PRIMARY-FINE DECOUPLING FOR ACTION GENERATION IN ROBOTIC IMITATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-modal distribution in robotic manipulation action sequences poses critical challenges for imitation learning. To this end, existing approaches often model the action space as either a discrete set of tokens or a continuous, latent-variable distribution. However, both approaches present trade-offs: some methods discretize actions into tokens and therefore lose fine-grained action variations, while others generate continuous actions in a single stage tend to produce unstable mode transitions. To address these limitations, we propose Primary-Fine Decoupling for Action Generation (PF-DAG), a two-stage framework that decouples coarse action consistency from fine-grained variations. First, we compress action chunks into a small set of discrete modes, enabling a lightweight policy to select consistent coarse modes and avoid mode bouncing. Second, a mode conditioned MeanFlow policy is learned to generate high-fidelity continuous actions. Theoretically, we prove PF-DAG's two-stage design achieves a strictly lower MSE bound than single-stage generative policies. Empirically, PF-DAG outperforms state-of-the-art baselines across 56 tasks from Adroit, DexArt, and MetaWorld benchmarks. It further generalizes to real-world tactile dexterous manipulation tasks. Our work demonstrates that explicit mode-level decoupling enables both robust multi-modal modeling and reactive closed-loop control for robotic manipulation.

1 INTRODUCTION

In robotic manipulation, capturing multi-modal distributions in action sequences is essential for learning robust and reliable imitation policies (Florence et al., 2022; Chi et al., 2023). Offline expert trajectories often admit multiple valid actions for the same or similar observations: for example, when an obstacle lies in front of the end-effector, demonstrators may steer either left or right. This richness of valid behaviors complicates learning from offline data and thus motivates the development of various imitation learning approaches to tackle this challenge.

Among these imitation learning approaches, Behavioral Cloning (BC) treats policy learning as supervised regression $a = \pi(o)$ and therefore commonly collapses multiple valid actions into a single mean (Levine et al., 2016; Torabi et al., 2018), as visualized in Figure 1 (a). Action discretization represents multiple modes by predicting categorical bins (Brohan et al., 2022; Zitkovich et al., 2023; Kim et al., 2024), but coarse discretization introduces reconstruction error and temporal discontinuities (Shafiullah et al., 2022), failing to match the smoothness of human demonstrations (see Figure 1 (b)). Generative latent-variable methods instead model $a = \pi(o, z)$ so that sampling different z yields different plausible actions (Zhao et al., 2023; Chi et al., 2023). However, independent per-step resampling of z tends to produce random switches among modes (Chen et al., 2025) (see Figure 1 (c)). Such erratic transitions directly lead to trajectory discontinuities and end-effector pose instability, while further undermining the overall task execution accuracy.

We observe that many manipulation tasks naturally decompose actions into a small set of discrete, interpretable *primary modes* (coarse prototypes such as "lift-and-fold" or "lift-and-rotate") together with continuous, within-mode variations that adjust details like grasp offsets and minor trajectory tweaks. Intuitively, primary modes capture coarse, discrete decisions while within-mode residuals encode fine-grained variations. This observation motivates an explicit separation between i) selecting a coarse, discrete mode consistently and ii) generating the fine-grained continuous action conditioned on that mode.

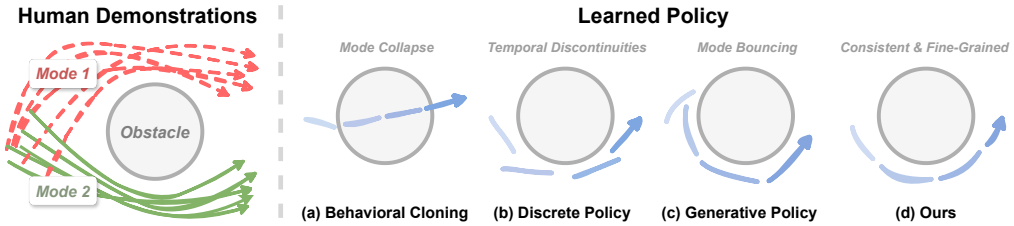


Figure 1: A 2D example illustrating multi-modal expert demonstrations and trajectories predicted by different imitation policies. Behavioral cloning predictions collapse into a single mean. Discrete Policy succeeds but introduces temporal discontinuities. Generative Policy bounces between mode 1 and 2. Our work predicts consistent and fine-grained trajectory.

Motivated by the above, we propose Primary-Fine Decoupling for Action Generation (PF-DAG), a two-stage imitation framework that explicitly separates primary mode selection from continuous action generation. Concretely, PF-DAG first learns a discrete vocabulary of primary modes and a lightweight policy that greedily selects a mode coherently. Then, we introduce a mode conditioned MeanFlow policy, which is a one-step continuous decoder to generate high-fidelity actions conditioned on the selected mode and the current observation. This explicit two-stage decomposition preserves intra-mode variations while reducing mode bouncing by enforcing stable primary choices.

We validate PF-DAG with theoretical and empirical evidence. Among existing methods, single-stage generative policies (Chi et al., 2023; Zhao et al., 2023) are the most direct and competitive end-to-end approach for modeling continuous, multi-modal action distributions, so we focus our theoretical comparison on this family. Under realistic mode-variance assumptions we show that the two-stage design attains a no-higher optimal MSE lower bound than single-stage generative baselines, with a strict improvement whenever the inter-mode variance term is positive. Empirically we test PF-DAG across 56 simulation manipulation tasks (including high-DOF dexterous hands and low-DOF grippers) as well as on real world tactile dexterous manipulation. Results show consistent improvements in accuracy, stability, and sample efficiency compared to diffusion and flow-based baselines, and ablations quantify the contribution of key components. Together, these results suggest that explicitly decoupling coarse discrete decisions from fine-grained continuous generation yields practical and statistical advantages for closed-loop robotic imitation.

2 Related Work

2.1 Behavior Cloning

Behavior cloning (BC) casts policy learning as supervised regression on demonstration data (Wang et al., 2017; Torabi et al., 2018; Mandlekar et al., 2021; Hu et al., 2024). In BC, a policy is trained to predict the expert's action for each observed state, yielding a deterministic mapping from states to actions. This approach is highly sample-efficient in practice (*e.g.* for pick-and-place tasks), but it suffers from well-known limitations. In particular, BC policies tend to underfit multi-modal behavior (Mandlekar et al., 2021; Shafiullah et al., 2022; Florence et al., 2022; Chi et al., 2023) and also incur compounding errors at test time (Ross et al., 2011; Ke et al., 2021; Tu et al., 2022; Zhao et al., 2023). To mitigate these issues, recent work has explored more expressive BC models. Implicit BC and energy-based models learn an action-energy landscape per state and solve for actions by optimization (Florence et al., 2022), while mixture-density networks and latent-variable BC attempt to represent multi-modal distributions explicitly (Jang et al., 2022).

2.2 Discrete Policy

Discretizing continuous robot actions is viewed as tokenization: converting a high-frequency, high-dimensional control signal into a sequence of discrete symbols so that standard sequence-modeling methods can be applied. Framing actions as tokens has two immediate benefits for manipulation imitation. First, next-token prediction over a discrete vocabulary represents multi-modal conditional action distributions without collapsing modes into a single mean. Second, sequence models bring powerful context modeling and scalable pretraining recipes from language and vision to control, enabling cross-task and cross-embodiment generalization when token vocabularies are shared or aligned. Recent Vision-Language-Action (VLA) efforts articulate this reframing and its practical

advantages for large, generalist robot policies (Zitkovich et al., 2023; O'Neill et al., 2024; Kim et al., 2024; Zawalski et al., 2024; Wen et al., 2025; Black et al., 2024; Zheng et al., 2024; Zhen et al., 2024; Cheang et al., 2024; Duan et al., 2024; Zhao et al., 2025).

Existing action tokenizers fall into a few broad families. The simplest and most commonly used approach maps each continuous action dimension at each step to one of a fixed set of bins (Brohan et al., 2022; Zitkovich et al., 2023; Kim et al., 2024). Frequency-space methods like FAST (Pertsch et al., 2025) departs from it and instead compresses action chunks using a time-series transform and lightweight quantization. Others use Vector Quantization (VQ) as latent tokenizers. VQ-based tokenizers learn a shared codebook of action atoms and quantize continuous latent representations to nearest codebook entries (Lee et al., 2024; Wang et al., 2025). While effective at capturing multi-modal action distributions, these approaches inherently trade off reconstruction fidelity for discrete simplicity. Our work differs by leveraging tokenization solely for high-level primary mode selection.

