PHASEFORMER: MULTIMODAL POLICIES WITHOUT ITERATIVE COMPUTE OR STOCHASTIC INFERENCE

Anonymous authors
Paper under double-blind review

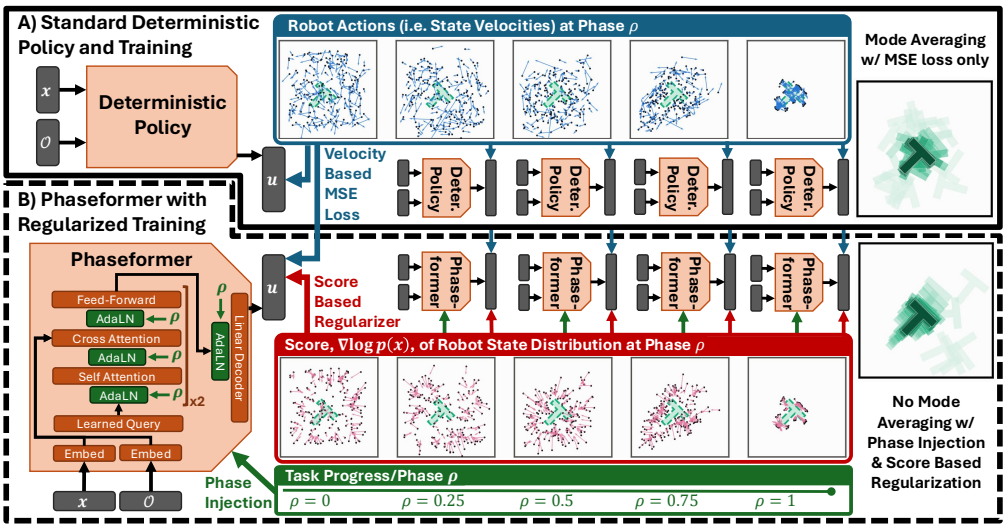

Figure 1: Phaseformer is a deterministic behavior cloning policy that uses Adaptive Layer Norms (AdaLN) to inject task progress (ρ) into the model. A) Deterministic policies trained via velocity-based MSE loss experience catastrophic mode-averaging action prediction problems. B) Inspired by flow matching, our proposed solution, ***Phaseformer***, uses no stochastic sampling or iterative computation. Instead it leverages the stochasticity in the training data via task progress conditioning and a score-based regularizer to avoid mode averaging at key decision points and avoid stalling in robot manipulation tasks.

ABSTRACT

Deterministic behavior cloning policies struggle to capture multimodal robotic action distributions, suffering from mode averaging. Accepted solutions transform stochastic samples into robot actions, often via iterative computation (e.g. diffusion and flow matching). Flow matching backbones themselves are deterministic however, leading us to ask: why do flow matching models succeed at capturing flows between endpoint distributions while deterministic policies fail in similar situations? In this work, we prove that flow matching is a special case of behavior cloning, show that conditioning a deterministic policy on task progress lets it reproduce multimodal demonstrations whose start states vary, and characterize a score-based regularizer that pulls the policy toward the locally dominant demonstrated velocity where modes overlap. We demonstrate this empirically on multimodal robot navigation and in-plane manipulation tasks. Our findings indicate that lightweight, single-step deterministic policies can solve multimodal tasks while providing a Pareto-optimal trade-off for ultra-low latency, CPU based control.

1 INTRODUCTION

The evolution of behavior cloning policies in robotics has been defined by the struggle to overcome mode averaging; the problem of policies averaging actions from multimodal demonstration distributions. These average actions cause collisions, stalling, and poor performance (Florence et al., 2022).

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

The converged on solutions have driven the field away from fast, deterministic policies and toward heavily parameterized, iterative sampling methods (Chi et al., 2024). These solutions follow similar approaches, policies that learn stochastic latent representations and transform them into robot actions (Zhao et al., 2023; Chi et al., 2024; Black et al., 2024).

Recent works have focused on reducing computational burden in order to achieve lower inference time needed for fast, feedback-based robot control. Pan et al. (2025) evaluate the root success of generative policies and suggest that their advantage stems primarily from supervised iterative computation and stochasticity, rather than an inherent ability to capture multimodality. To validate this, Pan et al. (2025) introduced a minimal iterative policy (MIP), a two-step regression model. Concurrently, streaming flow policy (Jiang et al., 2025) proposed treating action trajectories directly as continuous flows, training a temporally conditioned action generation model which denoises samples from a tight Gaussian centered on the previous action. Another latent-variable policy that has shown competitive multimodal performance is the action chunking transformer (ACT), which trains a conditional variational autoencoder (CVAE) to generate action chunks conditioned on observations and decodes the prior mean of its latent at inference (Zhao et al., 2023).

Like streaming flow policy, we examine unrolling robot trajectories in real-time using temporal conditioning. Unlike streaming flow policy, which operates in an abstract action space and requires a stochastic Gaussian initialization to resolve multi-modality, we formulate trajectories as a flow of state derivatives which allows us to evaluate the spatial gradients (score) of the expert demonstrations directly at training time. Our policy, ***Phaseformer***, uses *phase injection* and *score-based regularization*, which pulls the policy toward the locally dominant branch where demonstrated modes overlap, to overcome mode averaging without relying on iterative compute or stochastic sampling at inference. Phaseformer is a lightweight, phase-injected transformer-based policy capable of performing multimodal in-plane robot manipulation. A lightweight, phase-injected multi-layer perceptron (MLP) alternative capable of multimodal robot navigation is also presented. These policies offer Pareto-optimal, low-latency alternatives for real-time feedback-based control of robotic systems.

2 PRELIMINARIES

Behavior Cloning policies approximate the demonstrated expert policy $\pi^*(\mathbf{u}|\mathbf{x})$ with a learned policy π_ϕ by supervising the conditional action distribution during offline training (Pomerleau, 1988). The policy weights ϕ^* minimize the expected loss:

$$\phi^* = \arg\min_{\phi} \mathbb{E}_{(\mathbf{x},\mathbf{u})\sim\mathcal{D}}[\mathcal{L}(\pi_\phi(\mathbf{x}), \mathbf{u})] \tag{1}$$

over the demonstration dataset of state-action tuples, $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{u}_i)\}_{i=1}^N$, where $\mathcal{L}(\pi_\phi(\mathbf{x}), \mathbf{u}) = \|\pi_\phi(\mathbf{x}) - \mathbf{u}\|^2$ for continuous actions and $\mathcal{L} = -\log \pi_\phi(\mathbf{u} \,|\, \mathbf{x})$ for discrete actions. With the squared loss a deterministic π_ϕ learns only the conditional mean $\mathbb{E}[\mathbf{u} \,|\, \mathbf{x}]$; Sec. 3 characterizes when rolling out that mean does and does not reproduce the demonstrations. Often $\mathbf{u} = \dot{\mathbf{x}}$ (Urain et al., 2024).

Flow Matching is a form of generative modeling which transforms samples between endpoint distributions $p_0(\mathbf{x})$ and $p_1(\mathbf{x})$ using continuous flows approximated by a learned phase-dependent velocity field $d\mathbf{x}/d\rho = v_\psi(\mathbf{x}, \rho)$ (Lipman et al., 2023). Because the true flow field is intractable (Lipman et al., 2023), conditional flow matching (CFM) marginalizes over individual flow fields between start and goal samples.

CFM regresses v_ψ onto the velocities of linear conditional paths between paired samples, $\mathbf{x}_\rho = \rho\mathbf{x}_1 + (1-\rho)\mathbf{x}_0$ with $\rho \sim \mathcal{U}[0,1]$, whose velocity is $u_\rho(\mathbf{x}|\mathbf{x}_1) = d\mathbf{x}_\rho/d\rho = \mathbf{x}_1 - \mathbf{x}_0$. With independently drawn endpoints these paths cross, so the learned average flow is curved and needs many integration steps (Tong et al., 2024; Pooladian et al., 2023). Optimal transport (OT) pairing straightens the flow by minimizing crossing paths on each minibatch (Tong et al., 2024). With a Gaussian prior $p_0 = \mathcal{N}(0, I)$, data distribution q, and $\gamma^\star$ the W_2-optimal coupling of the two, the OT-CFM loss is

$$\mathcal{L}_{\text{CFM}}^{\text{OT}}(\psi) = \mathbb{E}_{\rho\sim\mathcal{U}[0,1],\,(\mathbf{x}_0,\mathbf{x}_1)\sim\gamma^\star} \|v_\psi(\mathbf{x}_\rho, \rho) - (\mathbf{x}_1 - \mathbf{x}_0)\|^2, \tag{2}$$

and drawing $\mathbf{x}_0 \sim p_0$ and $\mathbf{x}_1 \sim q$ independently gives standard CFM. The continuous flow matching ODE is often discretized using the forward Euler method into the iterative update $\mathbf{x}_{\rho_{i+1}} = \mathbf{x}_{\rho_i} + \Delta\rho\, v_\psi(\mathbf{x}_{\rho_i}, \rho_i)$ to approximate the final trajectory over N steps (Lipman et al., 2024).

3 Flow Matching as a Special Case of Behavior Cloning

We focus on the task of predicting robot actions that are state derivatives, $\mathbf{u} = \dot{\mathbf{x}}$, typical in robot control (Urain et al., 2024). Let our learned flow model be $\mathbf{u} = d\mathbf{x}/d\rho = v_\psi(\mathbf{x}, \rho)$ and our behavior cloned policy $\mathbf{u} = \pi_\phi(\mathbf{x})$. From the demonstrations we build a phase-labeled dataset

$$\mathcal{D}_\rho = \left\{ \left(\rho_j, \mathbf{x}_{\rho j}, \frac{d\mathbf{x}_{\rho j}}{d\rho} \right) \right\}_{j=1}^{M} \tag{3}$$

with $\rho_j \sim \mathcal{U}[0,1]$ and boundary conditions $\mathbf{x}_{0j} \sim P_S(\mathbf{x}_0)$, $\mathbf{x}_{1j} \sim P_G(\mathbf{x}_T)$. The standard approach in policy learning is to use a flow model as a generative policy where $P_S(\mathbf{x}_0) = \mathcal{N}(\mathbf{0}, \mathbf{I})$ and $P_G(\mathbf{x}_T)$ is the distribution of robot actions. Instead, we replace the policy, $\pi_\phi(\mathbf{x})$, with the flow model, $v_\psi(\mathbf{x}, \rho)$, creating a phase-conditioned policy, written $\pi_\phi(\mathbf{x}, \rho)$ from here on, that directly models the flow of the robot states in the demonstrations. We can do this because flow matching is itself a special case of imitation learning. To illustrate the similarity, consider Fig. 2, which shows a temporally evolving distribution of robot states $\mathbf{x} = [x\ y\ \theta]$ on the SE(2) manifold for a navigation scenario (B, Sec. 5). This distribution splits from a single start distribution into two paths, arriving at two distinct goal distributions. Visually, this evolution is ambiguous: it is impossible to distinguish whether this represents a series of intermediate distributions in a denoising flow, or a time-evolving distribution of robot states from demonstrated trajectories. In both frameworks, the states at any given phase/time form a distribution where the most optimal, or most likely, states possess the highest density.

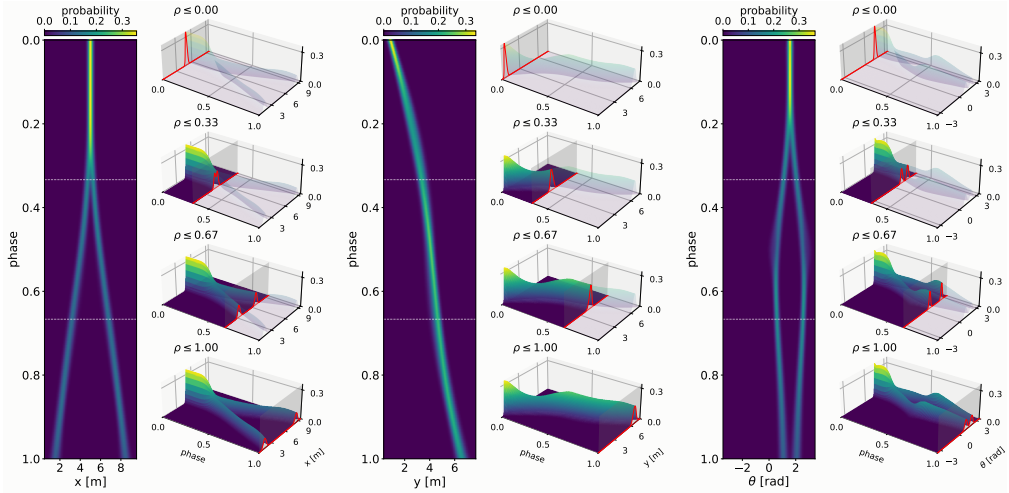

Figure 2: Evolution of the Marginal Robot State Distributions (x, y, and θ) over phase for the single-start, dual-goal, dual-path demonstrations (scenario B). Snapshots at $\rho = 0.0, 0.33, 0.67, 1.0$ show the evolution between the endpoint distributions.

Despite this structural similarity, standard approaches to behavior cloning and flow matching diverge in execution. Flow matching conditions a deterministic network on an explicit phase variable and regresses onto the average velocity at each state and phase. Velocity-based behavior cloned policies typically attempt the same task without phase. On demonstrations that visit a state at different phases or whose branches cross, they average velocities across phases and modes, hence modern policies' dependence on stochastic latent representations or iterative compute (Chi et al., 2024; Zhao et al., 2023). Flow matching in this context is used as a multistep generative policy that denoises sampled robot actions, rather than being applied directly to predicting the flow of robot states in the demonstrated trajectories. Recognizing that flow matching is phase-conditioned behavior cloning on straight-line demonstrations (Lemma 1) provides a direct path to improve standard imitation learning. We now formally prove this equivalence.

Setting. We drop the observation $\mathcal{O}$ and normalize time, so $\rho \in [0,1]$ and $\mathbf{x}_1 = \mathbf{x}_T$. A demonstration is a random path $(\mathbf{x}_\rho)_{\rho \in [0,1]}$ with $\mathbf{x}_0 \sim P_S$ and $\mathbf{x}_1 \sim P_G$, and p_ρ is the distribution of $\mathbf{x}_\rho$ (Fig. 2). **(A1)** The dynamics are single-integrator in $\mathbb{R}^d$, $\mathbf{u} = \dot{\mathbf{x}}$. **(A2)** P_S and P_G have finite second moments, and P_S has a density. **(A3)** The demonstrations minimize the expected control effort $\mathcal{E} = \mathbb{E} \int_0^1 \frac{1}{2} \|\dot{\mathbf{x}}_\rho\|^2 d\rho$ among all random paths with these endpoint distributions. (A3) is the

zero-noise limit of planning as inference (Levine, 2018) with P_S and P_G imposed exactly, i.e. of the Schrödinger bridge (Mikami, 2004; Léonard, 2012).

With phase as an input, the squared-loss (Gaussian negative log-likelihood) BC objective of Eq. 1, and CFM with a coupling γ of P_S and P_G, are

$$\mathcal{L}_{\mathrm{BC}}(\phi) = \mathbb{E}_{\rho\sim\mathcal{U}[0,1]}\,\mathbb{E}\left\|\pi_\phi(\mathbf{x}_\rho,\rho) - \dot{\mathbf{x}}_\rho\right\|^2, \tag{4}$$

$$\mathcal{L}^{\gamma}_{\mathrm{CFM}}(\phi) = \mathbb{E}_{\rho\sim\mathcal{U}[0,1],\,(\mathbf{x}_0,\mathbf{x}_1)\sim\gamma}\left\|\pi_\phi\big((1-\rho)\mathbf{x}_0 + \rho\mathbf{x}_1,\rho\big) - (\mathbf{x}_1-\mathbf{x}_0)\right\|^2, \tag{5}$$

and the W_2-optimal coupling $\gamma = \gamma^\star$ gives OT-CFM (Eq. 2, with P_S and P_G in place of $\mathcal{N}(0,I)$ and q). Over all phase-conditioned policies, Eq. 4 is minimized by the average demonstrated velocity at each state and phase, $\bar{v}(\mathbf{x},\rho) = \mathbb{E}[\dot{\mathbf{x}}_\rho \mid \mathbf{x}_\rho = \mathbf{x}]$. A phase-free policy $\pi_\phi(\mathbf{x})$ cannot see ρ, so its optimum averages $\bar{v}$ over the phases at which demonstrations visit $\mathbf{x}$, $\bar{v}_{\mathrm{free}}(\mathbf{x}) = \int_0^1 p_\rho(\mathbf{x})\,\bar{v}(\mathbf{x},\rho)\,d\rho \,/\, \int_0^1 p_\rho(\mathbf{x})\,d\rho$ (App. A.1.1). A rollout of a policy π solves $d\mathbf{x}/d\rho = \pi(\mathbf{x},\rho)$, and we call π $L(\rho)$-*Lipschitz* if $\|\pi(\mathbf{x},\rho) - \pi(\mathbf{y},\rho)\| \le L(\rho)\,\|\mathbf{x}-\mathbf{y}\|$ for all $\mathbf{x},\mathbf{y}$ and ρ. Thus $L(\rho)$ bounds how sharply the commanded velocity can change between nearby states at phase ρ; it is unrelated to the losses $\mathcal{L}$.

