Robotics: Science and Systems 2026
Sydney, Australia, July 13–17, 2026

Guided Streaming Stochastic Interpolant Policy

Puming Jiang[1,‡], Meiyi Wang[2,‡], Kelvin Lin[3], Ce Hao[4] and Harold Soh[5]
School of Computing, National University of Singapore
[‡]Equal contribution
Email: {p.jiang[1], meiyi.wang[2], cehao[4]}@u.nus.edu, {klinzw[3], harold[5]}@nus.edu.sg

***Abstract*—Inference-time guidance is essential for steering generative robot policies toward dynamic objectives without retraining, yet existing methods are largely confined to chunk-based architectures that exhibit high latency and lack the reactivity needed for test-time preference alignment or obstacle avoidance. In this work, we formally derive the optimal guidance term for Stochastic Interpolants (SI) by analyzing the value function's time evolution via the Backward Kolmogorov Equation, establishing a modified drift that theoretically guarantees sampling from a target distribution. We apply this framework to real-time control through the Streaming Stochastic Interpolant Policy (SSIP), which generalizes the deterministic Streaming Flow Policy (SFP). Unifying this guidance law with the streaming architecture enables fast and reactive control. To support diverse deployment needs, we propose two complementary mechanisms: training-free Stochastic Trajectory Ensemble Guidance (STEG) that computes gradients on-the-fly for zero-shot adaptation, and training-based Conditional Critic Guidance (CCG) for amortized inference. Empirical evaluations demonstrate that our guided streaming approach significantly outperforms conventional chunk-based policies in reactivity and provides superior, physically valid guidance for dynamic, unstructured environments.**

I. Introduction

Modern generative approaches, such as Diffusion Models [20, 12, 4] and Flow Matching [16, 17], excel at capturing complex, multi-modal behaviors from demonstrations. However, their standard formulations do not explicitly account for the dynamic constraints of physical deployment. In real-world settings, a robot must not only imitate demonstrated motion but also adapt to safety constraints (e.g., obstacles) and evolving user preferences absent from the training data. Consequently, the utility of a policy is determined not only by its base capabilities, but also by how readily it can be steered at inference time toward additional objectives during execution, without the cost of retraining.

This capability is essential for deploying generative policies in unstructured, real-world environments and is commonly referred to as *inference-time guidance*. Originally introduced for controlled image generation [7], inference-time guidance has since been adapted to robot control to enforce safety constraints [9, 6] and to align behavior with human preferences [22]. However, existing guidance methods are largely tailored to chunk-based policies, such as Diffusion Policy [4], and inherit their limitations. In particular, actions must be generated for an entire chunk before execution begins, introducing latency and often leading to discontinuous motion [14]. Moreover, once a chunk is committed, execution proceeds open-loop over the chunk horizon, preventing adaptation to objectives that change mid-execution (e.g., moving obstacles). As a result, these methods are suboptimal for dynamic environments, as illustrated in Fig. 1.

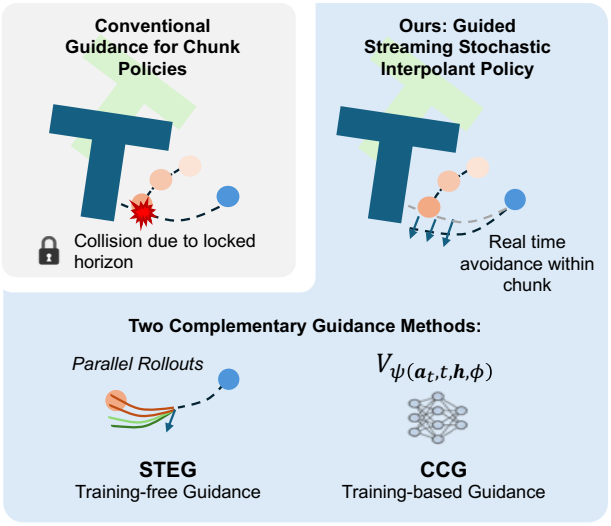


Fig. 1: We contribute a principled inference-time guidance framework for streaming generative policies. It enables fast reactive behavior within the action chunk, and can be deployed in either a training-based or training-free manner.