2.3 Generative Policy

A large class of imitation methods treat policy generation as a stochastic generative problem by introducing latent variables. In this view, a policy is written as $a = \pi(o, z)$ with z sampled from a learned prior. This formulation naturally represents multi-modal conditional action distributions because sampling different z values yields different valid actions for the same observation. Action Chunking with Transformers (ACT) (Zhao et al., 2023) is a sequence generator with Conditional Variational Autoencoder (CVAE) as backend. Diffusion Policy (DP) (Chi et al., 2023) treat action generation as conditional denoising. Starting from noise, the action is iteratively refined via a learned score or denoiser conditioned on observation. More recent normalizing-flow policies (Black et al., 2024; Hu et al., 2024; Zhang et al., 2025) provide tractable density estimation and efficient sampling while representing complex, multi-modal action distributions. Although generative policies represent multi-modal distributions, they often face mode bouncing (Chen et al., 2025), inference cost (Li et al., 2024), chunk trade-offs (Zhao et al., 2023). Other hierarchical approaches, such as Hierarchical Diffusion Policy (HDP) (Ma et al., 2024), also use a high-level policy to guide a low-level generator. However, HDP is designed to rely on explicit, task-specific heuristics like contact-point waypoints to define its hierarchy. In contrast, our PF-DAG learns its primary modes end-to-end directly from action-chunk clusters themselves, offering a more general abstraction not tied to predefined heuristics. Thus, we propose to combine the strengths of action tokenization with expressive generative decoders that handle the residual continuous variations. Our PF-DAG decouples the primary discrete mode selection from the fine-grained action generation and reduces mode bouncing while preserving continuous variations.

2.4 Hierarchical and Residual Policies

Our work is also situated within the broader context of hierarchical and residual policies for robot learning (Rana et al., 2023; Cui et al., 2025; Kujanpää et al., 2023; Liang et al., 2024). These approaches commonly decompose the complex control problem into a high-level policy that selects a skill, sub-goal, or context, and a low-level policy that executes control conditioned on the high-level selection (Mete et al., 2024; Feng et al., 2024). For instance, some methods learn residual policies that adapt a base controller (Rana et al., 2023), while others focus on discovering discrete skills from demonstration data or language guidance (Chen et al., 2023; Wan et al., 2024; Tanneberg et al., 2021). While PF-DAG shares this general hierarchical structure, its primary motivation and technical design are distinct. Many hierarchical methods focus on long-horizon planning or unsupervised skill discovery. In contrast, PF-DAG is specifically designed to address the problem of **mode bouncing** inherent in single-stage generative policies when modeling multi-modal action distributions at a fine temporal scale.

3 PF-DAG Formulation and Design

This section first defines the task formulation as a closed-loop action-sequence prediction problem, and then presents the three main components of our approach: i) Observation Feature Extraction, ii) a compact discrete representation learned with a Vector-Quantized VAE (VQ-VAE) (Van Den Oord

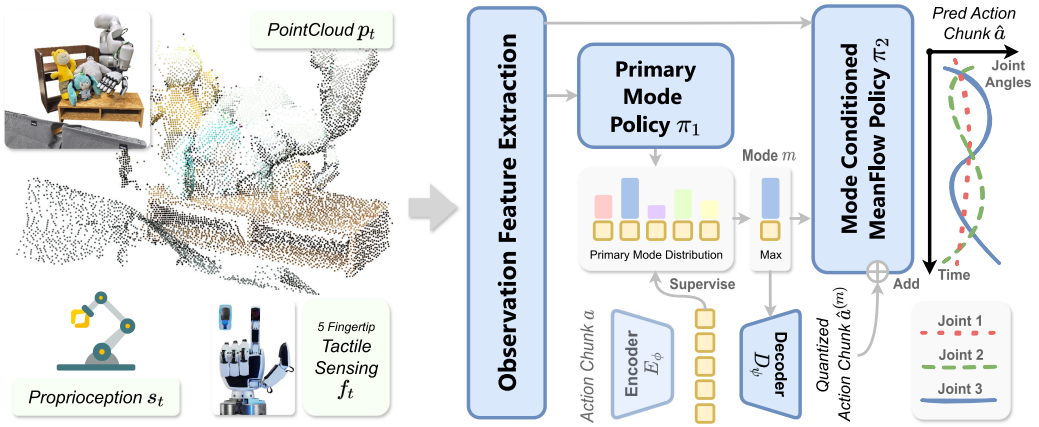


Figure 2: Overview of our PF-DAG framework. The input observation features are extracted via Observation Feature Extraction and then fed to the Primary Mode Policy π_1. The GT action chunks are compressed into discrete primary modes using VQ-VAE and supervise π_1, which are only used in training stage. The Mode Conditioned MeanFlow Policy π_2 takes the selected primary mode m and observation features as input, generating high-fidelity continuous actions.

et al., 2017) and a lightweight Primary Mode Policy that predicts those discrete modes, and iii) a mode conditioned one-step continuous decoder based on MeanFlow (Geng et al., 2025). Finally, we give a theoretical analysis that quantifies why a two-stage, coarse-to-fine decomposition reduces the MSE lower bound compared to single-stage generative models.

3.1 Closed-loop Action Sequence Prediction

Similar to previous work (Chi et al., 2023; Black et al., 2024), we formulate the manipulation task as closed-loop action sequence prediction. Concretely, at time t, the observation is $\mathbf{o}_t = (\mathbf{p}_t, \mathbf{s}_t, \mathbf{f}_t)$, where $\mathbf{p}_t$ denotes a fixed-size point cloud, $\mathbf{s}_t \in \mathbb{R}^{d_s}$ denotes robot proprioception, $\mathbf{f}_t \in \mathbb{R}^{5\times120\times3}$ represents tactile sensing data from the hand's 5 fingertips. For the dimension of $\mathbf{f}_t$, the first dimension 5 corresponds to the 5 individual fingertips, the 120 denotes the number of tactile taxels embedded in each fingertip and the last 3 represents the 3-dimensional force vector. The policy predicts an action chunk $\mathbf{a}_t \in \mathbb{R}^{T_p\times d_a}$ and executes the first $T_a \le T_p$ steps before re-planning.

$$\hat{\mathbf{a}}_t \sim \pi(\mathbf{o}_t), \qquad \text{execute } \hat{\mathbf{a}}_t[0:T_a-1], \text{ then } t \leftarrow t+T_a. \tag{1}$$

This yields a receding-horizon closed-loop control scheme that preserves temporal coherence and allows fast reaction to new observations. Hyperparameters are presented in Appendix A.4

3.2 Observation Feature Extraction

We first extract the shared observation embedding from input observation $\mathbf{o}_t = (\mathbf{p}_t, \mathbf{s}_t, \mathbf{f}_t)$. Following a simple PointNet-style (Qi et al., 2017) pipeline, each point's coordinates are lifted by an MLP, and LayerNorm (Ba et al., 2016) is applied inside that per-point MLP. Per-point features are aggregated by a symmetric max-pooling. The proprioception $\mathbf{s}_t$ and tactile sensing $\mathbf{f}_t$ are passed through respective MLPs and then concatenated with the point-cloud embedding. A final projection MLP fuses the concatenated vector into the shared observation embedding.

3.3 Primary Mode Policy and VQ-VAE

Given the shared observation embedding, the framework first selects a primary mode. This subsection describes how we learn a compact VQ-VAE codebook for action chunks and train a lightweight classifier to predict these primary modes from the observation embedding.

Vector Quantized Variational Autoencoder. Continuous action chunks $\mathbf{a}$ are compressed into a small discrete set of primary modes $m \in \{1,\dots,K\}$ using VQ-VAE. Let the deterministic encoder be $E_\phi : \mathbb{R}^{T_p\times d_a} \to \mathbb{R}^D$ and decoder $D_\psi : \mathbb{R}^D \to \mathbb{R}^{T_p\times d_a}$. Let the codebook be $\mathbf{C} = \{\mathbf{e}_k \in \mathbb{R}^D\}_{k=1}^K$ with codebook size K. We choose K to be small to capture coarse primary action prototypes and make primary policy easy to learn. Given an action chunk $\mathbf{a}$, the encoder

produces $\mathbf{z}_e = E_\phi(\mathbf{a})$ and we quantize it to the nearest codebook vector:

$$k^* = \arg\min_k \|\mathbf{z}_e - \mathbf{e}_k\|_2, \qquad \tilde{\mathbf{z}} = \mathbf{e}_{k^*}, \qquad m := k^*. \tag{2}$$

We define m as the primary mode. Reconstruction is $\hat{\mathbf{a}}^{(m)} = D_\psi(\tilde{\mathbf{z}})$. We train the VQ-VAE with the standard commitment and reconstruction terms:

$$\mathcal{L}_{\text{VQ}}(\mathbf{a}) \;=\; \|\mathbf{a} - D_\psi(\tilde{\mathbf{z}})\|_2^2 \;+\; \|\text{sg}[E_\phi(\mathbf{a})] - \tilde{\mathbf{z}}\|_2^2 \;+\; \beta\|E_\phi(\mathbf{a}) - \text{sg}[\tilde{\mathbf{z}}]\|_2^2, \tag{3}$$

where $\text{sg}[\cdot]$ denotes stop-gradient and β is the commitment weight. The primary policy is a classifier $\pi_1(m \mid \mathbf{o})$ trained to predict the VQ code m from observation $\mathbf{o}$. At test time, we select the discrete mode m for the current chunk by choosing the highest predicted probability from π_1. Both encoder E_ϕ and decoder D_ψ are implemented as compact MLPs.

Primary Mode Policy. The primary policy $\pi_1(m \mid \mathbf{o})$ maps the shared observation embedding to a categorical distribution over the K VQ bins. We implement π_1 as a lightweight MLP classifier. During training π_1 is optimized with a standard cross-entropy objective that matches the encoder-assigned VQ indices. At test time we use greedy mode selection for reliability. The separation of primary-mode selection as an explicit classifier drastically reduces coarse mode bouncing.