Theorem 1 (Phase-Conditioned BC on Minimum-Effort Demonstrations is OT-CFM)**.** *Under (A1)–(A3):*

- *(i)* ***Demonstrations are OT-paired straight lines.*** *Almost every demonstration is $\mathbf{x}_\rho = (1-\rho)\mathbf{x}_0 + \rho\,\mathcal{T}(\mathbf{x}_0)$, where $\mathcal{T} = \nabla f$ (f convex) is the Brenier map from P_S to P_G, so $(\mathbf{x}_0,\mathbf{x}_1)\sim\gamma^\star$ and a dataset $\mathcal{D}_\rho$ (Eq. 3) built from minimum-effort demonstrations is automatically OT-paired.*
- *(ii)* ***Same objective.*** *$\mathcal{L}_{\mathrm{BC}}(\phi) = \mathcal{L}^{\gamma^\star}_{\mathrm{CFM}}(\phi)$ for every phase-conditioned policy π_ϕ.*
- *(iii)* ***No mode averaging with phase.*** *Demonstrations never cross in state–phase space, so $\bar{v}$ fits every demonstration exactly: $\bar{v}(\mathbf{x}_\rho,\rho) = \mathbf{x}_1 - \mathbf{x}_0$ and $\mathcal{L}_{\mathrm{BC}}(\bar{v}) = 0$. On $(0,1)$, $\bar{v}$ is $L(\rho)$-Lipschitz with $L(\rho) = \max\{1/\rho, 1/(1-\rho)\}$, so each demonstration is the unique rollout of $\bar{v}$ on $[\rho_0,1)$ from its state at any $\rho_0\in(0,1)$.*
- *(iv)* ***Phase is necessary.*** *There are P_S, P_G satisfying (A2) for which every phase-free policy has $\mathcal{L}_{\mathrm{BC}} > 0$ and the rollouts of $\bar{v}_{\mathrm{free}}$ do not reach P_G.*

Proof idea (App. A.1.1). Jensen's inequality and the definition of W_2 give $\mathcal{E} \ge \frac{1}{2}\mathbb{E}\|\mathbf{x}_1-\mathbf{x}_0\|^2 \ge \frac{1}{2}W_2^2(P_S,P_G)$, with equality exactly for straight, W_2-optimally paired paths; with Brenier's theorem this gives (i), and substitution into Eq. 4 gives (ii). For (iii), monotonicity of $\mathcal{T}$ keeps demonstrations from different starts apart until $\rho = 1$. Step (ii) holds for straight demonstrations with any coupling, including the independent coupling of standard CFM, and flow-matching inference is the policy's rollout (Lemma 1).

Proposition 2 (Phase-Conditioned BC Preserves the Demonstration Marginals)**.** *Drop (A1)–(A3): let the demonstrations be any random paths with $\mathbb{E}\int_0^1 \|\dot{\mathbf{x}}_\rho\|^2 d\rho < \infty$, curved or crossing, with any pairing. Then p_ρ solves the continuity equation $\partial_\rho p_\rho + \nabla\cdot(p_\rho\,\bar{v}) = 0$, and if $\bar{v}$ is $L(\rho)$-Lipschitz with $\int_{\rho_0}^{\rho_1} L(\rho)\,d\rho < \infty$ and $\mathbb{E}\|\mathbf{x}_{\rho_0}\| < \infty$, rolling out $\bar{v}$ from $\mathbf{x}_{\rho_0}\sim p_{\rho_0}$ reproduces p_ρ for every $\rho\in[\rho_0,\rho_1]$. The phase-free optimum satisfies only the time-integrated identity $\nabla\cdot(\mu\,\bar{v}_{\mathrm{free}}) = p_0 - p_1$, where $\mu = \int_0^1 p_\rho\,d\rho$.*

What this means for deterministic policies. Averaging velocities does not average the distribution: demonstrations move p_ρ only through $\bar{v}$ (Proposition 2, App. A.1.3), so with phase the MSE optimum reroutes crossing demonstrations rather than collapsing them. Without phase, $\bar{v}_{\mathrm{free}}$ matches only the net flux $p_0 - p_1$, so rollouts can stall (App. A.1.1) or overshoot (Theorem 1(iv)). At a decision point, regularity is what fails: by Grönwall's inequality, a policy with $\int_0^1 L(\rho)\,d\rho < \infty$ maps starts to ends continuously and cannot split a connected start distribution into modes, which the optimum of Theorem 1(iii) escapes because its Lipschitz bound is not integrable at $\rho = 0$. A trained network, with finite $L(\rho)$, therefore averages modes at a split and needs start noise, and no deterministic policy can split demonstrations that share a state (App. A.1.4), a prediction we test in Sec. 5.

Scope. Theorem 1(i)–(iii) extend to Riemannian manifolds, except for the formula for $L(\rho)$, and Proposition 2 covers the nonholonomic navigation task but not objects moved through contact, such as the Push-T block (App. A.1.5).

The score regularizer targets the split that a smooth network cannot resolve. With $\mathbf{u}_\rho = \dot{\mathbf{x}}_\rho$ the demonstrated velocity and $\mathbf{s}_\rho = \nabla_{\mathbf{x}} \log \hat{p}_\rho$ the score of a Gaussian kernel density estimate of the demonstrated states at phase ρ (App. A.2), it is an L_1 penalty on the velocity error along the score, inspired by Huang et al. (2025):

$$\mathcal{L}_{reg}(\phi) = \mathbb{E}_{\rho\sim\mathcal{U}[0,1]}\, \mathbb{E}\left|\big(\mathbf{u}_\rho - \pi_\phi(\mathbf{x}_\rho, \rho)\big) \cdot \mathbf{s}_\rho(\mathbf{x}_\rho)\right|. \tag{6}$$

At a state $\mathbf{x}$ and phase ρ with $\mathbf{s} = \mathbf{s}_\rho(\mathbf{x}) \neq 0$, let $\hat{\mathbf{s}} = \mathbf{s}/\|\mathbf{s}\|$ and let $b = \mathbf{u}_\rho \cdot \hat{\mathbf{s}}$ be the demonstrated velocity along the score. Over the demonstrations at $(\mathbf{x}, \rho)$, its mean is $\bar{b} = \bar{v}(\mathbf{x}, \rho) \cdot \hat{\mathbf{s}}$; let m be its median (the one closest to $\bar{b}$ if the median is not unique).

Theorem 3 (Score Regularization Moves the Policy Toward the Median Velocity Along the Score)**.** *Let π_λ minimize $\mathcal{L}_{\mathrm{BC}} + \lambda\mathcal{L}_{reg}$ over all phase-conditioned policies. Where $\mathbf{s} = 0$, $\pi_\lambda = \bar{v}$. Where $\mathbf{s} \neq 0$: (i) the component of π_λ orthogonal to $\mathbf{s}$ equals that of $\bar{v}$; (ii) $a_\lambda = \pi_\lambda \cdot \hat{\mathbf{s}}$ lies between $\bar{b}$ and m, satisfies $|a_\lambda - \bar{b}| \leq \lambda\|\mathbf{s}\|/2$, and tends to m as $\lambda\|\mathbf{s}\| \to \infty$; (iii) $\pi_\lambda = \bar{v}$ wherever the demonstrations at $(\mathbf{x}, \rho)$ agree along $\mathbf{s}$, in particular wherever they do not cross (as in Theorem 1), and wherever b is symmetric about $\bar{b}$.*

Proof idea (App. A.1.6). Across $\mathbf{s}$ the pointwise objective is the squared loss alone. Along $\hat{\mathbf{s}}$ it is the sum of $\mathbb{E}(a-b)^2$, which is minimized by the mean $\bar{b}$, and $\lambda\|\mathbf{s}\|\,\mathbb{E}|a-b|$, which is minimized by the median m.

What the regularizer does. $\mathcal{L}_{reg}$ is not an OT surrogate: it uses no endpoint information, is invariant to $\mathbf{s} \to -\mathbf{s}$, is inactive where demonstrations do not cross, and cannot break a symmetric tie. Where branches overlap, the median lies in the heavier one, so the penalty is mode-seeking and steepens transitions between branches. Its cost is bias: a large $\lambda\|\mathbf{s}\|$ forfeits the marginal preservation of Proposition 2 and can suppress minority behavior (App. A.1.6), as seen with 95% duplicate demonstrations (App. A.3). We apply it at splits because a smooth network effectively averages nearby branches there, the overlapping regime of Theorem 3; this is motivation, not a result about trained networks.

4 METHODS

Inspired by Theorems 1 and 3, we build a deterministic BC policy that captures multimodal flows without iterative compute or stochastic sampling at inference, using two mechanisms: 1) a phase-conditioned policy, and 2) score-based regularization. Given a phase-labeled dataset $\mathcal{D}_\rho$ (Eq. 3), which need not be OT-paired, the learning task is

$$\phi^* = \arg\min_{\phi}\ \mathbb{E}_{(\rho,\mathbf{x},\mathcal{O},\mathbf{u})\sim\mathcal{D}_\rho}\left\|\pi_\phi(\mathbf{x}, \mathcal{O}, \rho) - \mathbf{u}\right\|^2 + \lambda\,\mathcal{L}_{reg}(\phi). \tag{7}$$

Phase-Conditioned Deterministic Policy

We define a *phase-conditioned* deterministic policy as,

$$\mathbf{u} = \pi_\phi(\mathbf{x}, \mathcal{O}, \rho) \tag{8}$$

where $\mathbf{u}$, $\mathbf{x}$, $\mathcal{O}$, and ρ are robot actions (state derivatives) and states, observations, and task progress respectively. We make ρ a deterministic, monotonically increasing function, like the phase in flow matching: either a linear clock with step $\Delta\rho$ or a monotone MLP task-progress estimator on the observation $\mathcal{O}$ with minimum increment ε, both clamped at 1:

$$\rho_{new} = \begin{cases} \rho + \Delta\rho & \text{(Linear)} \\ \max(\mathrm{MLP}_{\mathrm{TP}}(\mathcal{O}),\ \rho + \varepsilon) & \text{(MLP)} \end{cases} \tag{9}$$

We present two architectural implementations of our policy, 1) an MLP that takes $\mathbf{x}$, $\mathcal{O}$, and ρ as inputs and returns an action, $\mathbf{u}$, and 2) Phaseformer, a transformer backbone that takes $\mathbf{x}$ and $\mathcal{O}$ and returns $\mathbf{u}$. Phaseformer injects ρ into each block via Adaptive Layer Norms (AdaLN) (Peebles & Xie, 2023), whose phase-dependent scale, shift, and gate can be viewed as creating a different policy at each step of the task (Fig. 1). Unlike ACT, whose CVAE latent is sampled in training and set to its prior mean at inference, Phaseformer conditions on explicit task progress and relies on variation in the demonstrated start states to resolve mode averaging. Without start-state variation,

any deterministic policy averages modes at the fork (App. A.1.4); Sec. 5 shows that 5–10% noisy-start demonstrations suffice.

Score-Based Regularization
We add the regularizer of Eq. 6 (restated here),

$$\mathcal{L}_{reg}(\phi) = \mathbb{E}_{\rho,\,(\mathbf{x}_\rho,\mathbf{u}_\rho)\sim\mathcal{D}}\Big|\big(\mathbf{u}_\rho - \pi_\phi(\mathbf{x}_\rho,\rho)\big)\cdot\nabla_{\mathbf{x}}\log\hat{p}_\rho(\mathbf{x}_\rho)\Big| \tag{10}$$

which by Theorem 3 moves predictions from the average demonstrated velocity toward the median one along the score wherever demonstrations overlap. Rather than learning a score network, we use the closed-form gradient of a Gaussian KDE of the demonstrations at each phase step (Fukunaga & Hostetler, 1975; Comaniciu & Meer, 2002) (App. A.2). This works because our actions are state derivatives, unlike the time-dependent actions of streaming flow policy, so the score lies in the action space and velocity errors can be penalized along it. Divergence-based regularizers, including full conditional divergence matching (Huang et al., 2025), degraded performance (App. A.5).

5 Experiments

We evaluate our proposed models on two SE(2) robotics tasks: ***non-holonomic navigation*** and ***in-plane manipulation*** (Push-T). We compare against three state-of-the-art baselines: Diffusion Policy (DP; Chi et al., 2024), Action Chunking Transformer (ACT; Zhao et al., 2023), and Minimal Iterative Policy (MIP; Pan et al., 2025).

For ***robot navigation***, we evaluate a 134K-parameter version of our MLP policy on three scenarios: A) single start, single goal, single path, B) single start, dual goal, dual path, and C) single start, single goal, dual path (Fig. 3). The policy maps an augmented state (including phase) to a body twist, strictly satisfying non-holonomic kinematic constraints. Collision-free demonstrations were generated using Theseus (Pineda et al., 2022). During rollouts, phase was generated via Eq. 9 (linear). We simulated real-world execution by holding controls constant over integration steps and clamping outputs to the planner's operational bounds. Details are in Apps. A.4 and A.6.

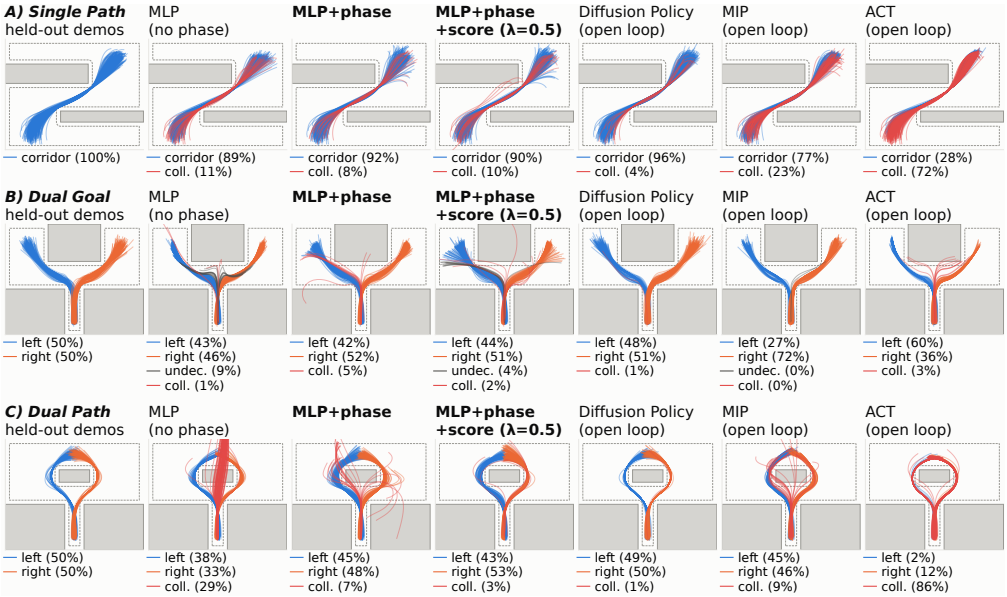

Figure 3: Robot Navigation Demonstrations and Rollouts (seed 0, 1024 held-out starts) of all policies.

For ***Push-T***, we evaluate both a larger (790K params) MLP policy and Phaseformer (907K params) trained on the human demonstrations of Chi et al. (2024). Both policies map the joint object-effector state and current phase to the effector's *phase velocity*; the MLP takes phase as an input and Phaseformer injects it via AdaLN. Phase comes from the MLP estimator of Eq. 9. See Appendix A.6 for network, estimator, normalization, and control details.