To address this challenge at a fundamental level, we ground our approach in the framework of *Stochastic Interpolants* (SI) [1]. SI provides a unifying formulation that encompasses score-based diffusion models (SDEs) [20] and flow matching methods (ODEs) [16, 17]. It has also been applied to robot policies to exploit the property that sampling need not originate from pure Gaussian noise, which accelerates the inference process [3]. While guidance mechanisms are well established for standard diffusion models, a rigorous treatment within the more general SI framework remains underexplored.

In this work, we first derive the optimal inference-time guidance for Stochastic Interpolants. This formulation is *general* and extends beyond policy guidance. We then propose the *Streaming Stochastic Interpolant Policy (SSIP)*, a streaming generative control framework that addresses the latency and motion smoothness limitations of chunk-based policies [3]. Similar to very recent work on streaming flow policies (SFP) [14], SSIP aligns generative evolution with the robot's physical execution time to enable incremental action generation during execution. However, unlike prior

streaming approaches, SSIP instantiates the more general Stochastic Interpolant framework. This changes the dynamics from a deterministic Ordinary Differential Equation (ODE) to a Stochastic Differential Equation (SDE), rendering the streaming policy mathematically compatible with the optimal inference-time guidance derived for SI.

With the theoretical guidance formulation and the SSIP backbone established, we propose two complementary strategies for computing the guidance gradient:

- Stochastic Trajectory Ensemble Guidance (STEG): A training-free approach utilizing parallel inference-time rollouts. By backpropagating through the differentiable trajectory ensemble, STEG estimates the guidance gradient on-the-fly for zero-shot adaptation.
- Conditional Critic Guidance (CCG): A training-based approach that amortizes the utility gradient into a learned critic network, which offers efficient inference for known obstacle classes.

Experiments on Push-T and Robomimic show that our unified framework improves latency and reactivity over existing guided Diffusion and Flow-Matching policies. For example, in the challenging Push-T Chase setting, chunk-based DP and FP achieve only 12.0% and 8.0% success, while SSIP variants maintain 30.0–36.0%. We further validate the framework on a real Panda robot for moving-obstacle avoidance and position-preference grasping, demonstrating its applicability beyond simulation.

II. Background & Related Work

Generative Robot Policies. Recent work has applied generative modeling to robot policy learning, commonly using action chunking to represent complex behaviors by predicting fixed-horizon action sequences (e.g., $a_{t:t+H}$). Representative examples include Diffusion Policy (DP) [4], Flow Policies (FP) [16], and Action Chunking with Transformers (ACT) [28]. Despite their expressiveness, these approaches require generating an entire action chunk prior to execution and operate open-loop within each chunk, with feedback incorporated only at chunk boundaries. Recent methods such as Consistency Policy [18], One-Step Diffusion Policy [23], and DPPO [19] improve the speed or performance of chunk-based diffusion policies; using smaller chunks can also shorten the open-loop horizon. Real-time chunking methods such as RTC [2] further improve asynchronous execution of chunk-based policies via inpainting and cross-chunk continuity. These directions are complementary to our focus on deriving test-time guidance for execution-time-aligned policies.

Very recent work on Streaming Flow Policies (SFP) [14] removes the reliance on action chunking by modeling action trajectories directly as flow trajectories. By aligning generative flow time with physical execution time, SFP enables incremental action generation and reduces latency while preserving performance. Our work builds on this execution-time-aligned perspective and studies how to equip streaming policies with principled test-time guidance.

Stochastic Interpolants. Stochastic Interpolants (SI) [1] provide a unified framework that connects flow-based generative models (probability-flow ODEs) and diffusion-based models (SDEs). Given endpoint densities ρ_0 and ρ_1, SI defines a family of intermediate marginals $\{\rho_t\}_{t\in[0,1]}$ via a stochastic interpolant

$$\mathbf{x}_t = I(t, \mathbf{x}_0, \mathbf{x}_1, \mathbf{u}) + \gamma(t)\mathbf{z}, \quad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{1}$$

where $(\mathbf{x}_0, \mathbf{x}_1) \sim \pi$ has marginals ρ_0, ρ_1, $\mathbf{u}$ is an auxiliary latent variable, I satisfies the endpoint conditions, and typically $\gamma(0) = \gamma(1) = 0$. The induced density ρ_t satisfies a transport equation and a family of forward/backward Fokker–Planck equations with a tunable diffusion coefficient $\epsilon(t)$, leading to associated ODE/SDE samplers (e.g., a forward SDE of the form $d\mathbf{x}_t = \mathbf{b}(\mathbf{x}_t, t)dt + \sqrt{2\epsilon(t)}\, d\mathbf{W}_t$). For simplicity, we assume independent couplings and omit $\mathbf{u}$. In robotics, Chen et al. [3] leverage *conditional* stochastic interpolants to bridge from informative (non-Gaussian) source policies, improving few-step policy diffusion performance.