3.4 Mode Conditioned MeanFlow Policy

After selecting a primary mode m, we recover a high-quality continuous action chunk that respects the selected mode. To balance generation quality and real-time responsiveness, we use a one-step generative modeling inspired by MeanFlow (Geng et al., 2025). Instead of multi-step denoising iterations, a learned average velocity field predicts the displacement from noise to the desired action in one function evaluation. Let m be the selected discrete mode and $\hat{\mathbf{a}}^{(m)} := D_\psi(\mathbf{e}_m)$ be the VQ-decoder reconstruction of the mode. The role of the one-step generator is to produce a residual $\Delta\mathbf{a}$ conditioned on observation $\mathbf{o}$ and mode m, such that the final action chunk is $\hat{\mathbf{a}} = \hat{\mathbf{a}}^{(m)} + \Delta\mathbf{a}$.

Mode and Observation Conditioned Average Velocity Field. Following MeanFlow (Geng et al., 2025), we implement the residual as an average velocity field $\bar{\mathbf{v}}_\theta\big(\mathbf{z}_r,\, \tau,\, r;\, \mathbf{o}, m\big)$, where $\mathbf{z}_r$ denotes a state on the interpolation path between noise sample and the target action, $\tau \in [0,1]$ is the interpolation start time, and $r \in (0,1]$ is the end time. The MeanFlow field is trained to match the ground-truth average velocity over arbitrary intervals $[\tau, r]$, which is written as

$$\bar{\mathbf{v}}^*(\mathbf{z}_r, \tau, r) = \text{sg}\Big(\tfrac{d\mathbf{z}_r}{dr} - (r-\tau)\Big(\tfrac{d\mathbf{z}_r}{dr}\tfrac{\partial \bar{\mathbf{v}}_\theta}{\partial \mathbf{z}} + \tfrac{\partial \bar{\mathbf{v}}_\theta}{\partial r}\Big)\Big). \tag{4}$$

The $\frac{d\mathbf{z}_r}{dr}$ is the instantaneous velocity of $\mathbf{z}_r$ at time r. $\frac{\partial \bar{\mathbf{v}}_\theta}{\partial \mathbf{z}}$ describes how the average velocity responds to perturbations in the residual draft, and $\frac{\partial \bar{\mathbf{v}}_\theta}{\partial r}$ captures how it evolves as the interpolation approaches the target residual. We train $\bar{\mathbf{v}}_\theta$ with squared-error objective that supervises the predicted average velocity. More detailed derivations of the formulation are provided in Appendix A.3.

Implementation Details. For backbone modeling we use a DiT-style transformer backbone (Peebles & Xie, 2023). Each action chunk is represented as a sequence of tokens. The time-related scalars τ and r are expanded via sinusoidal embeddings (Vaswani et al., 2017), which are added to observation embedding, as well as a learnable embedding of the discrete mode m. During training, (τ, r) is sampled from a uniform distribution and $\mathbf{z}_0$ is from standard normal distribution.

3.5 Theoretical analysis

With the two-stage architecture defined, we now provide a concise theoretical analysis that explains why this coarse-to-fine decomposition strictly reduces the minimum achievable MSE compared to single-stage generative predictors. Single-stage generative methods produce actions by sampling a latent code $\mathbf{z} \sim \mathcal{N}(0, I)$ and decoding $\hat{\mathbf{a}}_g = \pi(\mathbf{o}, \mathbf{z})$. Under the squared-error criterion, the best point estimate is the conditional expectation $\hat{\mathbf{a}}_g^*(\mathbf{o}) = \mathbb{E}_{\mathbf{z}}[\pi(\mathbf{o}, \mathbf{z})]$. The resulting expected MSE decomposes into an irreducible data variance term and a model bias:

$$\mathbb{E}_{\mathbf{o},\mathbf{a}}\big[\|\mathbf{a} - \hat{\mathbf{a}}_g^*(\mathbf{o})\|^2\big] = \mathbb{E}_{\mathbf{o}}\big[\text{Var}(\mathbf{a} \mid \mathbf{o})\big] + \mathbb{E}_{\mathbf{o}}\big[\|\mathbb{E}[\mathbf{a} \mid \mathbf{o}] - \hat{\mathbf{a}}_g^*(\mathbf{o})\|^2\big]. \tag{5}$$

When the model is unbiased the second term vanishes and the minimum achievable error equals $\mathbb{E}_{\mathbf{o}}[\text{Var}(\mathbf{a} \mid \mathbf{o})]$.

Method	Adroit			DexArt				MetaWorld		Success
	Hammer	Door	Pen	Laptop	Faucet	Toilet	Bucket	Medium (6)	Hard (5)	
IBC	0.00±0.00	0.00±0.00	0.10±0.01	0.01±0.01	0.07±0.02	0.15±0.01	0.00±0.00	0.11±0.02	0.09±0.03	0.08
BC-H	0.10±0.09	0.07±0.05	0.16±0.03	0.09±0.02	0.13±0.04	0.21±0.02	0.10±0.01	0.15±0.03	0.18±0.05	0.15
DP	0.48±0.17	0.50±0.05	0.25±0.04	0.69±0.04	0.23±0.08	0.58±0.02	0.46±0.01	0.20±0.05	0.19±0.03	0.30
DP3	**1.00±0.00**	0.62±0.04	0.43±0.06	0.83±0.01	0.63±0.02	**0.82±0.04**	0.46±0.02	0.45±0.05	0.35±0.02	0.51
FlowPolicy	**1.00±0.00**	0.58±0.05	0.53±0.12	0.85±0.02	0.42±0.10	0.80±0.05	0.39±0.06	0.47±0.07	0.37±0.07	0.51
PF-DAG (Ours)	**1.00±0.00**	**0.65±0.03**	**0.65±0.01**	**0.90±0.02**	**0.72±0.05**	**0.82±0.02**	**0.47±0.02**	**0.68±0.04**	**0.72±0.03**	**0.72**

Table 1: Quantitative comparison of PF-DAG against state-of-the-art baselines on 18 tasks from three simulation benchmarks.

In our two-stage scheme the primary stage selects a discrete mode $\hat{m}(\mathbf{o})$ and the second stage outputs $\hat{\mathbf{a}}(\mathbf{o}, m, \mathbf{z}) = \pi_2(\mathbf{o}, m, \mathbf{z})$. For any fixed $(\mathbf{o}, m)$, the optimal MSE predictor collapses the stochasticity in $\mathbf{z}$ to the conditional expectation $\hat{\mathbf{a}}^*(\mathbf{o}, m) = \mathbb{E}_{\mathbf{z}}[\pi_2(\mathbf{o}, m, \mathbf{z})]$, yielding the irreducible residual $\mathbb{E}_{\mathbf{o},m}[\mathrm{Var}(\mathbf{a} \mid \mathbf{o}, m)]$ when the model is unbiased. By the law of total variance,

$$\mathbb{E}_{\mathbf{o},m}\big[\mathrm{Var}(\mathbf{a} \mid \mathbf{o}, m)\big] = \mathbb{E}_{\mathbf{o}}\big[\mathrm{Var}(\mathbf{a} \mid \mathbf{o})\big] - \mathbb{E}_{\mathbf{o}}\big[\mathrm{Var}_{m|\mathbf{o}}\big(\mathbb{E}[\mathbf{a} \mid \mathbf{o}, m]\big)\big], \tag{6}$$

which is no greater than $\mathbb{E}_{\mathbf{o}}[\mathrm{Var}(\mathbf{a} \mid \mathbf{o})]$, and is strictly smaller whenever $\mathrm{Var}_{m|\mathbf{o}}\big(\mathbb{E}[\mathbf{a} \mid \mathbf{o}, m]\big) > 0$. Intuitively, discretizing into primary modes removes inter-mode variance from the residual error, lowering the MSE bound compared to single-stage latent samplers.

4 Experiments

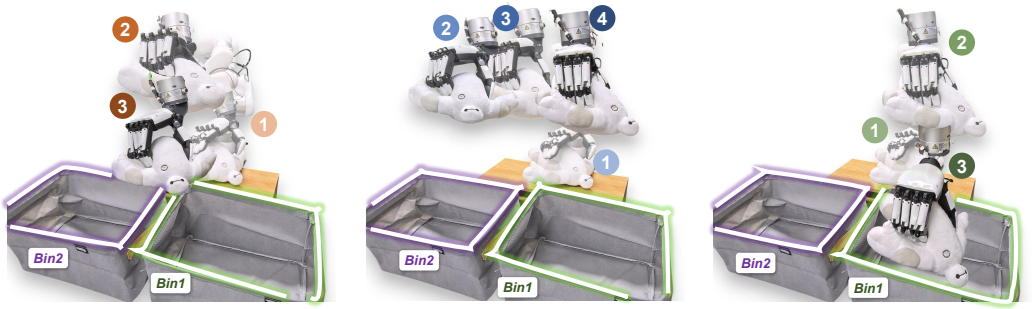
4.1 Simulation Evaluation

Benchmarks and Datasets. We evaluate our method on manipulation benchmarks that cover a broad range of control domains. We use Adroit (Rajeswaran et al., 2017), DexArt (Bao et al., 2023) and MetaWorld (Yu et al., 2020) as our simulation benchmarks. These are implemented on physics engines like MuJoCo (Todorov et al., 2012) and IsaacGym (Makoviychuk et al., 2021). For fair comparison we adopt the same task splits and data collection pipelines as in prior work (Ze et al., 2024): Adroit tasks with high-dimensional Shadow hand and MetaWorld with low-dimensional gripper are trained with 10 expert demos per task, while DexArt with Allegro hand uses 90 expert demos. Demonstrations are collected using scripted policies for MetaWorld tasks, and RL-trained expert agents (Wang et al., 2022; Schulman et al., 2017) for Adroit and DexArt. Each experiment is run with three random seeds. For each seed we evaluate the policy for 20 episodes every 200 training epochs and then compute the average of the top-5 highest success rates (Ze et al., 2024). The final metric is the mean and standard deviation across the three seeds.