We trained ten seeds of each policy with and without phase and score, and ten seeds each of DP, MIP, and ACT as in their original works (App. A.7). For ***Push-T*** we report score (maximum goal coverage over an episode, normalized as in Chi et al. (2024)), stalled episodes, final position and orientation error, and success rate (final coverage above 0.95). We also benchmark CPU and GPU latency for all policies.

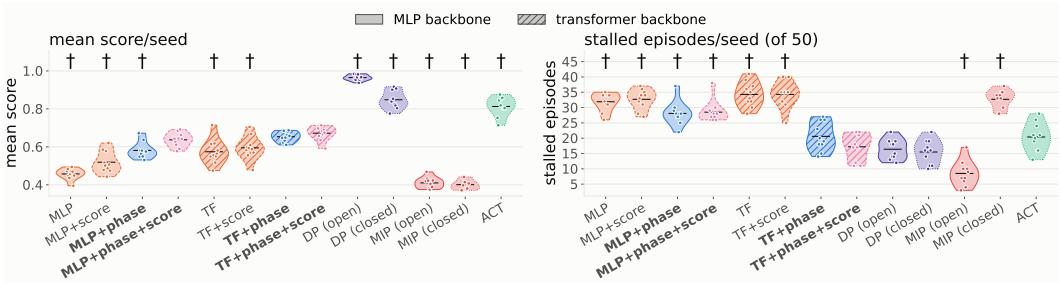

Figure 4: Push-T Episode Performance: mean score and stalled episodes (pusher moved < 1 px over the last 50 steps). Dots: 10 seeds; bar: mean. ACT runs closed loop. †: $p < 0.05$ vs. TF+phase+score (Welch over seeds).

Phase injection and score regularization reduce mode averaging. In navigation scenario C (Fig. 3), the deterministic policy without phase or score drives straight into the block on 29% of rollouts; with phase it selects left (45%) or right (48%), cutting collisions to 7%, and score regularization cuts them to 3%, comparable to DP's 1% (more in App. A.8). In Push-T, phase injection reduces stalls for the transformer but not the MLP: Phaseformer and Phaseformer+score (TF+phase and TF+phase+score in plots) stall at rates not significantly different from DP (open/closed-loop) and ACT (Fig. 4). Mean score also rises with phase: MLP+phase+score and Phaseformer are not significantly different from Phaseformer+score, while MLP+phase and all no-phase arms score significantly lower.

Theory predicts where deterministic policies will fail and experiments confirm it. For demonstrations that share a start state, Lemma 2 (App. A.1.4) says the MSE target at the fork is the average of the two branches and a deterministic policy carries the shared state along a single path; the score penalty leaves this optimum unchanged, since the kernel score vanishes where demonstrations coincide and past the fork they agree (Theorem 3(iii)). No combination of phase and score should succeed. We test this by rebuilding scenario C with a fixed start and no via-point jitter: 512 copies each of a left and right path (App. A.3). As predicted, no seed of the four MLP arms reaches the goal from the nominal start (Fig. 9). Via-point jitter, which spreads the demonstrations only after the fork ($\rho \approx 0.15$), still fails in every seed (Lemma 2(c)), while every arm placed on a demonstration at $\rho \geq 0.3$ follows it to the goal in 95–100% of restarts, so the failure is the split, not tracking. Restoring start variation in 5% of demonstrations recovers 80–88% success for MLP, MLP+score, and MLP+phase, and 10–15% gives 96–99% for every arm, even without phase, since here start variation, not phase, assigns starts to modes. This shows that deterministic BC needs stochasticity, but it can come from the training starts rather than from inference.

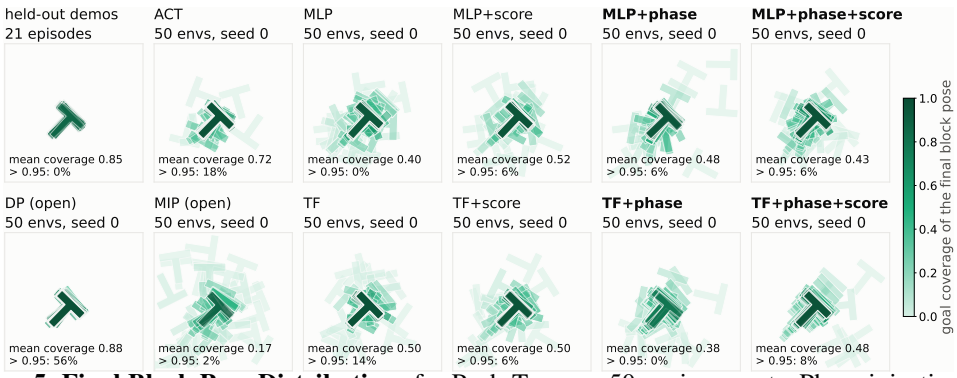

Figure 5: Final Block Pose Distributions for Push-T across 50 environments. Phase injection and score regularization let deterministic policies resolve orientation errors.

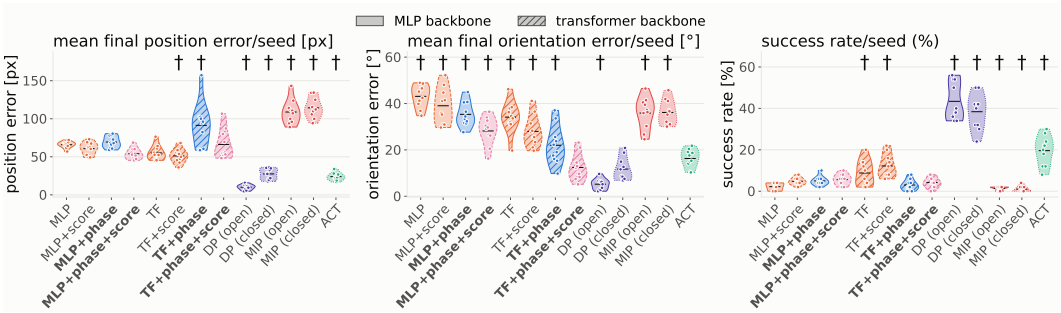

Figure 6: Push-T Final-Object-Pose Accuracy measured by position error, orientation error, and success rate. Per-seed panels: 10 seeds (dots), mean (bar). Success: final goal coverage > 0.95. † marks $p < 0.05$ against TF+phase+score (Welch over seeds).

Phase injection and score regularization resolve orientation errors in the Push-T rollouts (Figs. 5 and 6). Only open-loop DP has significantly lower orientation error than Phaseformer+score, which is not significantly different from closed-loop DP and ACT and significantly better than all other deterministic arms and both MIP variants. Phaseformer+score's low success rate stems mainly from final position error: success requires 95% goal coverage, which a correctly oriented but translated block misses.

Score-regularized deterministic models with phase are Pareto optimal. Figure 7 shows the trade-off between CPU and GPU inference time and success for both tasks. Our deterministic Push-T policies operate right at the control rate needed for tactile control (Massari et al., 2019; Tidare et al., 2018). They also have faster CPU runtimes than GPU runtimes due to low parameter counts. All of the Pareto-optimal policies that do not require iterative compute (our deterministic policies and ACT) run closed-loop under the suggested control rate for visual control (Keipour et al., 2022).

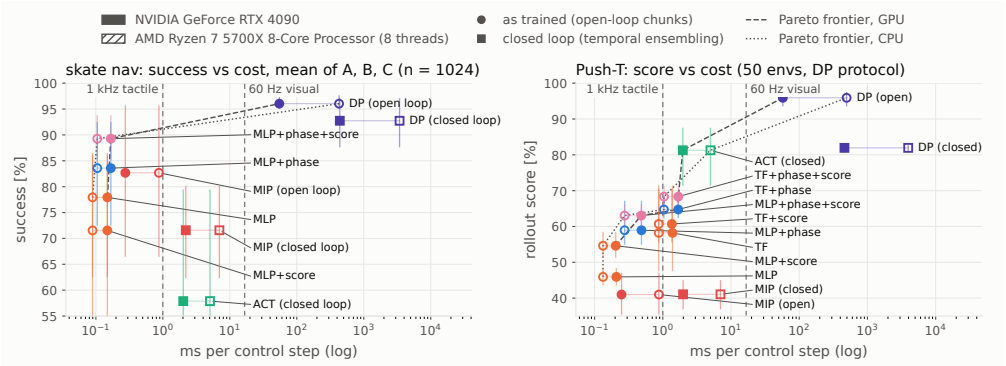

Figure 7: Task Success Versus Inference Latency Pareto frontier for robot navigation and Push-T. Tactile (Massari et al., 2019) and visual (Keipour et al., 2022) feedback control rates demarcated.

6 Discussion and Conclusions

While recent works have established a post-training equivalence between flow matching and imitation learning through adversarial optimization (Ma et al., 2026) and dual on-policy distillation frameworks (Wan et al., 2026), our derivation provides a fundamentally distinct perspective. Rather than relying on post-hoc alignment or minimax games, Theorem 1 derives this connection from minimum-effort demonstrations and Lemma 1 extends it to any coupling, isolating phase conditioning as the structural gap between flow matching and deterministic behavior cloning.

Historically, machine learning overcomes mode averaging by conditioning the distribution on an additional latent or categorical variable. Mixture density networks resolve multimodality by conditioning outputs on a discrete mode identity (Bishop, 1994) and CVAEs use a latent variable z to collapse a complex multimodal distribution $p(\mathbf{u}|\mathbf{x})$ into a unimodal conditional distribution $p(\mathbf{u}|\mathbf{x}, z)$ (Sohn et al., 2015). Flow matching conditions each training label on a target $\mathbf{x}_1$, making it unimodal (Lipman et al., 2023), but the network learns only the labels' average $\bar{v}(\mathbf{x}, \rho)$, which still transports

every marginal p_ρ (Proposition 2); its modes come from a stochastic start. Phaseformer keeps the phase conditioning and takes that start variation from the demonstrations.

A second, largely orthogonal thread concerns whether a policy represents task progress at all. Dynamic Movement Primitives (DMPs) made an explicit phase variable, a canonical system that decays monotonically over one execution, the organizing structure of a long line of work in motor control (Ijspeert et al., 2002; 2013). Their probabilistic relatives do not separate competing demonstrations on their own: a single ProMP is a unimodal Gaussian over basis-function weights (Paraschos et al., 2013), and GMR from a task-parameterized GMM returns a responsibility-weighted conditional mean (Calinon, 2016), so multimodality was handled by fitting several primitives and selecting among them (e.g., Maeda et al., 2017), the same mechanism as the MDN/CVAE solutions above. Because these models were fit over low-dimensional state and phase had to be aligned across demonstrations of different speed, the lineage scaled poorly to high-dimensional sensory input, and the shift to reactive deep policies dropped phase indexing in part for this reason, reintroducing the temporal aliasing behind the phase-free failures of Theorem 1(iv). Diffusion and flow-matching policies restore a phase-like variable, but it indexes denoising rather than task progress (Sec. 3). Phaseformer is closer to the DMP lineage, but in Push-T its phase comes from a learned monotone estimator (Eq. 9), and it modulates a shared backbone via AdaLN rather than weighting a fixed kernel bank. Nor does Phaseformer select an explicit primitive: modes are resolved implicitly by phase-and-state conditioning (Theorem 1(iii)), with the score regularizer moving the policy toward the median demonstrated velocity where branches overlap (Theorem 3). Whether this implicit route is preferable to an explicit categorical one, rather than a less inspectable reparameterization of it, is a question these experiments do not settle.

Our results also qualify recent benchmarking that attributes the success of generative policies to supervised iterative computation paired with noise injection rather than to an inherent ability to capture multimodality (Pan et al., 2025). Neither iterative computation nor stochastic sampling at inference is required: a single-step deterministic policy resolves multimodal decision points if its training starts vary, with phase removing temporal aliasing (Sec. 5). ACT (Zhao et al., 2023) already follows this pattern: it decodes the prior mean of its latent at inference, so its stochasticity enters only during training, through the CVAE. Phaseformer obtains the same property without training a CVAE. It has no encoder, latent variable, or KL term; the variation needed to assign starts to modes comes directly from the demonstrated start states, and phase supplies the temporal context. The resulting policies also run with lower CPU latency than ACT on both tasks (Figs. 7 and 19).

The regularized Phaseformer's ability to resolve block orientation errors and the low rate of stalled rollouts are significant findings. In complex multimodal tasks like Push-T the average mode is often zero-action (Chi et al., 2024). Phaseformer (regularized and unregularized) reduces stalls to the level of DP and ACT, while no-phase TF and all MLP variants do not, so phase injection in the transformer removes the zero-action averaging behind stalls. Score regularization leaves stall rates unchanged; its effect in Push-T is on orientation (Fig. 6).

For tasks with critical decision points (e.g., navigation scenarios B and C), the combination of demonstrations drawn from a start distribution, temporal context from phase, and a score regularizer that steepens transitions where modes overlap lets deterministic architectures solve multimodal tasks with single-step inference, underscoring the value of task-progress models that infer completion from scene observations (Kroemer et al., 2015). Our experiments are limited to short-horizon SE(2) tasks and need varied demonstration starts (App. A.3); scaling to SE(3) will require longer horizons, more complex demonstration distributions and strategies for training task-progress models. Deterministic behavior-cloned policies are thus a viable, low-latency, single-step alternative to multi-step diffusion and flow matching models.

AI USE AND REPRODUCIBILITY

AI Use Disclosure Statement:
In this work, we used generative AI tools to help: generate synthetic data sets, design and provide feedback on research methodology or experiments, implement methods, clean and reformat datasets, support qualitative and thematic data analysis, provide critical ingredients for proving mathematical claims, create and modify scientific figures, sourcing/searching for information, refine proofs, and edit a research paper to improve readability, identify relevant literature, and format references. We did not use generative AI tools to: propose or refine hypotheses, help develop theoretical models or conceptual frameworks, formulate mathematical claims, or interpret results. Assisting with translation was not applicable to this work. We have reviewed all AI-assisted work to ensure that authors received credit where due, that our claims were mathematically sound, and our results were accurate. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Reproducibility Statement:
All code is included in the supplemental materials and is available at `https://github.com/anonymousphaseformer/phaseformer.git`

REFERENCES

Luigi Ambrosio, Nicola Gigli, and Giuseppe Savaré. *Gradient Flows in Metric Spaces and in the Space of Probability Measures*. Lectures in Mathematics ETH Zürich. Birkhäuser, Basel, second edition, 2008.

G. K. Batchelor. *An Introduction to Fluid Dynamics*. Cambridge University Press, February 2000. ISBN 978-1-139-64346-7. Google-Books-ID: aXQgAwAAQBAJ.

Christopher M. Bishop. Mixture density networks, 1994. Num Pages: 438543 Place: Birmingham Publisher: Aston University.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. Pi$_0$: A Vision-Language-Action Flow Model for General Robot Control, November 2024. arXiv:2410.24164 [cs].

Yann Brenier. Polar factorization and directed graphs with applications to the fluid mechanics boundary value problem. *Communications on pure and applied mathematics*, 44(4):375–417, 1991.

Sylvain Calinon. A tutorial on task-parameterized movement learning and retrieval. *Intelligent Service Robotics*, 9(1):1–29, 2016.

José E. Chacón, Tarn Duong, and M. P. Wand. Asymptotics for general multivariate kernel density derivative estimators. *Statistica Sinica*, 21(2):807–840, 2011.

Ricky TQ Chen and Yaron Lipman. Flow matching on general geometries. In *International Conference on Learning Representations*, volume 2024, pp. 47922–47945, 2024.

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, October 2024. ISSN 0278-3649. doi: 10.1177/02783649241273668. Publisher: SAGE Publications Ltd STM.

Gregory S. Chirikjian. *Stochastic Models, Information Theory, and Lie Groups, Volume 2: Analytic Methods and Modern Applications*. Springer Science & Business Media, 2011.

Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

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 *Conference on Robot Learning*, pp. 158–168. PMLR, 2022.

Keinosuke Fukunaga and Larry D. Hostetler. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*, 21(1):32–40, 1975.

Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. FFJORD: Free-form Continuous Dynamics for Scalable Reversible Generative Models, October 2018. arXiv:1810.01367 [cs].