Inference-time Guidance. Inference-time guidance methods, popularized in image synthesis through classifier guidance [7] and related energy- or loss-based guidance schemes [10, 26], modify the sampling process using gradients derived from auxiliary likelihoods, costs, or differentiable constraints. Related formulations include posterior-sampling methods with diffusion or flow priors, e.g., DPS [5], DAPS [27], and FlowDPS [15], as well as optimal-control-based flow guidance such as OC-Flow [21]; these methods mainly target inverse problems or general controlled generation.

In robotics, inference-time guidance has been used to enforce safety constraints, temporal-logic specifications, and kinematic requirements [22, 9, 25, 6, 11]. Most existing approaches rely on diffusion- or flow-based policies [4, 8, 13] that generate complete action chunks in a single inference step. While effective for static objectives, this open-loop execution paradigm exhibits the *commitment problem* [24], limiting responsiveness to time-varying constraints and dynamic environments.

Our contributions. We extend SI-based policies [3] to a *guided streaming* formulation, improving efficiency and reactivity. In doing so, we develop a general inference-time conditioning framework for *general* stochastic interpolants. When instantiated for robot control, this framework enables online adaptation to test-time constraints. From a complementary perspective, we generalize SFP from deterministic to stochastic dynamics, which permits a principled derivation of inference-time guidance using the Feynman–Kac representation and the Doob h-transform.

This stochastic-path perspective provides a natural route to guided streaming control, where stochasticity serves as the mathematical structure enabling the guidance derivation rather than as a claim of superior performance over flow matching.

III. Guided Streaming Stochastic Interpolant Policies (Guided SSIP)

Our main objective is to sample from a conditional target distribution defined over complete trajectories $q(\tau \mid \xi)$, where τ denotes a trajectory over the horizon $[0, T]$ (a path in action/state space) and ξ is an exogenous context (e.g., a goal specification or an observation/history embedding). This target is constructed by reweighting a base dynamics distribution $p(\tau \mid \xi)$ via a scalar cost function $J(\tau; \xi)$:

$$q(\tau \mid \xi) = \frac{1}{Z_\xi} p(\tau \mid \xi) e^{-J(\tau;\xi)} \tag{2}$$

where $Z_\xi = \int p(\tau \mid \xi) e^{-J(\tau;\xi)} d\tau$ is the global partition function. The cost function $J(\tau; \xi)$ encapsulates any inference-time objective used to guide the robot, such as obstacle avoidance or alignment with human preferences.

In this section, we establish the theoretical framework for Guided Streaming Stochastic Interpolant Policies. First, we derive the optimal guidance law for *general* stochastic interpolants (Section III-A). Next, we introduce our specific architecture, the Streaming Stochastic Interpolant Policy (SSIP), which generalizes the deterministic Streaming Flow Policy (SFP) into a Stochastic Differential Equation (SDE) to provide the necessary support for this guidance (Section III-B). Finally, we bridge global trajectory objectives and instantaneous control by deriving the action target conditional over the current time step (Section III-C). This yields a utility function that is compatible with the martingale-based guidance law, justifying its direct application to the streaming setting (Section III-D).