Experiment Setup. All networks are optimized with AdamW (Loshchilov & Hutter, 2017). We apply a short linear warmup followed by cosine decay for the learning rate. Training proceeds in stages: first we pretrain the VQ-VAE to learn compact primary prototypes; then we freeze the codebook and jointly train the Primary Mode Policy π_1 (cross-entropy to the VQ indices) and the mode-conditioned MeanFlow generator $\bar{v}_\theta$ (squared-error supervision on sampled (τ, r) intervals). At inference we set $(\tau, r) = (0, 1)$ for one-step continuous action chunk generation.

Baselines. We compare against the following representative baselines. Implicit Behavioral Cloning (IBC) (Florence et al., 2022) serves as a representative implicit BC method. BC-H (Foster et al., 2024) represents non-generative approaches for mitigating mode instability. Diffusion Policy (DP) (Chi et al., 2023) pioneers the original formulation of image-conditioned diffusion-based policies. While 3D Diffusion Policy (DP3) (Ze et al., 2024) represents a recent advancement in 3D-point-cloud conditioned diffusion-based policies, Flow Policy (FP) (Zhang et al., 2025) falls into the category of normalizing-flow-based policy variants. These baselines provide a spectrum from energy-based model to expressive generative policies.

Key Findings. Across Adroit, DexArt and MetaWorld, our method substantially outperforms diffusion and other baselines. Table 1 highlights our work performance on 18 core tasks, while comprehensive results across all 56 tasks are detailed in Appendix A.2. Beyond that, our two-stage design preserves primary-mode consistency even when action chunks are short, which approaches closed-loop and highly reactive operation. Meanwhile, our primary-mode tiny MLP and the one-step generator together yield fast generation while maintaining high success rates, as discussed in Ablations section. These findings indicate that explicitly decoupling coarse discrete mode selection from continuous intra-mode variation yields both statistical and practical benefits.



(a) Mode Collapse (b) Mode Bouncing (c) Ours

Figure 3: Visual comparison of failure modes in baselines versus PF-DAG. **Mode Collapse** outputs "average" actions, while **Mode Bouncing** randomly switches between consecutive time steps.

Task Description	Task Parameters					Success Rate		
	# Variations	# Demos	End Effector	Tactile	Action Dim	Vanilla BC	DP3	**PF-DAG (Ours)**
Pick Cube	3	50	Gripper	✗	$7+1$	0.20	0.60	**0.70**
Place Baymax	1	10	Gripper	✗	$7+1$	0.40	0.85	**0.90**
Wipe Table	5	50	XHand	✓	$7+12$	0.00	0.55	**0.70**
Place Toy Into Bin	4	50	XHand	✓	$7+12$	0.00	0.60	**0.80**

Table 2: Quantitative comparison of success rates of different methods on real world manipulation tasks. The table presents key task parameters alongside the performance of each method.

4.2 Real World Evaluation

Hardware. We evaluate our method on two single-arm hardware configurations commonly used in manipulation research: a) an UFACTORY xArm manipulator [1] equipped with a two-finger parallel gripper, and b) an xArm paired with ROBOTERA XHand [2] for dexterous manipulation. For visual sensing we use a third-person Intel RealSense L515 LiDAR camera that provides aligned color and depth frames. For the *xArm + gripper* setup we additionally use a low-cost 3D-printed demonstration arm from GELLO (Wu et al., 2024) as teleoperation device. For the *xArm + XHand* setup, human hand motion is captured from a Meta Quest 3 headset and retargeted to the XHand. All computation runs on a single workstation equipped with an NVIDIA RTX 4090 laptop GPU. The robot and sensors are controlled over a local area network.

Teleoperation. We collect demonstrations with two teleoperation pipelines. *xArm + gripper* demonstrations are collected using the GELLO framework (Wu et al., 2024), a low-cost and intuitive teleoperation system that allows operators to demonstrate end-effector motions with a separate low-cost manipulator. *xArm + XHand* demonstrations are recorded by capturing human hand kinematics via a Meta Quest 3 headset. The recorded wrist 6-DoF pose is mapped to the xArm end-effector via Inverse Kinematics (IK), finger joint values are retargeted to the XHand via AnyTeleop (Qin et al., 2023).

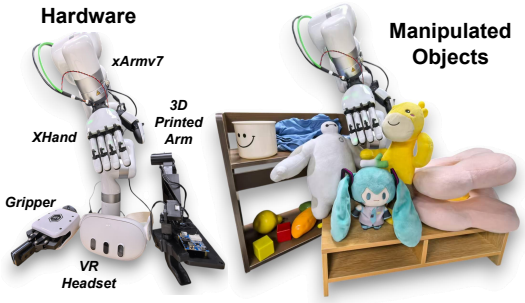


Figure 4: Hardware and manipulated objects used in real world experiments.

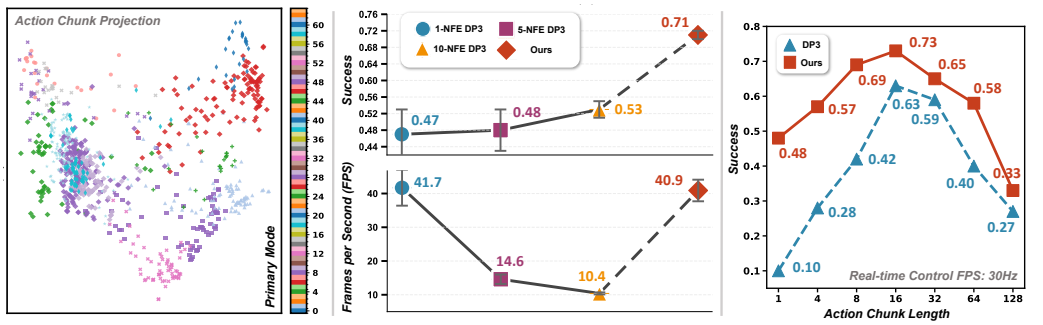
Observation and Action Spaces. Visual input is the RGB-D stream from the RealSense L515. Frames are backprojected to form a colored point cloud. We convert each frame into a fixed-size point cloud by applying Farthest Point Sampling. Proprioceptive observations include the xArm joint angles. When the XHand is integrated, the observation space is extended to include the XHand's joint angles as additional dimensions. For the XHand configuration we additionally log tactile readings from fingertip sensors. All observations are normalized using the statistics computed on the training split. The policy outputs actions directly in joint space for both setups. We operate in absolute joint position control.

Baselines. We compare our method to two baselines. **Vanilla BC** processes observations through the same Observation Feature Extraction pipeline used by our method, and a 3-layer MLP is trained

[1] https://www.ufactory.cc [2] https://www.robotera.com

Ablations				Benchmarks			
w. PM	w. MF	Token.	# Modes	Adroit (3)	DexArt (4)	MetaWorld (11)	Weighted Success
✓	✓	VQ-VAE	64	**0.77**±0.03	**0.72**±0.04	**0.70**±0.02	**0.72**
✓	✗	VQ-VAE	64	0.02±0.01	0.00±0.00	0.02±0.01	0.01
✗	✓	VQ-VAE	64	0.62±0.02	0.65±0.01	0.51±0.03	0.56
✓	✓	VQ-VAE	8	0.72±0.03	0.70±0.02	0.55±0.05	0.61
✓	✓	VQ-VAE	1024	0.66±0.02	0.68±0.03	0.52±0.06	0.58
✓	✓	K-means	64	0.76±0.05	0.70±0.01	0.69±0.02	0.70

Table 3: Ablation study on the impact of PF-DAG's key components and hyperparameters. **w. PM** denotes whether the primary mode policy is included. **w. MF** indicates whether the mode-conditioned MeanFlow policy is included. **Token.** means action tokenization method. **# Modes** represents the number of discrete primary modes.



(a) Primary Mode Clustering (b) Speed-Accuracy Trade-off (c) Action-Chunk Length *vs.* Reactivity

Figure 5: Illustration of critical properties of PF-DAG. (a) Action chunks are projected to 2D via PCA, colored by their assigned primary mode. (b) PF-DAG's one-step MeanFlow decoder achieves FPS comparable to 1-NFE DP3 while maintaining significantly higher success. (c) PF-DAG preserves high success even with short chunks by avoiding primary mode bouncing.

to regress actions in a standard behavior cloning setup (Levine et al., 2016). **DP3** (Ze et al., 2024) is a diffusion-based generative policy operating on 3D point-cloud-conditioned actions. At inference DP3 employs DDIM (Song et al., 2020) denoising to obtain actions. Both baselines are trained on the identical demonstration sets and evaluated under the same closed-loop control as our method.