Yuhao Huang, Taos Transue, Shih-Hsin Wang, William M Feldman, Hong Zhang, and Bao Wang. Improving flow matching by aligning flow divergence. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 25813–25834. PMLR, 13–19 Jul 2025.

M.F. Hutchinson. A Stochastic Estimator of the Trace of the Influence Matrix for Laplacian Smoothing Splines. *Communications in Statistics - Simulation and Computation*, 18(3):1059–1076, January 1989. ISSN 0361-0918. doi: 10.1080/03610918908812806. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/03610918908812806.

Auke Jan Ijspeert, Jun Nakanishi, and Stefan Schaal. Learning attractor landscapes for learning motor primitives. In *Advances in Neural Information Processing Systems*, volume 15, 2002.

Auke Jan Ijspeert, Jun Nakanishi, Heiko Hoffmann, Peter Pastor, and Stefan Schaal. Dynamical movement primitives: learning attractor models for motor behaviors. *Neural computation*, 25(2): 328–373, 2013.

Sunshine Jiang, Xiaolin Fang, Nicholas Roy, Tomás Lozano-Pérez, Leslie Pack Kaelbling, and Siddharth Ancha. Streaming Flow Policy: Simplifying diffusion/flow-matching policies by treating action trajectories as flow trajectories. *arXiv e-prints*, art. arXiv:2505.21851, May 2025. doi: 10.48550/arXiv.2505.21851.

Ali Keipour, Guilherme A. S. Pereira, Rogerio Bonatti, et al. Visual servoing approach to autonomous uav landing on a moving vehicle. *Sensors*, 22(17):6549, 2022. doi: 10.3390/s22176549.

Oliver Kroemer, Christian Daniel, Gerhard Neumann, Herke Van Hoof, and Jan Peters. Towards learning hierarchical skills for multi-phase manipulation tasks. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1503–1510. IEEE, 2015.

Christian Léonard. From the Schrödinger problem to the Monge–Kantorovich problem. *Journal of Functional Analysis*, 262(4):1879–1920, 2012.

Sergey Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

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

Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky T. Q. Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow Matching Guide and Code, December 2024. arXiv:2412.06264 [cs].

D. O. Loftsgaarden and C. P. Quesenberry. A Nonparametric Estimate of a Multivariate Density Function. *The Annals of Mathematical Statistics*, 36(3):1049–1051, June 1965. ISSN 0003-4851, 2168-8990. doi: 10.1214/aoms/1177700079. Publisher: Institute of Mathematical Statistics.

Yeyao Ma, Chen Li, Xiaosong Zhang, Han Hu, and Weidi Xie. FAIL: Flow matching adversarial imitation learning for image generation. *arXiv preprint arXiv:2602.12155*, 2026.

Guilherme J. Maeda, Gerhard Neumann, Marco Ewerton, Rudolf Lioutikov, Oliver Kroemer, and Jan Peters. Probabilistic movement primitives for coordination of multiple human–robot collaborative tasks. *Autonomous Robots*, 41(3):593–612, 2017. doi: 10.1007/s10514-016-9556-2.

Luca Massari, Calogero Maria Oddo, Edoardo Sinibaldi, et al. Tactile sensing and control of robotic manipulator integrating fiber bragg grating strain-sensor. *Frontiers in Neurorobotics*, 13:8, 2019. doi: 10.3389/fnbot.2019.00008.

Robert J. McCann. Polar factorization of maps on Riemannian manifolds. *Geometric and Functional Analysis*, 11(3):589–608, 2001.

Toshio Mikami. Monge's problem with a quadratic cost by the zero-noise limit of h-path processes. *Probability Theory and Related Fields*, 129(2):245–260, 2004.

Chaoyi Pan, Giri Anantharaman, Nai-Chieh Huang, Claire Jin, Daniel Pfrommer, Chenyang Yuan, Frank Permenter, Guannan Qu, Nicholas Boffi, Guanya Shi, and Max Simchowitz. Much ado about noising: Dispelling the myths of generative robotic control. *arXiv preprint arXiv:2512.01809*, 2025.

Alexandros Paraschos, Christian Daniel, Jan Peters, and Gerhard Neumann. Probabilistic movement primitives. In *Advances in Neural Information Processing Systems*, volume 26, 2013.

Emanuel Parzen. On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076, 1962. Publisher: JSTOR.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4199–4209, 2023.

Luis Pineda, Taosha Fan, Maurizio Monge, Shobha Venkataraman, Paloma Sodhi, Ricky TQ Chen, Joseph Ortiz, Daniel DeTone, Austin Wang, Stuart Anderson, Jing Dong, Brandon Amos, and Mustafa Mukadam. Theseus: A Library for Differentiable Nonlinear Optimization. *Advances in Neural Information Processing Systems*, 2022.

Dean A. Pomerleau. ALVINN: An Autonomous Land Vehicle in a Neural Network. In *Advances in Neural Information Processing Systems*, volume 1. Morgan-Kaufmann, 1988.

Aram-Alexandre Pooladian, Heli Ben-Hamu, Yaron Lipman, and Alexander Tong. Multisample flow matching: Straightening flows with minibatch optimal transport. In *International Conference on Machine Learning*, pp. 28131–28156. PMLR, 2023.

R. Tyrrell Rockafellar and Roger J.-B. Wets. *Variational Analysis*, volume 317 of *Grundlehren der mathematischen Wissenschaften*. Springer, 1998.

Abraham. Savitzky and M. J. E. Golay. Smoothing and Differentiation of Data by Simplified Least Squares Procedures. *Analytical Chemistry*, 36(8):1627–1639, July 1964. ISSN 0003-2700, 1520-6882. doi: 10.1021/ac60214a047.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

Jens Tidare, Erik Åstrand, and Mikael Ekström. Evaluation of closed-loop feedback system delay. In *Proceedings of the 11th International Joint Conference on Biomedical Engineering Systems and Technologies*, pp. 187–193. SciTePress, 2018. doi: 10.5220/0006598301870193.

Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research*, pp. 1–34, 2024.

Julen Urain, Ajay Mandlekar, Yilun Du, Mahi Shafiullah, Danfei Xu, Katerina Fragkiadaki, Georgia Chalvatzaki, and Jan Peters. Deep generative models in robotics: A survey on learning from multimodal demonstrations. *CoRR*, abs/2408.04380, 2024.

Cédric Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2009.

Zhenglin Wan, Jingxuan Wu, Xingrui Yu, Chubin Zhang, Mingcong Lei, Bo An, Ivor W. Tsang, and Yang You. Adversarial dual on-policy distillation from expressive teacher. *arXiv preprint arXiv:2605.27095*, 2026.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Robotics: Science and Systems XIX*. Robotics: Science and Systems Foundation, 2023. doi: 10.15607/rss.2023.xix.016.

A Appendix

A.1 Proofs and Supplementary Results for Theorems 1 and 3

A.1.1 Proof of Theorem 1

The best phase-free policy. A phase-free policy sees the state but not the phase, so its squared loss is minimized by $\mathbb{E}[\dot{\mathbf{x}}_\rho \mid \mathbf{x}_\rho = \mathbf{x}]$ with $\rho \sim \mathcal{U}[0,1]$ random. When the p_ρ have densities, Bayes' rule gives ρ the conditional density $p_\rho(\mathbf{x}) / \int_0^1 p_{\rho'}(\mathbf{x})\, d\rho'$ given $\mathbf{x}_\rho = \mathbf{x}$, so

$$\bar{v}_{\text{free}}(\mathbf{x}) = \frac{\int_0^1 p_\rho(\mathbf{x})\, \bar{v}(\mathbf{x}, \rho)\, d\rho}{\int_0^1 p_\rho(\mathbf{x})\, d\rho}. \tag{11}$$

It mixes velocities from different phases. Demonstrations that go out and come back along the same route give $\bar{v}_{\text{free}} = 0$ on that route, whatever their speeds: a pass at speed c spends time proportional to $1/c$ near each state, so it contributes $\pm c \cdot \frac{1}{c} = \pm 1$ to the numerator of Eq. 11, and the outward and return passes cancel. For the single demonstration $x_\rho = \sin(\pi\rho)$, the phase-free rollout from $x = 0$ never moves, although the demonstration visits all of $[0, 1]$.

Proof. **(i)** For each path, Jensen's inequality gives $\frac{1}{2}\|\mathbf{x}_1 - \mathbf{x}_0\|^2 = \frac{1}{2}\big\|\int_0^1 \dot{\mathbf{x}}_\rho\, d\rho\big\|^2 \le \int_0^1 \frac{1}{2}\|\dot{\mathbf{x}}_\rho\|^2 d\rho$, with equality iff the velocity is constant. Since $(\mathbf{x}_0, \mathbf{x}_1)$ is a coupling of P_S and P_G, averaging gives $\mathcal{E} \ge \frac{1}{2}\mathbb{E}\|\mathbf{x}_1 - \mathbf{x}_0\|^2 \ge \frac{1}{2}W_2^2(P_S, P_G)$. Straight lines between $\gamma^\star$-paired endpoints attain both bounds, so every minimizer does: its paths are straight and its endpoints are W_2-optimally paired. By (A2) and Brenier's theorem, $\gamma^\star$ is unique and supported on the graph of $\mathcal{T} = \nabla f$ (Brenier, 1991; Villani, 2009).

(ii) Substitute $\mathbf{x}_\rho = (1-\rho)\mathbf{x}_0 + \rho\mathbf{x}_1$ and $\dot{\mathbf{x}}_\rho = \mathbf{x}_1 - \mathbf{x}_0$ from (i) into Eq. 4. Lemma 1 extends this identity to every coupling.

(iii) Take two starts $\mathbf{a}, \mathbf{b}$ and let $\Delta_0 = \mathbf{a} - \mathbf{b}$ and $\Delta_1 = \mathcal{T}(\mathbf{a}) - \mathcal{T}(\mathbf{b})$. Because f is convex, $\mathcal{T}$ is monotone, $\langle \Delta_0, \Delta_1 \rangle \ge 0$, so the two demonstrations' states at phase ρ satisfy

$$\|\mathbf{x}_\rho - \mathbf{x}'_\rho\|^2 = \|(1-\rho)\Delta_0 + \rho\Delta_1\|^2 \;\ge\; (1-\rho)^2\|\Delta_0\|^2 + \rho^2\|\Delta_1\|^2. \tag{12}$$

For $\rho < 1$, equal states force $\mathbf{a} = \mathbf{b}$: demonstrations never cross, so the phase-ρ state determines the start and hence the velocity, and $\mathcal{L}_{\text{BC}}(\bar{v}) = 0$. The velocities differ by $\Delta_1 - \Delta_0$, and by Eq. 12, $\|\Delta_1 - \Delta_0\|^2 \le \|\Delta_0\|^2 + \|\Delta_1\|^2 \le \|\mathbf{x}_\rho - \mathbf{x}'_\rho\|^2 / \min(\rho, 1-\rho)^2$, which is the bound on $L(\rho)$. Every $\mathbf{x} \in \mathbb{R}^d$ can be written in exactly one way as $(1-\rho)\mathbf{a} + \rho\mathbf{t}$ with $\mathbf{t} \in \partial f(\mathbf{a})$, because $\frac{1-\rho}{2}\|\cdot\|^2 + \rho f$ is strongly convex (Rockafellar & Wets, 1998, Prop. 12.60); setting $\bar{v}(\mathbf{x}, \rho) = \mathbf{t} - \mathbf{a}$ extends the bound to all of $\mathbb{R}^d$, since ∂f is monotone too. Each demonstration is a rollout of $\bar{v}$, and it is the only one from its phase-ρ_0 state: two rollouts $\mathbf{z}_\rho, \mathbf{z}'_\rho$ satisfy $\frac{d}{d\rho}\|\mathbf{z}_\rho - \mathbf{z}'_\rho\| \le L(\rho)\,\|\mathbf{z}_\rho - \mathbf{z}'_\rho\|$, and $L(\rho)$ is bounded on every $[\rho_0, \rho_1] \subset (0,1)$, so by Grönwall's inequality rollouts that agree at ρ_0 agree on $[\rho_0, \rho_1]$.

(iv) Take $d = 1$, $P_S = \frac{1}{2}\mathcal{U}[-\delta, \delta] + \frac{1}{2}\mathcal{U}[1-\delta, 1+\delta]$ with $0 < \delta < \frac{1}{2}$, and $\mathcal{T}(a) = a + 2$ on the first block and $a + 4$ on the second, which is nondecreasing and hence the Brenier map onto $P_G = \mathcal{T}_\# P_S$. Both groups pass through $[1+\delta, 2-\delta]$, at speeds 2 and 4, so no phase-free policy fits both and $\mathcal{L}_{\text{BC}} > 0$. As $\delta \to 0$, their occupation densities there are $\frac{1}{4}$ and $\frac{1}{8}$, so Eq. 11 gives $\bar{v}_{\text{free}} = \frac{8}{3}$ on $[1, 2]$, 2 on $[0, 1)$ and 4 on $(2, 5]$, with loss $\frac{1}{3}$. Its rollouts from 0 and 1 end at 2.5 and 4.5 instead of 2 and 5. □

A.1.2 Flow Matching with Any Coupling

Theorem 1(ii) uses only that the demonstrations are straight. The following lemma states the identity for every coupling, so it covers standard CFM with independently drawn endpoints as well as OT-CFM, and it adds the inference side of the equivalence.

Lemma 1 (Flow matching is phase-conditioned BC for any coupling)**.** *Let γ be any coupling of P_S and P_G, which have finite second moments, and let the demonstrations be the straight paths $\mathbf{x}_\rho = (1-\rho)\mathbf{x}_0 + \rho\mathbf{x}_1$ with $(\mathbf{x}_0, \mathbf{x}_1) \sim \gamma$.*

(a) For every policy with $\mathbb{E}\|\pi_\phi(\mathbf{x}_\rho, \rho)\|^2 < \infty$,

$$\mathcal{L}_{\mathrm{BC}}(\phi) = \mathcal{L}^{\gamma}_{\mathrm{CFM}}(\phi) = \mathbb{E}_\rho \mathbb{E} \left\|\pi_\phi(\mathbf{x}_\rho, \rho) - \bar{v}(\mathbf{x}_\rho, \rho)\right\|^2 + C_\gamma, \qquad C_\gamma = \mathcal{L}_{\mathrm{BC}}(\bar{v}),$$

so the BC loss, the CFM loss and the regression onto $\bar{v}$ have the same minimizers. Under (A2), Theorem 1(iii) gives $C_{\gamma^\star} = 0$.

(b) Flow-matching inference, which integrates $d\mathbf{x}/d\rho = \pi_\phi(\mathbf{x}, \rho)$ from $\mathbf{x}_0 \sim P_S$, is the closed-loop rollout of the phase-conditioned policy; whenever the solution is unique, the two coincide.

(c) Standard CFM (independent coupling) and OT-CFM ($\gamma = \gamma^\star$) are therefore both phase-conditioned BC. The converse fails: the single demonstration $x_\rho = \sin(\pi\rho)$ is BC data that no straight-line dataset produces.