A. Optimal Guidance for Stochastic Interpolants

We derive the exact guidance law for a general conditional Stochastic Interpolant [1] whose base dynamics evolve over a generic state $\mathbf{x}_t$ and are governed by the diffusion

$$d\mathbf{x}_t = \mathbf{b}(\mathbf{x}_t, t, \xi)\, dt + \sqrt{2\epsilon(t)}\, d\mathbf{W}_t, \tag{3}$$

where $\mathbf{b}$ is the base drift, $\epsilon(t) \geq 0$ is an isotropic diffusion schedule, and $\mathbf{W}_t$ is a standard Wiener process. Our global objective (Eq. (2)) defines a *path-space* reweighting of the base trajectory measure conditional on ξ:

$$p_{\text{target}}(\tau_{0:T} \mid \xi) \propto p_{\text{base}}(\tau_{0:T} \mid \xi) e^{-J(\tau_{0:T},\xi)}, \tag{4}$$

where $J(\tau_{0:T}, \xi)$ is a trajectory-level cost. For the remainder of this subsection, we assume a standard *time-additive* form

$$J(\tau_{0:T}, \xi) \;=\; \int_0^T c(\mathbf{x}_s, s, \xi)\, ds \;+\; \phi(\mathbf{x}_T, \xi), \tag{5}$$

which includes pure terminal costs as the special case $c \equiv 0$.

Desirability and the martingale condition. Let $\{\mathcal{F}_t\}$ denote the natural filtration of the base process (3). Define the *global* likelihood-ratio process

$$D_t \triangleq \frac{\mathbb{E}_{\text{base}}\left[e^{-J(\tau_{0:T},\xi)} \mid \mathcal{F}_t\right]}{Z(\xi)} \tag{6}$$

where $Z(\xi) \triangleq \mathbb{E}_{\text{base}}\left[\exp\left(-J(\tau_{0:T}, \xi)\right)\right]$. By construction and the tower property, $\{D_t\}_{t\in[0,T]}$ is a (nonnegative) martingale with $\mathbb{E}[D_t] = 1$. Under the additivity assumption (5) and Markovity of (3), the conditional expectation in (6) factorizes into a *sunk* past factor and a *future* desirability:

$$\mathbb{E}_{\text{base}}\left[e^{-J(\tau_{0:T},\xi)} \mid \mathcal{F}_t\right] = \exp\left(-\int_0^t c(\mathbf{x}_s, s, \xi) ds\right) u(\mathbf{x}_t, t, \xi), \tag{7}$$

where the desirability (a.k.a. exponentiated future utility) is

$$u(\mathbf{x}, t, \xi) \triangleq \mathbb{E}_{\text{base}}\left[\exp\left(-\int_t^T c(\mathbf{x}_s, s, \xi)\, ds - \phi(\mathbf{x}_T, \xi)\right) \,\middle|\, \mathbf{x}_t = \mathbf{x}\right] \tag{8}$$

Applying Itô's lemma to the martingale in (7) yields the (backward) Feynman–Kac equation satisfied by u:

$$\partial_t u(\mathbf{x}, t, \xi) + \mathcal{L}_t u(\mathbf{x}, t, \xi) - c(\mathbf{x}, t, \xi)\, u(\mathbf{x}, t, \xi) = 0, \tag{9}$$

with $u(\mathbf{x}, T, \xi) = e^{-\phi(\mathbf{x},\xi)}$, and the time-dependent generator

$$\mathcal{L}_t f \;=\; \mathbf{b}(\mathbf{x}, t, \xi) \cdot \nabla_{\mathbf{x}} f \;+\; \epsilon(t)\, \Delta_{\mathbf{x}} f. \tag{10}$$

In the pure terminal-cost case ($c \equiv 0$), (9) reduces to the homogeneous backward Kolmogorov equation $\partial_t u + \mathcal{L}_t u = 0$.

Optimal drift via a Doob h-transform. We now characterize the dynamics under the tilted path measure (4). Using (6) as the Radon-Nikodym derivative on $\mathcal{F}_t$, Itô calculus together with (9) gives

$$dD_t \;=\; D_t \sqrt{2\epsilon(t)}\, \nabla_{\mathbf{x}} \log u(\mathbf{x}_t, t, \xi) \cdot d\mathbf{W}_t. \tag{11}$$

By Girsanov's theorem, under the tilted measure the process remains a diffusion with the *same* diffusion coefficient and a shifted drift:

$$d\mathbf{x}_t = \underbrace{\left[\mathbf{b}(\mathbf{x}_t, t, \xi) + 2\epsilon(t)\, \nabla_{\mathbf{x}} \log u(\mathbf{x}_t, t, \xi)\right]}_{\tilde{\mathbf{b}}(\mathbf{x}_t, t, \xi)} dt + \sqrt{2\epsilon(t)}\, d\widetilde{\mathbf{W}}_t,$$

where $\widetilde{\mathbf{W}}_t$ is a Wiener process under the tilted measure. Therefore, the exact guidance modification is

$$\Delta\mathbf{b}(\mathbf{x}, t, \xi) \triangleq \tilde{\mathbf{b}}(\mathbf{x}, t, \xi) - \mathbf{b}(\mathbf{x}, t, \xi) = 2\epsilon(t)\, \nabla_{\mathbf{x}} \log u(\mathbf{x}, t, \xi). \tag{12}$$

This result is the continuous-time analogue of KL-minimal control; we obtain the target path reweighting (4) by adding the smallest possible (in relative-entropy) drift correction consistent with the tilted measure.