Tasks. We evaluate on tasks spanning low-DOF gripper control and high-DOF tactile dexterous manipulation. Low-DOF examples include a *Pick Cube* task (gripper picks a cube from randomized table locations) and a *Place Baymax* task (place a toy "Baymax" from table into a cabinet). High-DOF experiments use a 12-DOF dexterous hand equipped with tactile sensing and include contact-rich tasks such as *Wipe Table* (multiple possible wiping contact points) and *Place Toy Into Bin* (multiple candidate toy-boxes yielding multi-modal valid outcomes). For task like *Place Baymax* we exploit a pretraining to fine-tuning regime. Models pretrained on a source task require substantially fewer target-task demonstrations to reach competitive performance.

Result Analysis. Our method consistently outperforms both baselines in success rate across low-DOF and high-DOF/tactile tasks (see Table 2). Qualitatively, Figure 3 visualizes common failure modes of baselines, while our policy commits to coherent, single-mode rollouts when appropriate and preserves intra-mode variations elsewhere. Typical failure cases for our method occur at out-of-distribution object placements or when tactile sensing is intermittently noisy. These failures are rare and amenable to mitigation via modest additional demonstrations or data augmentation.

4.3 Ablations

Primary Mode and MeanFlow Ablation. We ablate the two core components of our pipeline to establish their individual importance. First is to remove the Mode-conditioned MeanFlow Policy (MF) so that the system simply uses the Primary Mode Policy (PM)'s predicted VQ code and decodes it via the VQ-VAE reconstruction as the final action. Second is to remove the PM so that MF attempts to predict actions without being conditioned on a discrete mode. Results are reported in Table 3. Removing MF collapses performance almost completely, showing that a raw VQ reconstruction is insufficient as the final action when the number of modes is limited. The quantization error produces large reconstruction distortions that destroy task success. Conversely, removing PM yields a 0.16

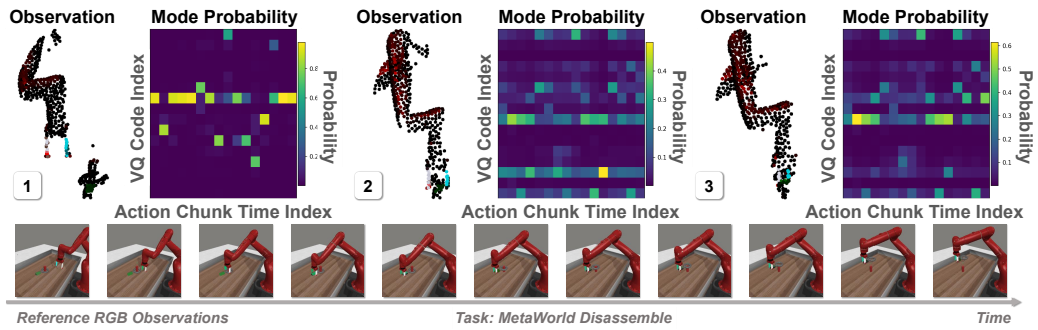


Figure 6: Visualization of the mode probability distribution predicted by the Primary Mode Policy π_1 at three selected frames. The vertical axis of the heatmap represents the mode index.

absolute drop in success, which demonstrates that an explicit primary-mode selection substantially eases the downstream continuous generation problem and prevents coarse-mode bouncing.

Mode Capacity and Tokenization. We study how the number of discrete primary modes K and the choice of tokenization affect the PM learning and final task performance. We vary $K \in \{8, 64, 1024\}$ and compare VQ-VAE with a k-means tokenization baseline. Results appear in Table 3. A small K helps the PM to learn, but it risks underfitting when the task's action-chunk distribution is complex. Large K increases expressivity but makes the PM hard to learn. In our domains the trade-off is modest and $K = 64$ achieves a good balance across tasks. Different tokenizers produce similar final success rates, suggesting the approach is robust to discretization methods. Our method mainly needs a reasonable set of coarse modes rather than a specific quantizer. To further illustrate mode structure we project action-chunks into 2D (PCA) and color by assigned mode. The visualization shows clear coarse-mode clusters on the action manifold, as visualized in Figure 5 (a). We also visualize the policy's outputs at different timesteps (see Figure 6). For a more detailed qualitative analysis of the primary mode policy's behavior, see Appendix A.1.

Ablation on Meanflow. This ablation aims to evaluate the performance difference between MeanFlow and Conditional Flow Matching (CFM) (Lipman et al., 2022), where CFM is tested under different Ordinary Differential Equation (ODE) numerical integration methods. Though CFM theoretically defines a constant velocity field when mapping from noise to target distribution, the parameterized neural network introduces nonlinearity in numerical computations. This makes the exploration of diverse ODE integrators non-trivial. For Runge-Kutta integration, we adopt the Dormand-Prince 5 method, a widely used choice for adaptive-stepsize ODE solving. As shown in Table 4, varying ODE numerical integrators yields negligible performance improvements for CFM. In contrast, replacing CFM with MeanFlow results in a performance gain.

Variations	ODE Solver	Success
CFM	1-NFE Euler	$0.69_{\pm 0.03}$
CFM	10-NFE Euler	$0.69_{\pm 0.02}$
CFM	Runge-Kutta	$0.68_{\pm 0.01}$
Meanflow	1-NFE	$\mathbf{0.72}_{\pm \mathbf{0.02}}$

Table 4: Comparison of success for MeanFlow (MF) and Conditional Flow Matching (CFM), varying the ODE solver and NFE.

Speed–Accuracy Trade-off. We examine how the number of function evaluations (NFE) during inference affects both inference speed and success. We compare our one-step MeanFlow decoder to DP3 at different NFE settings, plots are in Figure 5 (b). Our one-step generator achieves inference speed comparable to DP3 with 1-NFE while delivering substantially higher success. More generally, we observe that within the tested range the total NFE has a surprisingly small influence on success, which suggests that for these simulated tasks the NFE is not the dominant bottleneck. We hypothesize this limited sensitivity is due to the tasks' tolerance to small action perturbations in simulation.

Action-chunk Length and Reactivity. All experiments here are conducted on real settings described in the Real World Evaluation section. We sweep action-chunk length and measure success. Shorter chunks make the controller more open-loop reactive and therefore better able to respond to unexpected environment changes. However, short chunks also tend to increase trajectory jitter and occasional stoppages. Our method maintains relatively high success even at short chunk lengths, showing the two-stage design preserves primary-mode consistency while allowing rapid reactivity. Results appear in Figure 5 (c).

4.4 Failure Cases and Limitations

While PF-DAG demonstrates strong performance across a wide range of manipulation benchmarks, it has two notable limitations. First, because primary-mode selection operates on discretized action chunks, the method exhibits reduced temporal granularity in very high-dynamics, low-latency tasks. Second, the discrete codebook introduces a trade-off between expressivity and learnability. Larger K improves representational capacity but makes primary-policy learning harder, while smaller K constrains diversity. We address this in the paper via targeted ablations and validation sweeps. Promising directions to reduce per-task tuning include shared or meta-learned codebooks, end-to-end distillation, and multi-task pretraining to improve generalization and reduce pipeline overhead.

5 Conclusion

In this work we present PF-DAG, a two-stage imitation learning framework that decouples primary mode selection from fine-grained action generation. PF-DAG first uses a VQ-VAE to tokenize action chunks into discrete modes. A lightweight primary policy is then trained to predict these modes from observations, allowing for stable and consistent coarse mode selection. The framework then employs a mode conditioned MeanFlow policy to produce high-fidelity continuous actions conditioned on the selected mode. We prove that, under realistic variance assumptions, PF-DAG attains a strictly lower MSE bound than comparable single-stage generative policies. Empirically, PF-DAG outperforms state-of-the-art baselines on 56 simulated tasks and on real world tactile dexterous manipulation. Future work will extend PF-DAG to long-horizon hierarchical control and investigate uncertainty-aware refiners for improved robustness.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Chen Bao, Helin Xu, Yuzhe Qin, and Xiaolong Wang. Dexart: Benchmarking generalizable dexterous manipulation with articulated objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21190–21200, 2023.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. $\pi 0$: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, et al. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024.

Lili Chen, Shikhar Bahl, and Deepak Pathak. Playfusion: Skill acquisition via diffusion from language-annotated play. In *Proceedings of the Conference on Robot Learning (CoRL)*, pp. 2012–2029. PMLR, 2023.

Zhuoqun Chen, Xiu Yuan, Tongzhou Mu, and Hao Su. Responsive noise-relaying diffusion policy: Responsive and efficient visuomotor control. *arXiv preprint arXiv:2502.12724*, 2025.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, pp. 02783649241273668, 2023.

Te Cui, Meiling Wang, Guangyan Chen, Xinran Jiang, and Yufeng Yue. Hierarchical autoregressive modeling with multi-scale refinement for robot policy learning. *IEEE Robotics and Automation Letters*, 2025.

Jiafei Duan, Wilbert Pumacay, Nishanth Kumar, Yi Ru Wang, Shulin Tian, Wentao Yuan, Ranjay Krishna, Dieter Fox, Ajay Mandlekar, and Yijie Guo. Aha: A vision-language-model for detecting and reasoning over failures in robotic manipulation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024.