Proof. (a) Substituting $\mathbf{x}_\rho = (1-\rho)\mathbf{x}_0 + \rho\mathbf{x}_1$ and $\dot{\mathbf{x}}_\rho = \mathbf{x}_1 - \mathbf{x}_0$ into Eq. 4 gives Eq. 5 term by term. Writing $\pi_\phi - \dot{\mathbf{x}}_\rho = (\pi_\phi - \bar{v}) + (\bar{v} - \dot{\mathbf{x}}_\rho)$ and expanding, the cross term vanishes because $\bar{v}$ is the conditional mean of $\dot{\mathbf{x}}_\rho$ given $(\mathbf{x}_\rho, \rho)$. (b) Both procedures solve the same initial-value problem. (c) For straight demonstrations the velocity $\mathbf{x}_1 - \mathbf{x}_0$ does not depend on ρ, so its distribution is the same at every phase; for $x_\rho = \sin(\pi\rho)$ the velocity at phase ρ is $\pi\cos(\pi\rho)$, which changes with ρ. □

A.1.3 Proof of Proposition 2, and a Closed-Form Example

Proof. For a smooth, compactly supported h, $\frac{d}{d\rho}\mathbb{E}\, h(\mathbf{x}_\rho) = \mathbb{E}[\nabla h(\mathbf{x}_\rho)\cdot\dot{\mathbf{x}}_\rho] = \mathbb{E}[\nabla h(\mathbf{x}_\rho)\cdot\bar{v}(\mathbf{x}_\rho, \rho)]$. The second step averages the velocity first, which is allowed because $\nabla h(\mathbf{x}_\rho)$ is fixed once $\mathbf{x}_\rho$ is; this is the weak form of the continuity equation. Rollouts of $\bar{v}$ satisfy the same equation from the same initial distribution, and for such a Lipschitz field its solution is unique (Ambrosio et al., 2008, Sec. 8.1); the regularity this needs is checked below. Integrating over $\rho \in [0, 1]$ and using $\int_0^1 p_\rho\, \bar{v}\, d\rho = \mu\, \bar{v}_{\mathrm{free}}$ (Eq. 11) gives the phase-free identity. □

The phase-free identity in one dimension. For $d = 1$, integrating $\frac{d}{dx}(\mu\, \bar{v}_{\mathrm{free}}) = p_0 - p_1$ gives $\mu\, \bar{v}_{\mathrm{free}} = F_S - F_G$, where F_S and F_G are the cumulative distribution functions of P_S and P_G; the constant of integration is zero because $\int |\mu\, \bar{v}_{\mathrm{free}}|\, dx \le \mathbb{E}\int_0^1 |\dot{x}_\rho|\, d\rho < \infty$. The phase-free optimum therefore depends on the demonstrations only through their endpoint distributions and the time they spend at each state. In the example of Theorem 1(iv), $F_S - F_G = 1$ and $\mu = \frac{3}{8}$ on $(1, 2)$, which gives $\bar{v}_{\mathrm{free}} = \frac{8}{3}$ again, and for $x_\rho = \sin(\pi\rho)$, $F_S = F_G$ gives $\bar{v}_{\mathrm{free}} = 0$.

Regularity. The uniqueness step uses the theory of Ambrosio et al. (2008, Sec. 8.1) for the Lipschitz extension of $\bar{v}$. A sufficient condition is $\int_{\rho_0}^{\rho_1} \big(L(\rho) + \|\bar{v}(\mathbf{0}, \rho)\|\big)\, d\rho < \infty$, which bounds $\bar{v}$ and its Lipschitz constant on every ball by integrable functions of ρ. The second term is finite whenever $\mathbb{E}\|\mathbf{x}_{\rho_0}\| < \infty$: for every $\mathbf{x}$, $\|\bar{v}(\mathbf{0}, \rho)\| \le \|\bar{v}(\mathbf{x}, \rho)\| + L(\rho)\|\mathbf{x}\|$, and averaging over $\mathbf{x} \sim p_\rho$ bounds it by $\mathbb{E}\|\bar{v}(\mathbf{x}_\rho, \rho)\| + L(\rho)\,\mathbb{E}\|\mathbf{x}_\rho\|$, where $\mathbb{E}\|\mathbf{x}_\rho\| \le \mathbb{E}\|\mathbf{x}_{\rho_0}\| + \mathbb{E}\int_0^1 \|\dot{\mathbf{x}}_\rho\|\, d\rho$.

A closed-form example. Take independent standard Gaussian endpoints. Then $\mathbf{x}_\rho \sim \mathcal{N}(\mathbf{0}, \sigma_\rho^2 \mathbf{I})$ with $\sigma_\rho^2 = (1-\rho)^2 + \rho^2$, and $\bar{v}(\mathbf{x}, \rho) = (2\rho - 1)\,\mathbf{x}/\sigma_\rho^2$, which is $L(\rho)$-Lipschitz with $L(\rho) = |2\rho - 1|/\sigma_\rho^2 \le 1$. Every state receives conflicting labels ($C_\gamma > 0$), yet the rollout $\mathbf{x}(\rho) = \sigma_\rho\, \mathbf{x}(0)$ matches p_ρ at every phase and ends at $\mathbf{x}(1) = \mathbf{x}(0)$. Averaging the velocities preserved every marginal and replaced the random pairing by the identity, which is the OT pairing of $\mathcal{N}(\mathbf{0}, \mathbf{I})$ with itself.

A.1.4 Splits and Smooth Policies

If a policy π is $L(\rho)$-Lipschitz, any two of its rollouts $\mathbf{z}_\rho, \mathbf{z}'_\rho$ satisfy $\frac{d}{d\rho}\|\mathbf{z}_\rho - \mathbf{z}'_\rho\| \le L(\rho)\, \|\mathbf{z}_\rho - \mathbf{z}'_\rho\|$, so by Grönwall's inequality

$$\|\mathbf{z}_1 - \mathbf{z}'_1\| \le e^{\int_0^1 L(\rho)\, d\rho}\, \|\mathbf{z}_0 - \mathbf{z}'_0\|.$$

When $\int_0^1 L(\rho)\, d\rho < \infty$, the rollout is therefore a continuous map from start to end. It sends a connected start distribution to a connected set, so it cannot produce disjoint modes, and starts that end

in modes a gap g apart must begin at least $g\, e^{-\int_0^1 L(\rho)\, d\rho}$ apart. Splitting modes needs $\int_0^1 L(\rho)\, d\rho = \infty$, which the optimum of Theorem 1(iii) allows: its $L(\rho) = \max\{1/\rho, 1/(1-\rho)\}$ grows like $1/\rho$ at the start, where the mode is chosen. A trained network is smooth, with finite $L(\rho)$ on the states it visits, so it can only approximate this split, and that is where it averages modes; by the gap bound, it needs starts that are spread out, i.e. start noise.

Under (A3) no two demonstrations share a state before $\rho = 1$, and modes separate only through nearby starts that $\mathcal{T}$ sends to different goals, which is where the $1/\rho$ growth occurs. Demonstrations in general can share a state later, as on the shared trunk of App. A.3.

Lemma 2 (A split after a shared state must come from the input)**.** *Suppose the demonstrations at state $\mathbf{x}_*$ and phase ρ_* leave it along two constant-velocity branches $\mathbf{x}_* + (\rho - \rho_*)\mathbf{u}_k$, $k = 1, 2$, with $\mathbf{u}_1 \neq \mathbf{u}_2$ and weights $w_1, w_2 > 0$. Then:*

- *(a) the regression target $\bar{v}(\mathbf{x}_*, \rho_*)$ is the average $(w_1\mathbf{u}_1 + w_2\mathbf{u}_2)/(w_1 + w_2)$;*
- *(b) any policy whose rollouts follow both branches has $L(\rho) \geq 1/(\rho - \rho_*)$ for $\rho > \rho_*$, which is not integrable at ρ_*;*
- *(c) a deterministic policy with unique rollouts carries the state $\mathbf{x}_*$ at phase ρ_* along a single path, so it follows at most one branch; reaching both requires rollouts, and hence demonstrations, that differ before ρ_*.*

Proof. (a) is the conditional mean over the demonstrations at $(\mathbf{x}_*, \rho_*)$. (b) At phase ρ the two branches are $(\rho - \rho_*)\|\mathbf{u}_1 - \mathbf{u}_2\|$ apart and their velocities differ by $\|\mathbf{u}_1 - \mathbf{u}_2\|$, so the ratio is exactly $1/(\rho - \rho_*)$, and $\int_{\rho_*}^{\rho_1} d\rho/(\rho - \rho_*) = \infty$. (c) A deterministic rollout is a map from the state at ρ_* to later states. □

The $1/\rho$ bound of Theorem 1(iii) is the limiting case $\rho_* = 0$. In App. A.3 the demonstrations coincide until $\rho \approx 0.15$, so $\rho_* > 0$: the regression target at the fork is the average of the two modes, the rollouts collide, and start-state variation, which separates the demonstrations before the fork, removes the failure.

A.1.5 Scope

Parts (i)–(iii) of Theorem 1, apart from the formula for $L(\rho)$, extend to Riemannian manifolds with geodesics in place of straight lines (McCann, 2001; Chen & Lipman, 2024); on SE(2) and SE(3) these are geodesics of a Riemannian metric, which generally differ from group-exponential curves because $\mathrm{SE}(n)$ has no bi-invariant metric. Proposition 2 needs neither (A1) nor (A3), so it covers the nonholonomic navigation task (for $\dot{\mathbf{x}} = B(\mathbf{x})\mathbf{u}$ with orthonormal columns, as for the unicycle, the squared loss on twists equals that on state velocities), but not states moved through contact, such as the Push-T block, whose velocity is not linear in the action. Neither result extends to L_1: an L_1 control cost does not determine the demonstrated velocity, being invariant to reparametrization, and an L_1 loss learns the conditional median, which need not transport the demonstrations.

A.1.6 Proof of Theorem 3, and Properties of the Score Penalty

Proof. Both losses average over the demonstrations at each $(\mathbf{x}, \rho)$, and $\mathbf{s}$ depends only on $(\mathbf{x}, \rho)$, so $\pi_\lambda(\mathbf{x}, \rho)$ minimizes $J(\mathbf{v}) = \mathbb{E}\|\mathbf{v} - \mathbf{u}_\rho\|^2 + \lambda\, \mathbb{E}|(\mathbf{u}_\rho - \mathbf{v}) \cdot \mathbf{s}|$ over those demonstrations. Where $\mathbf{s} = 0$, J is the squared loss alone. Otherwise, writing $\mathbf{v} = a\hat{\mathbf{s}} + \mathbf{v}_\perp$ and $\mathbf{u}_\rho = b\hat{\mathbf{s}} + \mathbf{u}_\perp$ splits J into $\mathbb{E}\|\mathbf{v}_\perp - \mathbf{u}_\perp\|^2$, which is minimized by $\mathbb{E}\,\mathbf{u}_\perp$ and gives (i), and $g(a) = \mathbb{E}(a-b)^2 + \lambda\|\mathbf{s}\|\,\mathbb{E}|a-b|$. The function g is strictly convex, with subgradient $2(a - \bar{b}) + \lambda\|\mathbf{s}\|\, h(a)$, where $h(a)$ is the subgradient of $\mathbb{E}|a - b|$. Where b has no atom at a, $h(a) = \Pr(b < a) - \Pr(b > a)$, which gives the stationarity condition $\mathbf{v} = \bar{v} + \frac{\lambda}{2}\, \mathbb{E}\big[\mathrm{sgn}\big((\mathbf{u}_\rho - \mathbf{v}) \cdot \mathbf{s}\big)\big]\, \mathbf{s}$. In general, $h(a) \subseteq [0, 1]$ above the smallest median of b, $h(a) \subseteq [-1, 0]$ below the largest, and $0 \in h(a)$ exactly at the medians. At a_λ the subgradient contains 0, so $|a_\lambda - \bar{b}| \leq \lambda\|\mathbf{s}\|/2$. Because $2(a - \bar{b})$ has the sign of $a - \bar{b}$, the subgradient is negative when a is below both $\bar{b}$ and the largest median, and positive when a is above both $\bar{b}$ and the smallest median, so a_λ lies between $\bar{b}$ and m. As $\lambda\|\mathbf{s}\| \to \infty$, the minimizers of $g/(\lambda\|\mathbf{s}\|)$ approach the medians of b, and the quadratic term selects the one closest to $\bar{b}$; this proves (ii). Finally, 0 is a

subgradient at $a = \bar{b}$, i.e. $a_\lambda = \bar{b}$, exactly when $\bar{b}$ is a median of b, which holds if b is constant or symmetric about $\bar{b}$; with (i) this gives (iii). □

Size and direction of the shift. Where b has no atom at a_λ, the stationarity condition reads $a_\lambda = \bar{b} + \frac{\lambda\|\mathbf{s}\|}{2}\big(\Pr(b > a_\lambda) - \Pr(b < a_\lambda)\big)$: the prediction moves in proportion to how unevenly the demonstrations split around it, toward the side with more of them. The penalty depends on $\mathbf{s}$ only through its axis, since replacing $\mathbf{s}$ by $-\mathbf{s}$ leaves $\mathcal{L}_{reg}$ unchanged, so it cannot prefer moving up the density over moving down it; the direction of the shift is set by where the median lies relative to the mean. With velocities ± 1 along the score, equally likely, $a_\lambda = 0$ for every λ, so an exact symmetric tie is not broken. With velocity 0 with probability $\frac{3}{4}$ and 4 with probability $\frac{1}{4}$, and $\lambda\|\mathbf{s}\| = 2$, the mean is 1, the median is 0 and $a_\lambda = \frac{1}{2}$, a shift against $\mathbf{s}$; reflecting the law gives a shift along $\mathbf{s}$.

Bias. If a third of the demonstrations at a state move along $\hat{\mathbf{s}}$ at speed 3 and the rest do not, then $\bar{b} = 1$, $m = 0$ and $a_\lambda = \max\{1 - \lambda\|\mathbf{s}\|/6,\, 0\}$, so for $\lambda\|\mathbf{s}\| \geq 6$ the regularized policy ignores the moving third. More generally, $\pi_\lambda \neq \bar{v}$ wherever $\mathbf{s} \neq 0$ and $\bar{b}$ is not a median of b, so the rollouts of π_λ need not reproduce p_ρ (Proposition 2), and minority behavior can be suppressed.

Realizability. These pointwise laws arise from genuine demonstration distributions. At an interior phase ρ_*, draw $\mathbf{x}_{\rho_*} = Z \sim \mathcal{N}(1, 1)$ independently of the velocity u and set $\mathbf{x}_\rho = Z + (\rho - \rho_*)u$. The demonstrations at $x = 0$ then have the distribution of u, and the score of p_{ρ_*} at $x = 0$ is 1, or $1/(1 + h^2)$ for a Gaussian kernel estimate with bandwidth h.

A.2 KDE Score Calculation

The kernel score $\nabla \log \hat{p}_\rho$ is the gradient of a Gaussian kernel density estimate, built at each phase from the demonstrations at that phase and expressed in each pose's body tangent space through the log map,

$$\hat{p}_\rho(\mathbf{x}) = \frac{1}{M}\sum_{j=1}^{M} \mathcal{N}\big(\mathbf{x}; \mathbf{x}_\rho^{(j)}, h^2 I\big), \qquad \nabla_{\mathbf{x}} \log \hat{p}_\rho(\mathbf{x}) = \frac{1}{h^2}\sum_{j=1}^{M} w_j(\mathbf{x})\big(\mathbf{x}_\rho^{(j)} - \mathbf{x}\big), \tag{13}$$

with $w_j(\mathbf{x}) \propto \exp\big(-\|\mathbf{x}_\rho^{(j)} - \mathbf{x}\|^2/2h^2\big)$ normalized to sum to one.

Push-T. For Push-T the kernel score is computed over the pusher position only, in pixel coordinates ($\mathbb{R}^2$, bandwidth $h = 32$ px), from the training episodes' pusher positions at each point of a 128-point phase grid. It is a direction in the same space as the pusher phase velocity that the policy predicts, and it depends only on inputs of the policy, as Theorem 3 requires.

Bandwidth. A fixed bandwidth attenuates the score where the demonstrations are narrow: for a Gaussian marginal of spread σ, the kernel score is $\sigma^2/(\sigma^2 + h^2)$ times the true score (App. A.1.6, Realizability). The score enters $\mathcal{L}_{reg}$ only through its axis and the weight $\lambda\|\mathbf{s}\|$ (Theorem 3), so for a locally Gaussian marginal this attenuation lowers the effective weight in narrow regions without changing the axis of the correction. This is why we accept a fixed-bandwidth KDE for the score but not for the divergence targets of App. A.5, which enter the loss as regression targets.

Extending the Score to $\mathrm{SE}(3)$. For M demonstrations and N phase steps it costs $\mathcal{O}(NM^2)$ pairwise log-map evaluations. It is computed once before training, and no policy needs it at inference. The estimator carries over to $\mathrm{SE}(3)$ unchanged, with six-dimensional twists in place of three-dimensional ones. A direct implementation costs a small constant factor more per pair: 1.4–1.7× on a GPU and 2.6–3.8× on a CPU (Figure 8). The larger cost is statistical. Even with the best bandwidth, the mean squared error of a kernel estimate of a density gradient decays as $M^{-4/(D+6)}$ in D dimensions (Chacón et al., 2011). Matching the accuracy that 10^3 demonstrations give on $\mathrm{SE}(2)$ ($D = 3$) therefore takes on the order of 10^4 on $\mathrm{SE}(3)$ ($D = 6$), and the number of pairs grows about a hundredfold. Even so, the precompute stays under a second on a single GPU (Figure 8). How much of this statistical cost applies depends on how many of the added degrees of freedom the demonstrations actually vary in. It can be reduced by summing only over the k nearest neighbors, which brings the kernel evaluations down to $\mathcal{O}(NMk)$, or by replacing the kernel estimate with a learned score model.

