Nitzan, Y.: Self-evaluation unlocks any-step text-to-image generation. arXiv preprint arXiv:2512.22374 (2025)
54. Yuan, Z., Qian, C., Tang, J., Chen, R., Song, Z., Sun, L., Chu, X., Cai, Y., Zhang, D., Li, S.: Autodrive-r ^{2}: Incentivizing reasoning and self-reflection capacity for vla model in autonomous driving. arXiv preprint arXiv:2509.01944 (2025)
55. Ze, Y., Zhang, G., Zhang, K., Hu, C., Wang, M., Xu, H.: 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. arXiv preprint arXiv:2403.03954 (2024)
56. Zhang, F., Gienger, M.: Affordance-based robot manipulation with flow matching. arXiv preprint arXiv:2409.01083 (2024)

57. Zhang, Q., Liu, Z., Fan, H., Liu, G., Zeng, B., Liu, S.: Flowpolicy: Enabling fast and robust 3d flow-based policy via consistency flow matching for robot manipulation. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 39, pp. 14754–14762 (2025)
58. Zheng, R., Liang, Y., Huang, S., Gao, J., Daumé III, H., Kolobov, A., Huang, F., Yang, J.: Tracevla: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. arXiv preprint arXiv:2412.10345 (2024)
59. Zhou, Z., Cai, T., Zhao, S.Z., Zhang, Y., Huang, Z., Zhou, B., Ma, J.: Autovla: A vision-language-action model for end-to-end autonomous driving with adaptive reasoning and reinforcement fine-tuning. arXiv preprint arXiv:2506.13757 (2025)