Ruoxuan Feng, Di Hu, Wenke Ma, and Xuelong Li. Play to the score: Stage-guided dynamic multisensory fusion for robotic manipulation. *arXiv preprint arXiv:2408.01366*, 2024.

Pete Florence, Corey Lynch, Andy Zeng, Oscar A Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *Proceedings of the Conference on Robot Learning (CoRL)*, pp. 158–168, 2022.

Dylan J Foster, Adam Block, and Dipendra Misra. Is behavior cloning all you need? understanding horizon in imitation learning. *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 37:120602–120666, 2024.

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.

Xixi Hu, Qiang Liu, Xingchao Liu, and Bo Liu. Adaflow: Imitation learning with variance-adaptive flow-based policies. *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 37:138836–138858, 2024.

Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Proceedings of the Conference on Robot Learning (CoRL)*, pp. 991–1002, 2022.

Liyiming Ke, Jingqiang Wang, Tapomayukh Bhattacharjee, Byron Boots, and Siddhartha Srinivasa. Grasping with chopsticks: Combating covariate shift in model-free imitation learning for fine manipulation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6185–6191, 2021.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Kalle Kujanpää, Joni Pajarinen, and Alexander Ilin. Hierarchical imitation learning with vector quantized models. In *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 17896–17919. PMLR, 2023.

Seungjae Lee, Yibin Wang, Haritheja Etukuru, H Jin Kim, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. Behavior generation with latent actions. *arXiv preprint arXiv:2403.03181*, 2024.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(39):1–40, 2016.

Steven Li, Rickmer Krohn, Tao Chen, Anurag Ajay, Pulkit Agrawal, and Georgia Chalvatzaki. Learning multimodal behaviors from scratch with diffusion policy gradient. *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 37:38456–38479, 2024.

Zhixuan Liang, Yao Mu, Hengbo Ma, Masayoshi Tomizuka, Mingyu Ding, and Ping Luo. Skilldiffuser: Interpretable hierarchical planning via skill abstractions in diffusion-based task execution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16467–16476, 2024.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Xiao Ma, Sumit Patidar, Iain Haughton, and Stephen James. Hierarchical diffusion policy for kinematics-aware multi-task robotic manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18081–18090, 2024.

Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021.

Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.

Atharva Mete, Haotian Xue, Albert Wilcox, Yongxin Chen, and Animesh Garg. Quest: Self-supervised skill abstractions for learning continuous control. *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 37:4062–4089, 2024.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903, 2024.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4195–4205, 2023.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 652–660, 2017.

Yuzhe Qin, Wei Yang, Binghao Huang, Karl Van Wyk, Hao Su, Xiaolong Wang, Yu-Wei Chao, and Dieter Fox. Anyteleop: A general vision-based dexterous robot arm-hand teleoperation system. In *Proceedings of the Robotics: Science and Systems (RSS)*, 2023.

Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv preprint arXiv:1709.10087*, 2017.

Krishan Rana, Ming Xu, Brendan Tidd, Michael Milford, and Niko Sünderhauf. Residual skill policies: Learning an adaptable skill-based action space for reinforcement learning for robotics. In *Proceedings of the Conference on Robot Learning (CoRL)*, pp. 2095–2104. PMLR, 2023.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (ICAIS)*, pp. 627–635, 2011.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Nur Muhammad Shafiullah, Zichen Cui, Ariuntuya Arty Altanzaya, and Lerrel Pinto. Behavior transformers: Cloning k modes with one stone. *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 35:22955–22968, 2022.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

Daniel Tanneberg, Kai Ploeger, Elmar Rueckert, and Jan Peters. Skid raw: Skill discovery from raw trajectories. *IEEE robotics and automation letters*, 6(3):4696–4703, 2021.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5026–5033. IEEE, 2012.

Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 4950–4957, 2018.

Stephen Tu, Alexander Robey, Tingnan Zhang, and Nikolai Matni. On the sample complexity of stability constrained imitation learning. In *Proceedings of the Learning for Dynamics and Control Conference*, pp. 180–191, 2022.

Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 30, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 30, 2017.

Weikang Wan, Yifeng Zhu, Rutav Shah, and Yuke Zhu. Lotus: Continual imitation learning for robot manipulation through unsupervised skill discovery. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 537–544. IEEE, 2024.

Che Wang, Xufang Luo, Keith Ross, and Dongsheng Li. Vrl3: A data-driven framework for visual deep reinforcement learning. *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 35:32974–32988, 2022.

Yating Wang, Haoyi Zhu, Mingyu Liu, Jiange Yang, Hao-Shu Fang, and Tong He. Vq-vla: Improving vision-language-action models via scaling vector-quantized action tokenizers. *arXiv preprint arXiv:2507.01016*, 2025.

Ziyu Wang, Josh S Merel, Scott E Reed, Nando de Freitas, Gregory Wayne, and Nicolas Heess. Robust imitation of diverse behaviors. *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, 30, 2017.

Junjie Wen, Yichen Zhu, Jinming Li, Minjie Zhu, Zhibin Tang, Kun Wu, Zhiyuan Xu, Ning Liu, Ran Cheng, Chaomin Shen, et al. Tinyvla: Towards fast, data-efficient vision-language-action models for robotic manipulation. *IEEE Robotics and Automation Letters*, 2025.

Philipp Wu, Yide Shentu, Zhongke Yi, Xingyu Lin, and Pieter Abbeel. Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 12156–12163. IEEE, 2024.

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 (CoRL)*, pp. 1094–1100. PMLR, 2020.

Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning. *arXiv preprint arXiv:2407.08693*, 2024.

Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. *arXiv preprint arXiv:2403.03954*, 2024.

Qinglun Zhang, Zhen Liu, Haoqiang Fan, Guanghui Liu, Bing Zeng, and Shuaicheng Liu. 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 (AAAI)*, volume 39, pp. 14754–14762, 2025.

Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

Wei Zhao, Pengxiang Ding, Min Zhang, Zhefei Gong, Shuanghao Bai, Han Zhao, and Donglin Wang. Vlas: Vision-language-action model with speech instructions for customized robot manipulation. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025.

Haoyu Zhen, Xiaowen Qiu, Peihao Chen, Jincheng Yang, Xin Yan, Yilun Du, Yining Hong, and Chuang Gan. 3d-vla: A 3d vision-language-action generative world model. *arXiv preprint arXiv:2403.09631*, 2024.

Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé III, Andrey Kolobov, Furong Huang, and Jianwei Yang. Tracevla: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Proceedings of the Conference on Robot Learning (CoRL)*, pp. 2165–2183, 2023.

A Appendix

A.1 Visualizing Primary Mode Distribution

To provide a more intuitive understanding of the Primary Mode Policy π_1, we present a qualitative analysis of its behavior during evaluation episodes. Figure 7 visualizes the policy's outputs at selected keyframes from four representative simulation tasks. For each keyframe, the policy's input point cloud p_t is shown alongside a heatmap representing the predicted probability distribution, over the discrete modes in the VQ codebook. These visualizations reveal that the policy learns a structured and context-aware mapping from inputs to high-level action primitives. As the episode progresses and the observation changes (*e.g.* from approaching an object to making contact) the distribution of predicted modes shifts accordingly, concentrating probability mass on a sparse set of task-relevant modes.

A.2 More Simulation Results

To further verify the generality and stability of PF-DAG in robotic manipulation tasks, this appendix supplements the quantitative experimental results of PF-DAG and mainstream baselines on more tasks under the Adroit, DexArt, and MetaWorld benchmarks. These tasks cover both low-DOF gripper control and high-DOF dexterous hand manipulation, including additional fine-grained operations and complex task categories. All experimental settings are consistent with Section 6 of the main text, including the data collection pipeline, training hyperparameters, and evaluation metrics. The results in Table 5 further confirm that PF-DAG maintains consistent performance advantages over baselines across tasks of varying complexities.

A.3 MeanFlow Derivation

This section provides a detailed derivation of the training objective for our mode-conditioned MeanFlow policy, as mentioned in Section 3.4. The formulation is based on the principles introduced by MeanFlow (Geng et al., 2025), which models the *average velocity* of a generative path rather than the *instantaneous velocity*.

Let the path between a noise sample $z_0 \sim \mathcal{N}(0, I)$ and the target action residual Δa be defined by an interpolation z_r for a time variable $r \in [0, 1]$. The instantaneous velocity at time r is denoted by $v(z_r, r) = \frac{dz_r}{dr}$.

The core concept is to define an **average velocity field** $\bar{v}(z_r, \tau, r; o, m)$ over an arbitrary time interval $[\tau, r]$, where o is the observation and m is the selected primary mode. This field is formally defined as the displacement between two points on the path, divided by the time interval:

$$\bar{v}(z_r, \tau, r; o, m) \triangleq \frac{1}{r - \tau} \int_{\tau}^{r} v(z_s, s; o, m) ds, \tag{7}$$

where s is the integration variable for time. To make this definition amenable to training, we first rewrite it by clearing the denominator:

$$(r - \tau)\bar{v}(z_r, \tau, r; o, m) = \int_{\tau}^{r} v(z_s, s; o, m) ds. \tag{8}$$

Next, we differentiate both sides with respect to the end time r, treating the start time τ as a constant. Applying the product rule to the left-hand side and the Fundamental Theorem of Calculus to the

Table 5: Quantitative comparison of PF-DAG against baselines on more tasks from Adroit, DexArt, and MetaWorld benchmarks.