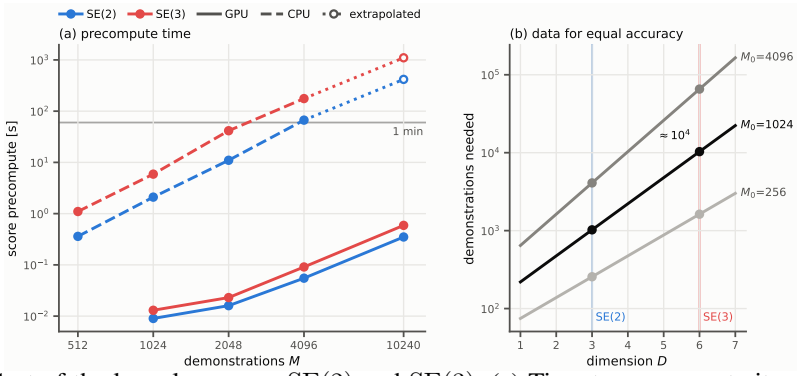

Figure 8: Cost of the kernel score on $\mathrm{SE}(2)$ and $\mathrm{SE}(3)$. (a) Time to precompute it over 51 phase steps against the number of demonstrations M (GPU: RTX 4090, float32, fused kernels; CPU: 16 threads, float64). Extrapolated points use the per-pair cost at $M = 4096$ and are lower bounds. (b) Demonstrations needed in D dimensions to match the accuracy of M_0 demonstrations on $\mathrm{SE}(2)$, $M_0^{(D+6)/9}$, from the $M^{-4/(D+6)}$ rate of kernel density gradient estimates (Chacón et al., 2011); an order of magnitude, not a prediction.

A.3 Mode Collapsing Under Fully Deterministic Training Data

To test whether the policies need stochasticity in their training data, we rebuild the dual-path scenario (C) as deterministic point-to-point planning. The start pose $(5.0, 0.9, \pi/2)$ and the goal pose $(5.0, 7.0, \pi/2)$ are fixed, and the planner's via points are not jittered. The planner then returns exactly one trajectory per homotopy class: the $n = 1024$ demonstrations are 512 copies of a left path and 512 copies of its mirror image on the right. We compare this set against four others. *+via* keeps the single start but restores C's via-point jitter ($\sigma = 0.3$ m), so the demonstrations spread along the middle of the path but share both endpoints. The *start-noise mixtures* replace 5, 10 or 15% of the demonstrations with demonstrations planned from C's start distribution ($\sigma = 8$ cm, 0.05 rad; no via jitter, same goal pose). *+start* uses that start distribution for every demonstration without via-point jitter, and C, the original scenario, combines it with via-point jitter. Each set trains the four MLP arms (MLP, MLP+score, MLP+phase and MLP+phase+score, $\lambda = 0.5$) over 10 seeds. We evaluate every policy two ways: from the single nominal start, where a deterministic policy drives one path per seed, and from C's 1024 held-out starts. A rollout This replaces the $3\sigma_g$ criterion of App. A.7, which gives zero tolerance for a fixed goal pose; all numbers in this section, including those for the original C, use the 0.6 m criterion. (Figs. 9 and 10).

Trained on a single start, every arm fails. No seed reaches the goal from the nominal start, and success from C's held-out starts is 0%, so spreading the starts at evaluation does not help either. The two demonstrations coincide until phase ≈ 0.15 and then cross within 9 mm of each other with opposite turning controls. The regression target there is the average of the two modes, and the rollouts either drive straight at the block (20–80% of seeds, depending on the arm) or commit to a side too late and collide. The failure is not a lack of coverage along the paths: *+via* spreads the demonstrations by up to 0.16 m mid-path and still fails in every seed, with 80–100% of seeds driving into the block. Nor is it a tracking failure: every arm placed exactly on a demonstration at phase ≥ 0.3 follows it to the goal in 95–100% of restarts. The score regularizer cannot help here either. Up to and including the fork, every demonstration sits at the same state, so the kernel score vanishes; past the fork, the demonstrations at any given state and phase all belong to one branch and agree. In both cases Theorem 3 leaves the regression optimum at $\bar{v}$, which at the fork is the average of the two branches (Lemma 2(a)).

A small amount of start noise removes the failure. With 5% noisy-start demonstrations (52 of 1024), MLP, MLP+score and MLP+phase reach the goal from the nominal start in 8–9 of 10 seeds and succeed in 80–88% of C's held-out starts. At 10–15%, every arm succeeds in 96–99% of C's starts, above both the all-noisy *+start* set (MLP 60%, MLP+phase 79%) and the original C runs (63% and 72%). A deterministic policy can express two modes only through its input: variation in the demonstrations' start states lets it assign starts to modes. Once it has learned that split, it resolves even the dead-center start that it cannot solve when trained on a single start. The score regularizer's effect depends on how much of the data is noisy. At 5%, adding it to MLP+phase hurts (66% vs. 88% success from C's starts, $p = 0.049$, Welch over seeds). This matches the bias of Theorem 3: where

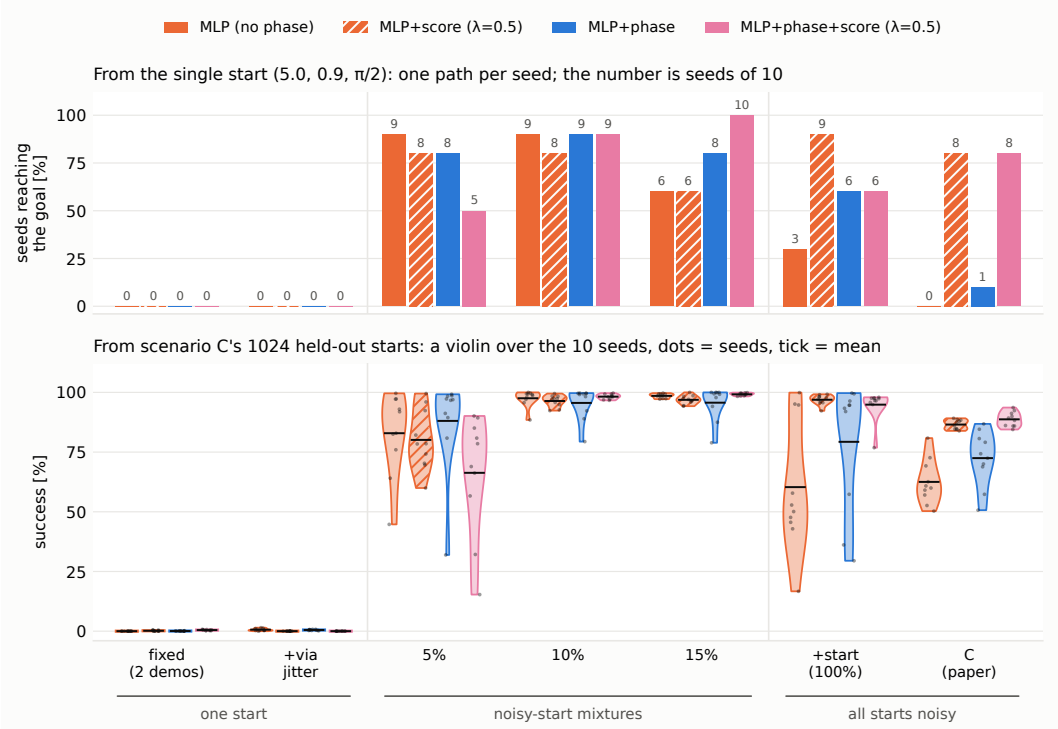

Figure 9: Success of the four MLP arms on the dual-path scenario for every training set, left to right: a single start pose (*fixed*: two distinct demonstrations; *+via*: via-point jitter), mixtures in which 5, 10 or 15% of the demonstrations start from C's start distribution, and training sets in which every start is noisy (*+start*; C, the original scenario). **Top**: share of the 10 seeds whose single deterministic rollout from the nominal start $(5.0, 0.9, \pi/2)$ reaches the goal without a collision; the number above each bar is seeds out of 10. **Bottom**: success from C's 1024 held-out starts, a violin over the 10 seeds (dots: seeds; black tick: mean). Seeds that all agree, as on the one-start sets, are drawn as a flat tick. $n = 1024$ demonstrations, score weight $\lambda = 0.5$.

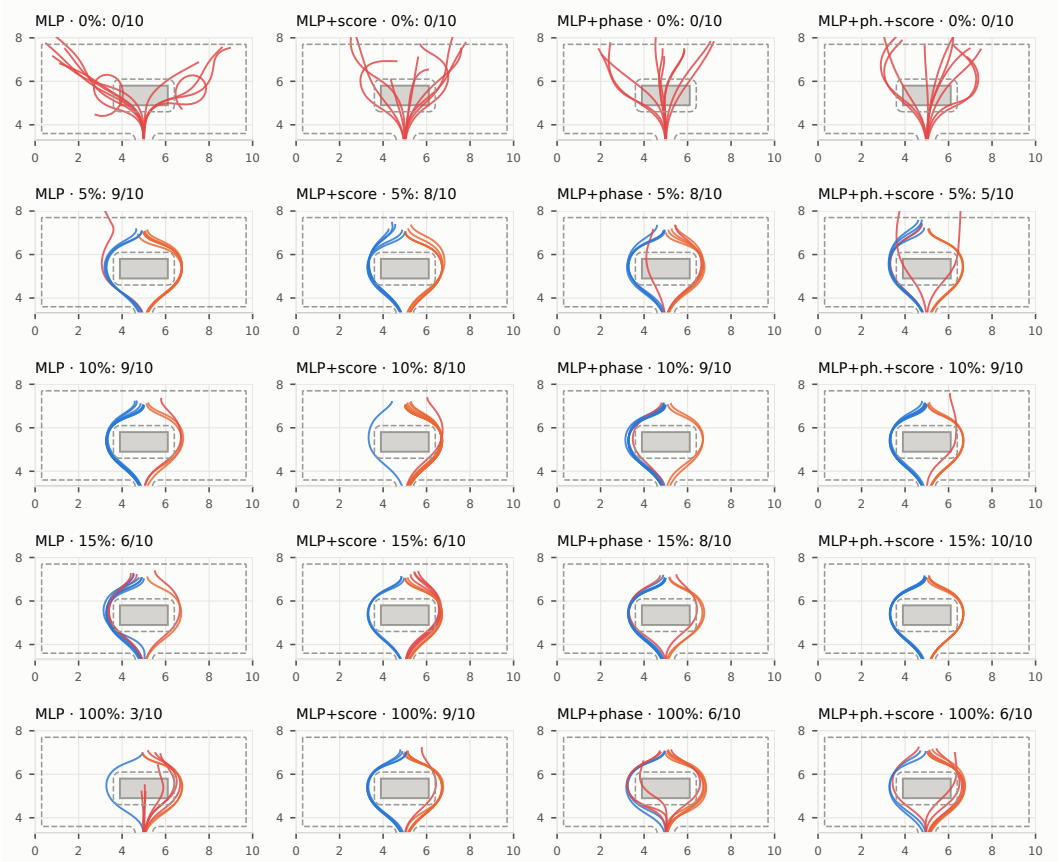

Figure 10: Rollouts from the single nominal start $(5.0, 0.9, \pi/2)$, one path per seed (10 seeds per panel), for each arm (columns) and each share of noisy-start demonstrations in training (rows: 0%, the fully deterministic set, through 100%). Blue and orange paths pass the block on the left and right; red paths collide. The count in each title is the number of seeds that reach the goal without a collision. Trained on a single start (top row), every seed averages the two modes at the fork and hits the block or a wall; 5% noisy-start demonstrations already give 5–9 successful seeds of 10.

the 52 varied-start demonstrations overlap the 972 duplicates, the median velocity follows the duplicates, so the regularizer suppresses exactly the minority that carries the start-to-mode assignment. At 10–15% MLP+phase+score is the most consistent arm (98–99% $\pm$ 1, and all 10 seeds reach the goal from the nominal start at 15%), though its gain over MLP+phase is not significant. When every start is noisy it matters most: MLP+score reaches 97% where the MLP reaches 60% ($p = 0.002$). Deterministic behavior cloning, with or without phase and score, therefore depends on stochasticity in the training starts, and 5–10% of the demonstrations is enough to supply it.

A.4 Planner Details

The planner solves for the optimal trajectory using Levenberg–Marquardt optimization with the following objective:

$$\begin{aligned}\min_{\{T_i\}_0^N,\{u_i\}_0^{N-1}} \tfrac{1}{2}\Big[w_d \sum_{i=0}^{N-1} \big\|\mathrm{Log}(T_i^{-1}T_{i+1}) - \Delta t\,[v_i,\,0,\,\omega_i]^\top\big\|^2 \\ + w_c \sum_{i=1}^{N} \max\big(0,\ \epsilon - d(\mathbf{p}_i)\big)^2 + w_l \sum_i \|h(u_i)\|^2 + w_s \sum_{i=0}^{N-2} \|u_{i+1}-u_i\|^2 \\ + w_e \sum_i \|u_i\|^2 + w_b\big(\|\mathrm{Log}(T_s^{-1}T_0)\|^2 + \|\mathbf{p}_N - \mathbf{p}_g\|^2\big)\Big] \quad (14)\end{aligned}$$

Weights were configured as $w_d = 2000$, $w_c = 60$, $w_l = 60$, $w_s = 2$, $w_e = 0.05$, and $w_b = 10^4$. The collision margin ϵ is 0.45 m (comprising a 0.3 m robot radius and 0.15 m clearance), and d is a 0.1 m grid signed distance field (SDF) of the obstacles. The control penalty h acts as a hinge on the velocity limits $v \in [0, 2]$ m/s and $|\omega| \le 2$ rad/s, and lateral slip is strictly prevented by the zero in the twist vector $[v, 0, \omega]^\top$.

A.5 Divergence Estimation Challenges and Regularization Ablation

We found that divergence-based regularization hurt model performance rather than improving it, while score-based regularization significantly improved it (Fig. 11). We theorize this is because divergence is difficult to compute from sparse training data, especially on higher dimensional manifolds like SE(2) and SE(3). Huang et al. (2025) regularize flow matching with conditional divergence matching (CDM), which penalizes the mismatch between the divergence of the conditional target flow and that of the learned field, together with a score-alignment term. Their target divergence is available in closed form, which is not the case for demonstration data. In our ablations, we tested three versions of this loss:

$$\mathcal{L} = \mathcal{L}_{\mathrm{BC}} + \lambda\,\mathcal{L}_{reg}, \quad \mathcal{L}_{reg} \in \{\mathbb{E}|e_{div} + e_{score}|\ (\mathrm{CDM}),\ \mathbb{E}|e_{div}|\ (\mathrm{div}),\ \mathbb{E}|e_{score}|\ (\mathrm{score})\} \quad (15)$$

where $e_{div} = \widehat{\nabla\cdot\mathbf{u}}_\rho(\mathbf{x}_\rho) - \nabla\cdot\pi_\phi(\mathbf{x}_\rho,\rho)$, with $\widehat{\nabla\cdot\mathbf{u}}_\rho$ the kNN estimate of the demonstrations' divergence described below, and $e_{score} = \big(\mathbf{u}_\rho - \pi_\phi(\mathbf{x}_\rho,\rho)\big)\cdot\nabla_{\mathbf{x}}\log\hat{p}_\rho(\mathbf{x}_\rho)$ is the integrand of Eq. 6 (the units align without rescaling: twists are in m/s and rad/s, divergence in 1/s, and the score in 1/m and 1/rad).

The demonstration data clearly shows that the score (Figs. 12 and 13) and the divergence (Fig. 12) evolve over phase. While accurately estimating these quantities improves Flow Matching (FM) performance (Huang et al., 2025), obtaining an unbiased divergence estimate—which measures the rate of particle expansion and contraction in the flow field (Batchelor, 2000)—proved challenging.

For the ***demonstration data***, we locally approximated divergence by tracking the distance r_k to the 16th nearest neighbor (kNN) among all demonstrations at the same phase step. Distances were measured using a left-invariant $SE(2)$ metric, defined as $d = |\mathrm{Log}(T_a^{-1}T_b)|$ at 1 m/rad. Empirically, this kNN approach significantly outperformed a fixed-bandwidth Kernel Density Estimator (KDE) by avoiding the attenuation of narrow distributions. We estimated the log-volume derivative as $\nabla\cdot u \approx D\,\frac{d}{dt}\log r_k$ (with $D = 3$), computing the derivative via a Savitzky–Golay filter (Savitzky & Golay, 1964). The filter used a 9-sample (1.35 s) window and a 2nd-order polynomial, evaluated at

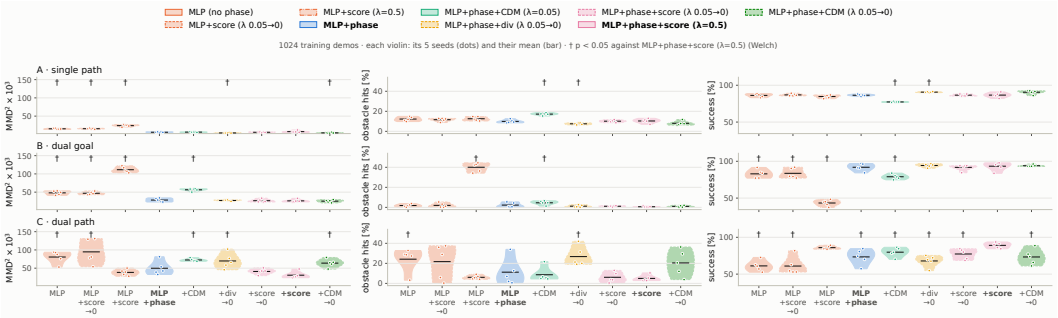

Figure 11: Regularizer ablation on the three navigation scenarios (1024 training demonstrations, 5 seeds per arm; † marks $p < 0.05$ against MLP+phase+score, $\lambda = 0.5$, Welch over seeds). Score-based regularizers are the only ones that improve performance significantly; divergence-based regularizers are neutral or significantly hurt performance. Labels such as λ $0.05 \to 0$ denote a weight annealed to zero over training.

$dt = 0.15$ s to match the planner's 6.67 Hz control step. For the demonstration score, the Gaussian KDE bandwidth was set to $h = 0.2$, calculated in each pose's body tangent space.

Conversely, for the ***policy divergence***, $\nabla \cdot \pi_\phi(\mathbf{x}_\rho, \rho)$, we avoided stochastic trace estimators (Grathwohl et al., 2018; Hutchinson, 1989) altogether. We efficiently computed the exact trace using reverse-mode vector-Jacobian products, summing the partial derivatives $\cos\theta\,\partial_x v + \sin\theta\,\partial_y v + \partial_\theta \omega$ evaluated at a zero perturbation $T\,\mathrm{Exp}(\epsilon)$.

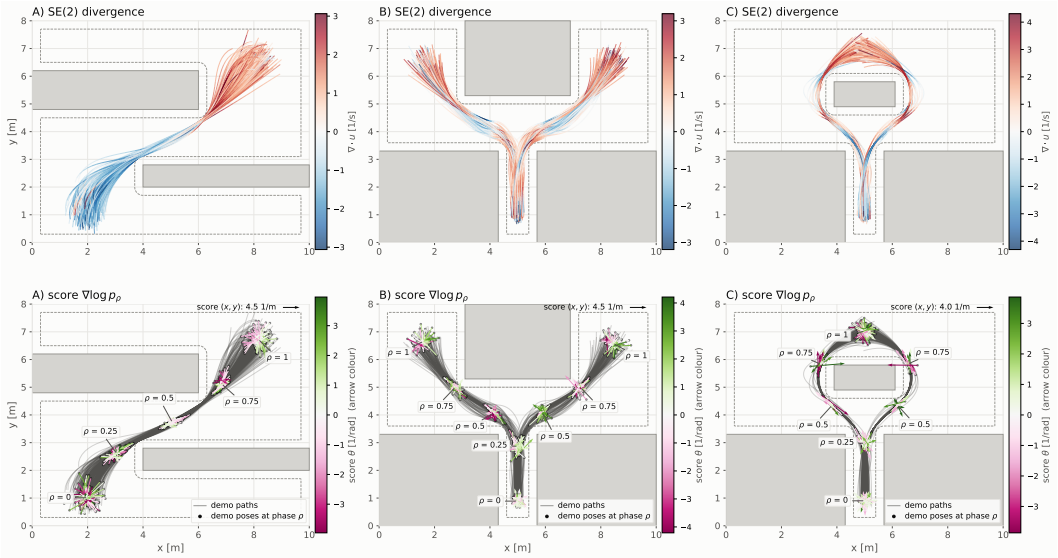

Figure 12: Estimated SE(2) divergence (top; kNN, $k = 16$) and kernel score (bottom; Gaussian KDE, $h = 0.2$) of the robot navigation demonstrations for scenarios A, B, and C, illustrating how the flow and density evolve over phase.

Both the ***kNN and KDE divergence estimators*** derive from the continuity equation (Batchelor, 2000). Along any particle trajectory x_t in a flow u, the divergence equals the negative rate of change of the log-density: $\nabla \cdot u(x_t, t) = -\frac{d}{dt} \log p(x_t, t)$. We compare two density estimates evaluated across demonstrations at the same phase step, using a left-invariant SE(2) metric ($d(T_a, T_b) = \|\mathrm{Log}(T_a^{-1} T_b)\|$, at 1 m/rad):

- **KDE**: A fixed-bandwidth Gaussian kernel density estimate (Parzen, 1962) (leave-one-out), yielding $\widehat{\nabla \cdot u} = -\frac{d}{dt} \log \hat{p}_h$.
- **kNN**: The k-nearest-neighbor density (Loftsgaarden & Quesenberry, 1965) $\hat{p} \propto k/(M r_k^D)$, where r_k is the distance to the k-th neighbor and $D = 3$. This yields $\widehat{\nabla \cdot u} = D\,\frac{d}{dt} \log r_k$.

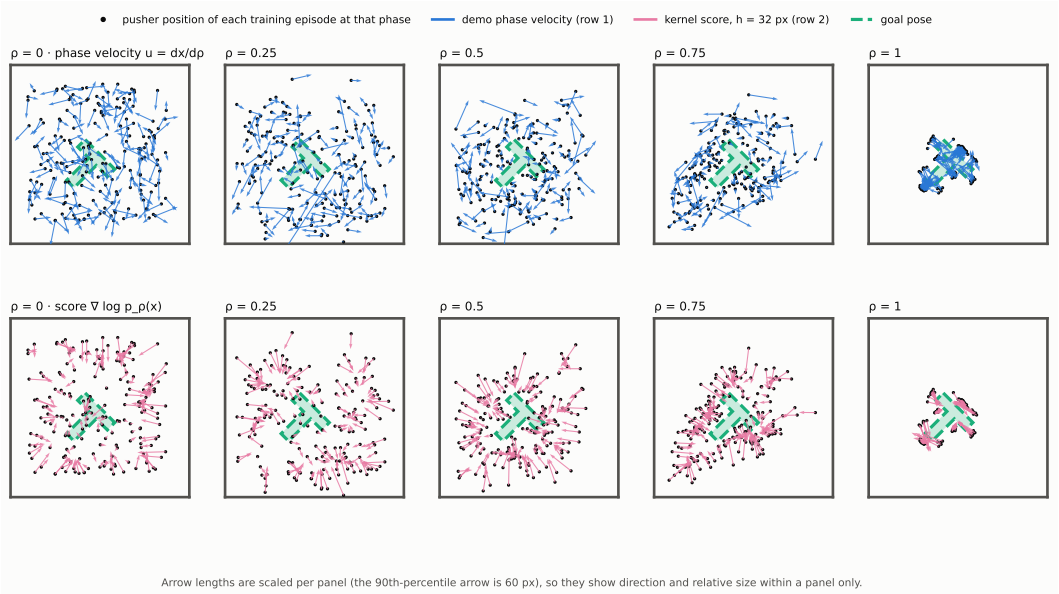

Figure 13: Demonstration phase velocities and corresponding kernel scores for the Push-T dataset. Snapshots at distinct temporal phases ($\rho \in \{0, 0.25, 0.5, 0.75, 1\}$) show the local modes of the spatial pusher distribution converging toward the final goal pose.

Both apply the same Savitzky–Golay derivative (Savitzky & Golay, 1964) (window 9, order 2), isolating the density estimator as the sole variable.

Synthetic flows of known divergence. To assess accuracy against ground truth, we generate SE(2) particle flows that are linear in exponential coordinates: $T_i(t) = c\,\mathrm{Exp}(\xi_i(t))$ with $\xi_i(t) = e^{Rt}\xi_i(0)$, where $R = \mathrm{diag}(r_x, r_y, r_\theta)$. The divergence of this flow is exactly $\mathrm{tr}\,R$ at every particle and time, neglecting a minor $O(\theta^2/12)$ Haar-measure correction (Chirikjian, 2011). We use $M = 1024$ particles with initial spreads $(0.3\,\mathrm{m},\ 0.3\,\mathrm{m},\ 0.15\,\mathrm{rad})$, matching the start distribution of scenario A.

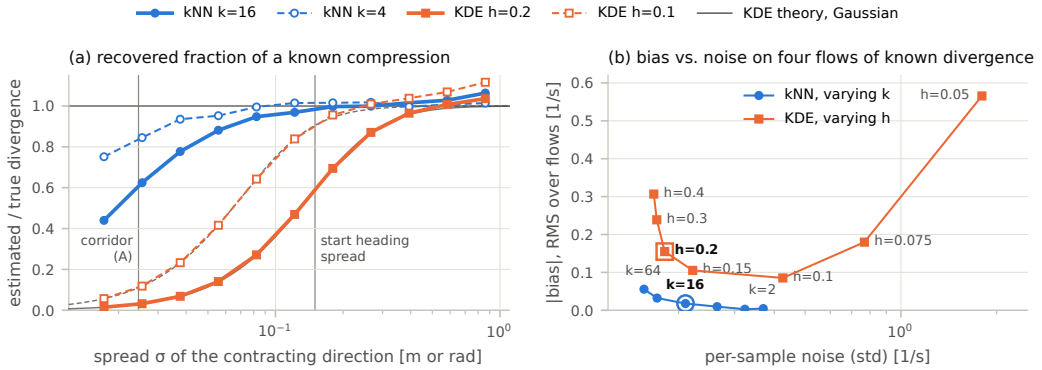

Figure 14: Divergence estimators on synthetic SE(2) flows of known divergence. (a) Fraction of a known compression recovered, against the spread of the contracting direction. Gray lines show the closed-form attenuation of a Gaussian KDE; vertical lines mark scenario A's narrowest corridor and its start-heading spread. (b) RMS bias over four flows against per-sample noise, sweeping k (kNN) and h (KDE); the circled points are the settings shown in (a) and in Fig. 15.

Fig. 14(a) tracks a single contracting direction against its spread σ. A fixed-bandwidth KDE mathematically attenuates the signal; for a Gaussian marginal, it recovers only a fraction $(1 + 2u^2)/(1 + u^2)^2$ of the true divergence, where $u = h/\sigma$. Our measured KDE curves perfectly match this closed-form attenuation. This shrinkage severely impacts our data: at scenario A's narrowest point (minor-axis spread $0.025\,\mathrm{m}$, at $\rho = 0.64$), KDE with $h = 0.2$ captures just 3% of the true compression. Conversely, kNN adapts its length scale to local sample spacing. With $k = 16$, kNN recovers 61%

at this narrowest point and reaches 95% by $\sigma = 0.08$ m (the start spread of scenarios B and C, where KDE $h = 0.2$ only recovers 26%).

Fig. 14(b) plots root-mean-square bias against per-sample noise over four flow types: isotropic expansion ($+1.2\,\mathrm{s}^{-1}$), y-compression (-0.4), heading compression (-0.4), and area-preserving strain (0). Every $k \in \{2, \ldots, 64\}$ yields a bias below $0.06\,\mathrm{s}^{-1}$, where increasing k smoothly trades slight bias for lower noise. KDE cannot match this: its lowest bias ($0.085\,\mathrm{s}^{-1}$ at $h = 0.1$) suffers twice the noise of kNN at $k = 16$. At $h = 0.2$, KDE bias jumps to $0.155\,\mathrm{s}^{-1}$ versus kNN's $0.018\,\mathrm{s}^{-1}$ for similar noise levels. Bias is a critical flaw for a regression target: while zero-mean noise averages out over a minibatch, bias permanently shifts the target.

We select $k = 16$ as the optimal balance. Smaller values ($k = 4$) reduce attenuation but increase per-sample noise by 50% (0.32 vs. $0.21\,\mathrm{s}^{-1}$). Furthermore, k must remain well below the per-mode demonstration count so that neighbors belong to the same mode; with $n = 1024$ demonstrations (512 per mode in B and C), $k = 16$ is far below that limit.

Estimates on the navigation datasets. Fig. 15 confirms these behaviors on the navigation datasets. The top row tracks mean divergence over phase, while the bottom row shows the spread across demonstrations within a mode. Both kNN settings agree closely, whereas KDE estimates artificially flatten toward zero.

This attenuation is most severe in scenarios B and C, where the initial distribution (0.08 m, 0.05 rad) is much narrower than the KDE kernel. In scenario C ($\rho = 0.3$), kNN detects a strong compression ($-2.2\,\mathrm{s}^{-1}$) that KDE $h = 0.2$ almost entirely misses ($+0.07\,\mathrm{s}^{-1}$). At $\rho = 0.2$, the estimators even disagree on sign (-0.86 vs. $+0.77\,\mathrm{s}^{-1}$). KDE's artificially small spread across demonstrations reflects this shrinkage toward zero, not statistical precision. Relying on the synthetic ground-truth benchmark, we confidently adopt kNN ($k = 16$) for all divergence targets.

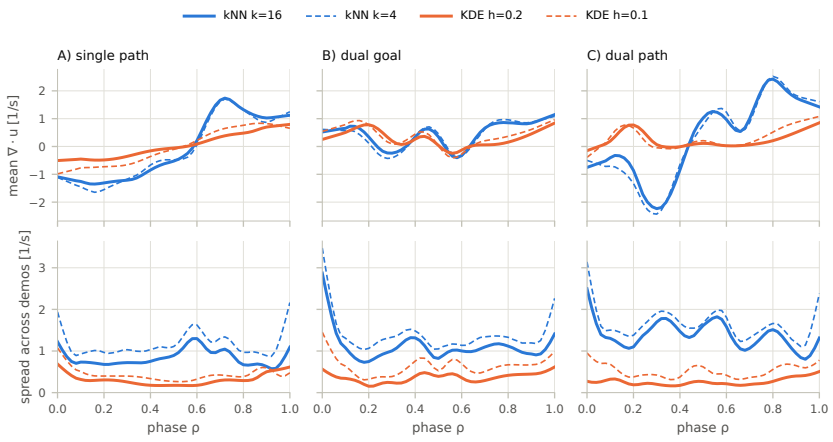

Figure 15: Divergence estimators on the navigation demonstrations. Top: mean demonstration divergence over phase (per-mode means, averaged). Bottom: spread across demonstrations within a mode. KDE estimates are flattened toward zero wherever the demonstration distribution is narrower than the kernel, most visibly in B and C.

Even with the lower-bias kNN targets, divergence regularization failed to improve model performance and often degraded it. The main difference between our divergence regularization, which fails, and that of Huang et al. (2025), which succeeds, is how the target is obtained (kNN estimation versus a closed-form expression), so we hypothesize that the per-sample noise of the estimated targets degrades the supervisory signal enough to hurt performance. We therefore regularize our models with the score term only, which significantly improved performance (Fig. 11).