	Adroit			**DexArt**				**Meta-World (Easy)**	
Alg \ Task	Hammer	Door	Pen	Laptop	Faucet	Toilet	Bucket	Button Press	Button Press Topdown
Diffusion Policy	45±5	37±2	13±2	69±4	23±8	58±2	46±1	99±1	98±1
3D Diffusion Policy	**100±0**	62±4	43±6	83±1	63±2	**82±4**	46±2	**100±0**	**100±0**
PF-DAG (Ours)	**100±0**	**65±3**	**65±3**	**90±2**	**72±5**	**82±2**	**65±3**	**100±0**	**100±0**

	Meta-World (Easy)						
Alg \ Task	Button Press Topdown Wall	Button Press Wall	Coffee Button	Dial Turn	Door Close	Door Lock	Door Open
Diffusion Policy	96±3	97±3	99±1	63±10	**100±0**	86±8	98±3
3D Diffusion Policy	99±2	99±1	**100±0**	**66±1**	**100±0**	**98±2**	99±1
PF-DAG (Ours)	**100±0**	**100±0**	**100±0**	55±10	**100±0**	94±6	**100±0**

	Meta-World (Easy)						
Alg \ Task	Door Unlock	Drawer Close	Drawer Open	Faucet Close	Faucet Open	Handle Press	Handle Pull
Diffusion Policy	98±3	**100±0**	93±3	**100±0**	**100±0**	81±4	27±22
3D Diffusion Policy	**100±0**	**100±0**	**100±0**	**100±0**	**100±0**	**100±0**	53±11
PF-DAG (Ours)	**100±0**	**100±0**	**100±0**	**100±0**	**100±0**	**100±0**	**55±5**

	Meta-World (Easy)						
Alg \ Task	Handle Press Side	Handle Pull Side	Lever Pull	Plate Slide	Plate Slide Back	Plate Slide Back Side	Plate Slide Side
Diffusion Policy	**100±0**	23±17	49±5	83±4	99±0	**100±0**	**100±0**
3D Diffusion Policy	**100±0**	**85±3**	79±8	**100±1**	99±0	**100±0**	**100±0**
PF-DAG (Ours)	**100±0**	77±4	**80±6**	**100±0**	**100±0**	**100±0**	**100±0**

	Meta-World (Easy)					**Meta-World (Medium)**	
Alg \ Task	Reach	Reach Wall	Window Close	Window Open	Peg Unplug Side	Basketball	Bin Picking
Diffusion Policy	18±2	59±7	**100±0**	**100±0**	74±3	85±6	15±4
3D Diffusion Policy	24±1	68±3	**100±0**	**100±0**	**75±5**	**98±2**	**34±30**
PF-DAG (Ours)	**29±5**	**71±3**	**100±0**	**100±0**	74±6	**98±2**	30±15

	Meta-World (Medium)						
Alg \ Task	Box Close	Coffee Pull	Coffee Push	Hammer	Peg Insert Side	Push Wall	Soccer
Diffusion Policy	30±5	34±7	67±4	15±6	34±7	20±3	14±4
3D Diffusion Policy	42±3	87±3	94±3	76±4	69±7	49±8	18±3
PF-DAG (Ours)	**70±5**	**89±5**	**95±3**	**100±0**	**71±1**	**69±2**	**34±3**

	Meta-World (Medium)		**Meta-World (Hard)**				
Alg \ Task	Sweep	Sweep Into	Assembly	Hand Insert	Pick Out of Hole	Pick Place	Push
Diffusion Policy	18±8	10±4	15±1	9±2	0±0	0±0	30±3
3D Diffusion Policy	**96±3**	15±5	**99±1**	14±4	14±9	12±4	51±3
PF-DAG (Ours)	92±5	**48±2**	98±2	**21±4**	**29±5**	**69±2**	**75±2**

	Meta-World (Hard)	**Meta-World (Very Hard)**					**Average**
Alg \ Task	Push Back	Shelf Place	Disassemble	Stick Pull	Stick Push	Pick Place Wall	
Diffusion Policy	0±0	11±3	43±7	11±2	63±3	5±1	55.4
3D Diffusion Policy	0±0	17±10	69±4	27±8	97±4	35±8	72.5
PF-DAG (Ours)	**6±5**	**52±8**	**77±7**	**59±6**	**100±0**	**82±6**	**79.6**

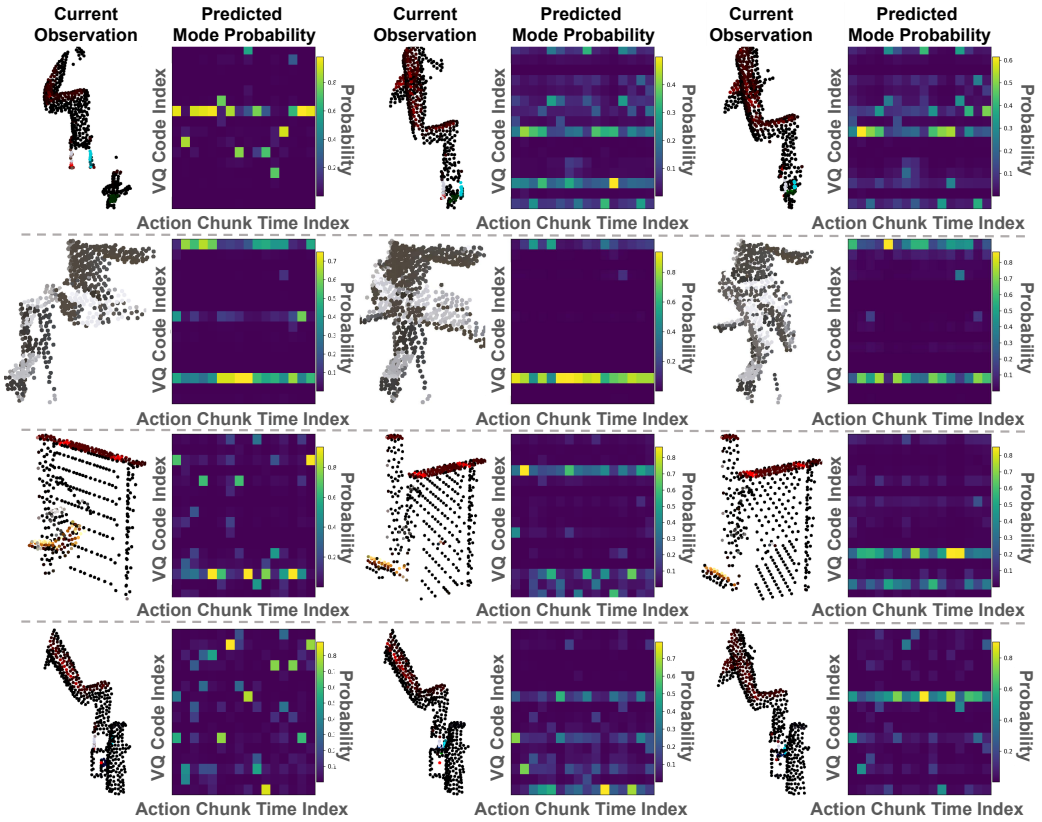


Figure 7: Qualitative visualization of the Primary Mode Policy (π_1) at keyframes from four different simulation tasks. Each row corresponds to a single task episode. Within each row, three keyframes show the point cloud observation (left) and the corresponding predicted probability distribution over the discrete primary modes (right) as a heatmap. The vertical axis of the heatmap represents the mode index. The shifting patterns in the heatmaps demonstrate that the policy learns a dynamic, context-dependent mapping from observation to a belief over high-level actions as the task progresses.

right-hand side yields:

$$\frac{d}{dr}\left[(r-\tau)\bar{v}(z_r,\tau,r)\right] = \frac{d}{dr}\int_\tau^r v(z_s,s)ds, \tag{9}$$

$$\bar{v}(z_r,\tau,r) + (r-\tau)\frac{d}{dr}\bar{v}(z_r,\tau,r) = v(z_r,r). \tag{10}$$

For clarity, we have omitted the conditioning on (o, m) in the last two steps. Rearranging the terms, we arrive at the **MeanFlow Identity**, which establishes a fundamental relationship between the average and instantaneous velocities:

$$\bar{v}(z_r,\tau,r) = v(z_r,r) - (r-\tau)\frac{d}{dr}\bar{v}(z_r,\tau,r). \tag{11}$$

This identity provides a way to define a target for our neural network without computing an integral. To do so, we must first express the total time derivative $\frac{d}{dr}\bar{v}$ in a computable form. Since $\bar{v}$ is a function of (z_r, τ, r), we expand the total derivative using the chain rule:

$$\frac{d}{dr}\bar{v}(z_r,\tau,r) = \frac{\partial\bar{v}}{\partial z}\frac{dz_r}{dr} + \frac{\partial\bar{v}}{\partial\tau}\frac{d\tau}{dr} + \frac{\partial\bar{v}}{\partial r}\frac{dr}{dr}. \tag{12}$$

Given that $\frac{dz_r}{dr} = v(z_r, r)$, $\frac{d\tau}{dr} = 0$ (as τ is independent of r), and $\frac{dr}{dr} = 1$, the expression simplifies to:

$$\frac{d}{dr}\bar{v}(z_r,\tau,r) = v(z_r,r)\frac{\partial\bar{v}}{\partial z} + \frac{\partial\bar{v}}{\partial r}. \tag{13}$$

Substituting this result (13) back into the MeanFlow Identity (11), we obtain an expression for the average velocity that only depends on the instantaneous velocity v and the partial derivatives of $\bar{v}$:

$$\bar{v}(z_r, \tau, r) = v(z_r, r) - (r - \tau)\left(v(z_r, r)\frac{\partial \bar{v}}{\partial z} + \frac{\partial \bar{v}}{\partial r}\right). \tag{14}$$

This equation forms the basis for our training objective. We parameterize the average velocity field with a neural network $\bar{v}_\theta(z_r, \tau, r; o, m)$. The right-hand side of the equation becomes the regression target, where we replace the true partial derivatives of $\bar{v}$ with those of our network $\bar{v}_\theta$. Following standard practice, we apply a stop-gradient operator, $\mathrm{sg}(\cdot)$, to the target to prevent backpropagation through the Jacobian-vector products, which stabilizes training.