A.6 Policy, Task, and Normalization Details

Robot Navigation: Demonstration trajectories consist of $N = 50$ steps with a time step of $\Delta t = 0.15$ s, yielding a 7.5 s horizon. The policy network is a 3×256 SiLU MLP (133,634 parameters). The input is formulated as $[(x - c_x)/s_x,\ (y - c_y)/s_y,\ \cos\theta,\ \sin\theta,\ 2\rho - 1]$. Positions are normalized

to $[-1, 1]$ using a fixed affine map based on a 10×8 m workspace center $(c_x, c_y) = (5, 4)$ m and half-extents $(s_x, s_y) = (5, 4)$ m. Because this scaling is tied to the physical workspace, it remains consistent across all scenarios and seeds. Heading is parameterized as $(\cos\theta, \sin\theta)$ to avoid discontinuities at $\theta = \pm\pi$. The network outputs $[v, \omega]$ remain unscaled in m/s and rad/s, allowing the policy divergence $\nabla\cdot\,\pi_\phi$ to be evaluated directly in s^{-1} to match the divergence targets of App. A.5.

Push-T: Demonstrations consist of Diffusion Policy's 206 human Push-T episodes (Chi et al., 2024) (25,650 steps recorded at 10 Hz, $\Delta t = 0.1$ s; episode lengths 49–246 steps, median 123), split 90/10 into 185 training and 21 held-out episodes. The state comprises the end-effector $\mathbf{x}_{ee} = [x_{ee}\ y_{ee}]$ and the object $\mathbf{x}_{obj} = [x_{obj}\ y_{obj}\ \theta_{obj}]$, min-max normalized over the training set, with θ_{obj} passed as $[\cos\theta_{obj}\ \sin\theta_{obj}]$.

The MLP backbone is a 4×512 SiLU network (793,090 parameters) taking the concatenated input $[\mathbf{x}_{obj}\ \mathbf{x}_{ee}\ 2\rho - 1]$. The TF backbone (907,522 parameters) stacks two blocks of width 128 with four heads. Each block applies self-attention, cross-attention, and a feed-forward layer, each preceded by an AdaLN layer whose shift, scale, and gate come from a sinusoidal embedding of ρ (Fig. 1). $\mathbf{x}_{ee}$ and $\mathbf{x}_{obj}$ are embedded as separate tokens that a single learned query reads via cross-attention, and an AdaLN-modulated linear decoder maps the query to the action.

Both networks are trained via MSE against the demonstrations' forward difference $\mathbf{u}_k = (L - 1)(\mathbf{x}_{ee,k+1} - \mathbf{x}_{ee,k})$ for an episode of L steps, i.e. the pusher's phase velocity with $\rho_k = k/(L-1)$. During rollouts, the commanded displacement is executed through the inverse of the environment's PD controller and clipped to the simulator's $[0, 512]^2$ limits; App. A.7 gives the phase increment and integration step.

Phase estimator. $\mathrm{MLP}_{\mathrm{TP}}$ (Eq. 9) is a 2×512 ReLU MLP with a sigmoid output (0.27M parameters) trained with an MSE loss to predict each demonstration's normalized phase from the current state and the episode's initial block pose, one estimator per policy seed; its cost is included in all reported latencies. We fit it on all demonstrations, including the validation episodes. The estimator is a way to give the policy a task-progress signal, not a claim that the phase regressor generalizes, so we let it fit the demonstrations closely. Every policy is evaluated in closed-loop simulator rollouts, not against held-out demonstrations.

A.7 Training Details

Robot Navigation: We train for 20,000 steps using Adam (no weight decay) with minibatches of 2048 samples drawn uniformly with replacement, cosine-annealing the learning rate from 10^{-3} to 10^{-5}. The regularizer weight is fixed at $\lambda = 0.5$. Final weights are evaluated directly without checkpoint selection.

Rollouts are performed from 1024 held-out start poses. MMD is computed using a Gaussian kernel ($\sigma = 0.3$) on $[x,\ y,\ \cos\theta,\ \sin\theta]$ across all $N + 1$ steps. Collisions are registered if the exact interpolated arc between steps breaches the 0.3 m robot radius against obstacles or boundaries. A success is recorded if a collision-free rollout terminates within $3\sigma_g$ of a goal mean. Methods are compared against the FM baseline using Welch's t-test across seeds ($p < 0.05$).

Push-T: We follow Diffusion Policy's low-dimensional Push-T recipe (Chi et al., 2024) rather than the navigation step budget: 500 epochs over the 185 training episodes (23,034 windows, 90 minibatches of 256 per epoch, 45,000 steps) with AdamW (lr $= 10^{-4}$, $\beta = (0.95, 0.999)$, $\epsilon = 10^{-8}$, weight decay 10^{-6}), a cosine schedule with 500 warmup steps stepped every minibatch, and an exponential moving average of the weights (power 0.75, maximum decay 0.9999) updated every step. The regularizer weight is held constant at $\lambda = 2.5$ with a kernel bandwidth of 32 px; the score is precomputed once per training row as the kernel score of the training episodes' pusher positions at that row's phase on a 128-point phase grid. Both were selected on validation loss over $\lambda \in \{0.5, 2.5, 10\}$, anneal fraction $\in \{0.5, 1\}$ and bandwidth $\in \{16, 32, 64\}$ px, and $\lambda = 2.5$ was confirmed on the TF backbone by a rollout sweep over $\lambda \in \{0.5, 2.5, 5, 10, 25\}$. Unlike the navigation task, the evaluated weights are the EMA model at the lowest EMA validation loss; the validation loss is the phase-velocity MSE alone, so every arm is selected on the same quantity. Each arm is trained for 10 seeds.

Rollouts follow Diffusion Policy's test protocol: 50 environments per seed (seeds 100000–100049), at most 300 steps each, giving 500 episodes per arm. The oracle phase is unavailable at test time,

so ρ comes from the phase estimator (App. A.6): $\rho_t = \min\{1,\ \max(\rho_{t-1} + \varepsilon,\ \mathrm{MLP}_{\mathrm{TP}}(\mathcal{O}_t))\}$ with minimum increment $\varepsilon = 1/245$ (one step of the longest demonstration), while the integration step is held at the median demonstration's $\Delta\rho = 1/(L_{\mathrm{med}} - 1)$, $L_{\mathrm{med}} = 123$. The commanded displacement $\mathbf{u}\,\Delta\rho$ is executed through the inverse of the environment's PD controller ($k_p = 100$, $k_v = 20$) and clipped to the simulator's $[0, 512]^2$ action space. We report the episode score $\max_t \mathrm{clip}(\mathrm{coverage}_t/0.95,\ 0,\ 1)$, the success rate (final goal coverage above 0.95, an IoU of about 0.90 against the goal pose), the terminal position and orientation error of the block averaged within a seed, and the stall rate (the pusher moving less than 1 px over the last 50 steps). Methods are compared against the Phaseformer+score arm using Welch's t-test across seeds ($p < 0.05$).

Training Results: The training and evaluation losses as well as model selection epochs for robot navigation and for Push-T are shown in Figs. 16 and 17, respectively.

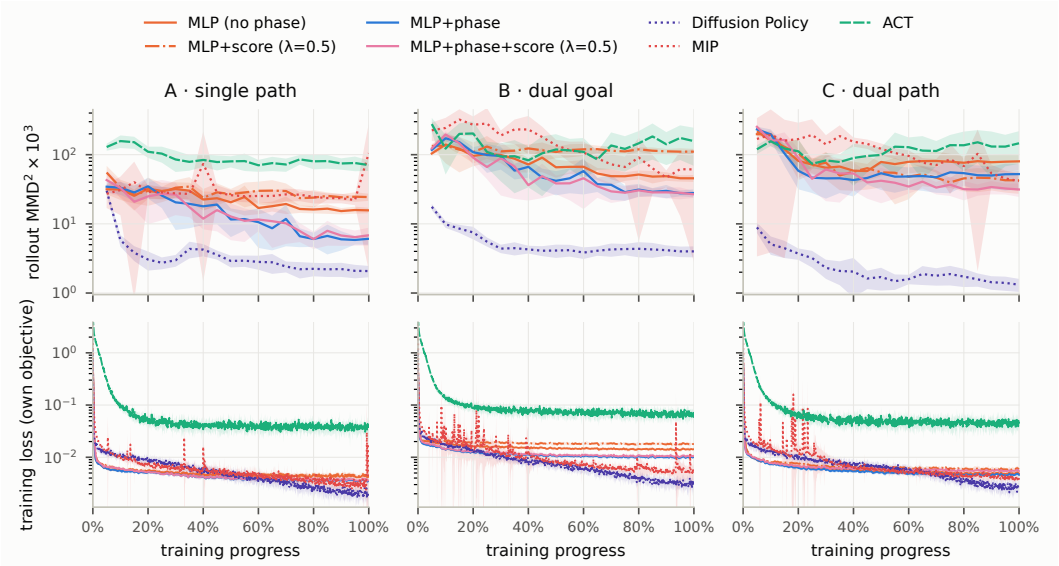

Figure 16: Robot Navigation Training: $n = 1024$ demonstrations, mean over 10 seeds (band: ± 1 s.d.). **Top**: squared MMD of the 1024 held-out rollouts against the held-out demonstrations, at the 20 evaluation points of each run's schedule. Log scale on both rows. **Bottom**: each model's own training objective: the arms' velocity MSE (the regression term alone, without the regularizer), Diffusion Policy denoising MSE, MIP regression MSE, ACT L1 + 10 KL. The four objectives are in different units and are not comparable by height; only their shape is. The arms train for 20k steps and the generative baselines for 100k, so both axes are the fraction of each model's own budget; the twenty evaluation points fall at the same fractions either way.

A.8 Additional Results

Figure 18 shows more evaluations of the robot navigation task, specifically the MMD^2, % obstacle hits, and % success of the deterministic MLP policies compared to the stochastic baselines.

A more detailed breakdown of model inference times is shown in Fig. 19.

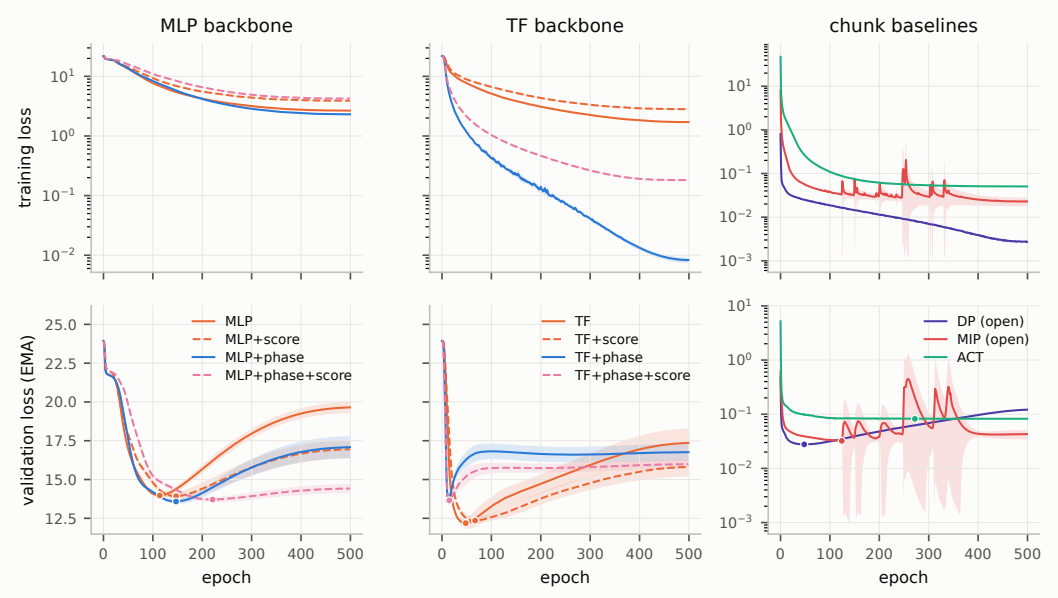

Figure 17: Push-T Training: 500 epochs of Diffusion Policy's low-dimensional recipe, mean over 10 seeds (band: ± 1 s.d.). **Top**: training loss. For the eight policy arms this is the phase-velocity MSE alone, so a score arm is comparable with the arm whose network it shares; for the chunk baselines it is their own objective (DP's denoising MSE, MIP's regression MSE, ACT's L1 + 10 KL), which is neither the arms' nor each other's. **Bottom**: the EMA model's validation loss, the quantity every arm's checkpoint is selected on; the dot marks the selected epoch. The training row is log scale, as is the baselines' validation panel; the arms' validation loss is linear. The two backbone columns share their axes, so an arm and its transformer twin are read straight across; the baselines keep their own.

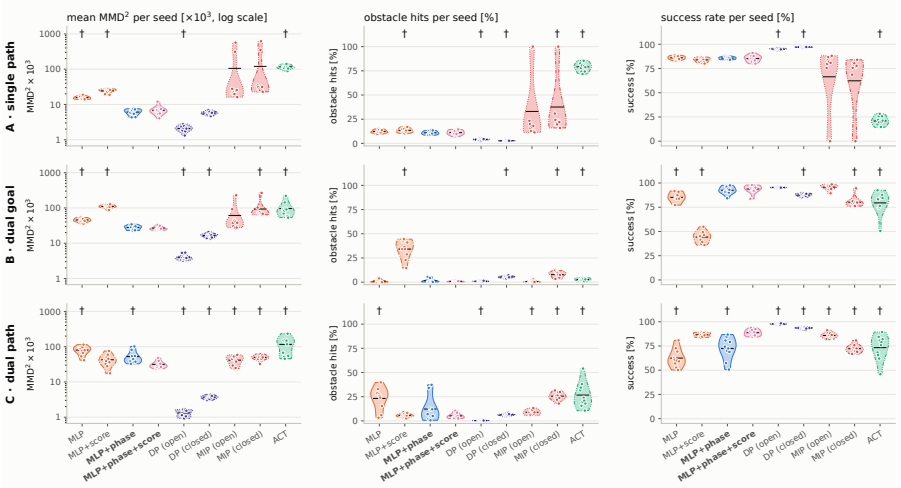

Figure 18: Robot Navigation Model Performance measured by MMD^2, % obstacle hits, and % success. 1024 training demos; each violin shows the model's 10 seeds (dots) and their mean (bar). Regularizer arms: $\lambda = 0.5$ held constant. The DP and MIP run open loop, as trained, and closed loop (temporal ensembling). ACT runs closed loop with temporal ensembling. † marks $p < 0.05$ against MLP+phase+score (λ=0.5) (Welch over seeds).

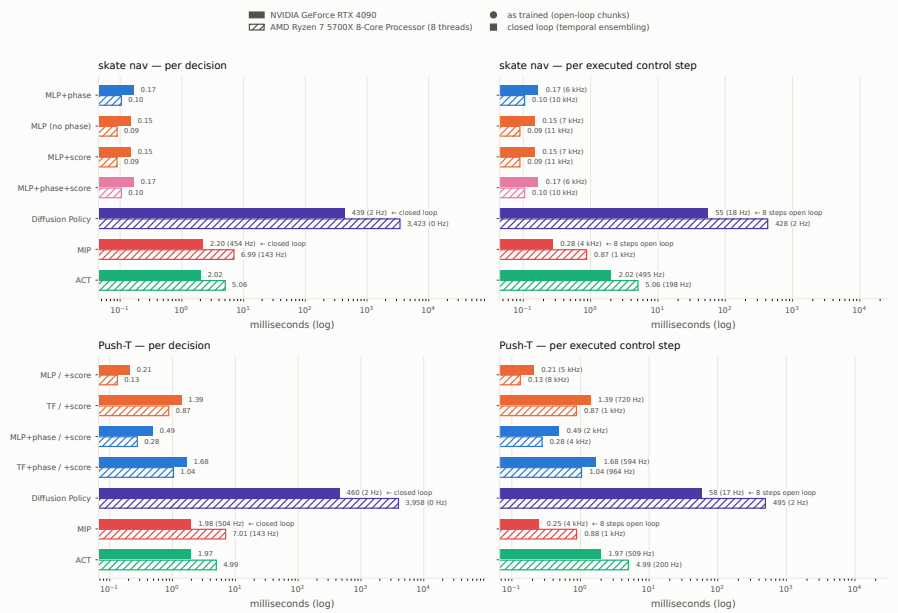

Figure 19: Inference latency breakdown per decision and per executed control step on CPU and GPU architectures. Deterministic MLP and TF policies execute faster than iterative baselines, satisfying the strict latency requirements necessary for real-time robotic control.