The resulting target, $\bar{v}_{tgt}$, is:

$$\bar{v}_{tgt} = v(z_r, r) - (r - \tau)\left(v(z_r, r)\frac{\partial \bar{v}_\theta}{\partial z} + \frac{\partial \bar{v}_\theta}{\partial r}\right). \tag{15}$$

The instantaneous velocity $v(z_r, r)$ is substituted with the conditional velocity (i.e., the ground-truth residual Δa minus the initial noise z_0). The final loss function is the expected squared ℓ_2 error between our network's prediction and this supervised target:

$$\mathcal{L}(\theta) = \mathbb{E}_{\Delta a, z_0, \tau, r} \left\| \bar{v}_\theta(z_r, \tau, r; o, m) - \mathrm{sg}(\bar{v}_{tgt}) \right\|_2^2. \tag{16}$$

This objective allows the network $\bar{v}_\theta$ to learn the average velocity field directly, enabling efficient one-step generation of the action residual Δa at inference time.

A.4 Implementation and Training Hyperparameters

This subsection details the key hyperparameters used for training and implementing our PF-DAG.

Hyperparameter	**Description**	**Value**
Prediction Horizon T_p	The total number of timesteps in a predicted action chunk.	32 / 16
Execution Horizon T_a	The number of timesteps from the chunk executed before re-planning.	16 / 8
Learning Rate	The peak learning rate after the warmup phase.	1e-4
Weight Decay	The weight decay value for the AdamW optimizer.	0.01
Batch Size	The number of samples processed per training step.	128
Codebook Size K	The number of discrete primary modes in the VQ-VAE codebook.	64
Commitment Weight β	The weight of the commitment loss term in the VQ-VAE objective.	0.25
VQ-VAE Latent Dim	The dimensionality of the VQ-VAE latent space.	64

Table 6: Hyperparameters for the PF-DAG framework.

A.5 Ablation Study on Mode Number

We present more results on mode number K, as seen in Table 7.

Ablations # Modes	Benchmarks Adroit (3)	DexArt (4)	MetaWorld (11)	Weighted Success
64	0.77±0.03	**0.72±0.04**	**0.70±0.02**	**0.72**
8	0.72±0.03	0.70±0.02	0.55±0.05	0.61
16	**0.79±0.02**	0.71±0.03	0.68±0.01	0.70
32	0.76±0.03	0.71±0.02	0.67±0.05	0.70
128	0.77±0.01	0.69±0.03	0.67±0.03	0.69
1024	0.66±0.02	0.68±0.03	0.52±0.06	0.58

Table 7: Ablation study on the mode number K of PF-DAG.

A.6 Quantitative Stability Analysis

To quantitatively validate our claim that PF-DAG produces more stable trajectories by reducing mode bouncing, we analyze the **total end-effector jerk** in our real-world experiments (Section 4.2), where stability is critical. Jerk, a standard metric for motion smoothness, is the integral of the squared magnitude of the third derivative of position over the trajectory duration T:

$$\text{Jerk} = \int_0^T \left\| \frac{d^3\mathbf{p}(t)}{dt^3} \right\|^2 dt$$

A lower total jerk indicates a physically smoother, less shaky, and more stable trajectory. We computed this metric for the contact-rich 'Wipe Table' task from our real-world evaluation, comparing PF-DAG against DP3. As shown in Table A.5, PF-DAG achieves significantly lower jerk, confirming it generates smoother, less erratic end-effector movements.

Method	Total Jerk (↓)
DP3	1.25
PF-DAG (Ours)	**0.45**

Table 8: Total end-effector jerk (↓) comparison on the real-world 'Wipe Table' task.

A.7 Rigorous Analysis of the MSE Trade-off

The analysis in the main text assumes an oracle π_1 to illustrate how our architecture decomposes variance. Here, we provide a more rigorous analysis of the practical trade-off, considering errors from our learned π_1.

A.7.1 The Problem with MSE-Optimal Predictors

The central thesis of our paper is that in multi-modal tasks, a predictor that is "optimal" under the Mean Squared Error (MSE) criterion is undesirable. A standard Behavioral Cloning (BC) model that predicts the conditional expectation $\hat{a}^*(o) = \mathbb{E}[a|o]$ is, by definition, the optimal deterministic predictor. Its minimum achievable loss is:

$$L_g^* = \mathbb{E}_o[Var(a|o)]$$

Using the law of total variance, we decompose this loss:

$$L_g^* = \underbrace{\mathbb{E}_{o,m}[Var(a|o,m)]}_{V_{\text{intra}}} + \underbrace{\mathbb{E}_o[Var_{m|o}(\mathbb{E}[a|o,m])]}_{V_{\text{inter}}}$$

- V_{intra}: The **within-mode variance**. This is the fine-grained variation that our π_2 must model.
- V_{inter}: The **inter-mode variance**. This is the variance between the means of the different modes (*e.g.*, the difference between "go left" and "go right").

The MSE-optimal predictor $\mathbb{E}[a|o]$ averages these modes, resulting in V_{inter} as a fundamental component of its error. This is precisely **mode collapse**, which is catastrophic for task success.

A.7.2 The PF-DAG Trade-Off: V_{INTER} vs. E_{CLASSIFY}

Our two-stage model, PF-DAG, makes a "hard" mode selection $\hat{m} = \pi_1(o)$. The final action is the prediction of the second stage, $\hat{a}_{\text{PF-DAG}}(o) = \mathbb{E}_z[\pi_2(o, \hat{m}, z)]$. For this analysis, let's assume a perfect π_2 that correctly predicts the mean of its target mode, *i.e.*, $\mathbb{E}_z[\pi_2(o, k, z)] = \mathbb{E}[a|o, m = k]$, which we denote $\mu_k(o)$. The practical MSE of our model is $L_{\text{PF-DAG}} = \mathbb{E}_{o,a}[||a - \mu_{\hat{m}(o)}(o)||^2]$. We decompose this by conditioning on the true, unobserved mode m:

$$L_{\text{PF-DAG}} = \mathbb{E}_o \left[\sum_m p(m|o) \mathbb{E}_{a|o,m}[||a - \mu_{\hat{m}(o)}(o)||^2] \right]$$

Using the identity $\mathbb{E}[||X - c||^2] = Var(X) + ||\mathbb{E}[X] - c||^2$, where $X = a|o, m$ and $c = \mu_{\hat{m}(o)}(o)$:

$$L_{\text{PF-DAG}} = \mathbb{E}_o \left[\sum_m p(m|o)(Var(a|o, m) + ||\mu_m(o) - \mu_{\hat{m}(o)}(o)||^2) \right]$$

$$L_{\text{PF-DAG}} = \underbrace{\mathbb{E}_{o,m}[Var(a|o, m)]}_{V_{\text{intra}}} + \underbrace{\mathbb{E}_o \left[\sum_m p(m|o)||\mu_m(o) - \mu_{\hat{m}(o)}(o)||^2 \right]}_{E_{\text{classify}}}$$

This reveals the explicit trade-off of our architecture:

- **Single-Stage (BC):** $L_g^* = V_{\text{intra}} + V_{\text{inter}}$
- **PF-DAG (Ours):** $L_{\text{PF-DAG}} = V_{\text{intra}} + E_{\text{classify}}$

PF-DAG is designed to trade V_{inter} (the guaranteed, catastrophic cost of mode collapse) for E_{classify} (the probabilistic cost of misclassification). Our strong empirical task success (Tables 1, 2, 5) supports our hypothesis that V_{inter} is fatal for task execution, while E_{classify} is a non-catastrophic and manageable error. Our framework replaces a guaranteed failure mode with a high-probability success, which is a highly desirable trade-off for robotic imitation.

A.8 Acknowledgments on LLM Usage

We acknowledge the use of a large language model (LLM) for aiding in the writing and polishing of this paper. The LLM is used as a tool to improve the clarity, grammar, and style of certain sections. Its contributions are limited to editorial and linguistic improvements, and it is not used to generate novel ideas, perform research, or formulate the core technical content. All scientific contributions, experimental results, and intellectual content are the original work of the human authors.