B. Streaming Stochastic Interpolant Policy (SSIP)

We define SSIP as an execution-time-aligned diffusion policy whose time-indexed marginals follow a stochastic interpolant path. This is strictly more general than any particular deterministic streaming flow; deterministic policies arise as special cases (e.g., drift-only samplers or zero-noise interpolants), while the diffusion formulation is what enables

principled guidance via the stochastic control results (in Section III-A).

Here, the diffusion state is $\mathbf{x}_t = \mathbf{a}_t$ and context $\xi = \mathbf{h}$ denotes a snapshot of the policy context at the current control cycle (e.g., observation/history embedding). Within a single SSIP rollout over $t \in [0, T]$, we treat $\mathbf{h}$ as fixed, similar to SFP [14]. Within each action chunk, while the history $\mathbf{h}$ remains fixed, each new drift is computed using the last action as input. This ensures reactivity, unlike conventional methods where the entire chunk is computed simultaneously.

Then, a stochastic interpolant policy specifies a family of conditional marginals $\{p_t(\mathbf{a} \mid \mathbf{h})\}_{t\in[0,T]}$ by constructing a noised sample as

$$\mathbf{a}_t = \mathcal{I}_t(\mathbf{h}, \zeta) + \gamma(t)\,\mathbf{z}, \qquad \mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \tag{13}$$

where: (i) $\mathcal{I}_t(\mathbf{h}, \zeta)$ is a *reference interpolant* ("signal path") that may be deterministic given $\mathbf{h}$ or may depend on a latent ζ capturing structured trajectory variation, and (ii) $\gamma(t)$ is a smooth scalar noise schedule controlling the marginal perturbation scale (often chosen to be small near the endpoints and larger mid-horizon). Note that Eq. (13) defines a *path of marginals* used for training (by sampling fresh $\mathbf{z}$ for each $(t, \mathbf{h})$); it does not mean that a single fixed $\mathbf{z}$ is carried through time as a Brownian path. The continuous-time sampler is an SDE driven by a Wiener process $d\mathbf{W}_t$ (defined below). Eq. (13) is the policy-level instance of Eq. (1): the generic state $\mathbf{x}_t$ corresponds to the action $\mathbf{a}_t$, and the interpolant is conditioned on the policy context $\mathbf{h}$.

To realize the marginal path induced by (13) at inference time, we use a diffusion with drift decomposed into a transport term and a score term. Let

$$s_\theta(\mathbf{a}, t, \mathbf{h}) \approx \nabla_{\mathbf{a}} \log p_t(\mathbf{a} \mid \mathbf{h})$$

denote a learned score, and let $v_\theta(\mathbf{a}, t, \mathbf{h})$ denote a learned transport/velocity field. We define the SSIP sampling dynamics as the one-parameter family

$$d\mathbf{a}_t = \Big[v_\theta(\mathbf{a}_t, t, \mathbf{h}) + \big(\epsilon(t) - \gamma(t)\dot\gamma(t)\big)\, s_\theta(\mathbf{a}_t, t, \mathbf{h})\Big]dt + \sqrt{2\epsilon(t)}\, d\mathbf{W}_t, \tag{14}$$

where $\epsilon(t) \geq 0$ controls the *amount of stochasticity* injected during sampling. The term $-\gamma(t)\dot\gamma(t)\, s_\theta$ is the deterministic correction associated with the marginal noising schedule $\gamma(t)$; the additional $\epsilon(t)\, s_\theta$ and $\sqrt{2\epsilon(t)}\, d\mathbf{W}_t$ provide a tunable stochastic sampler. In the idealized setting where s_θ equals the true score, different choices of $\epsilon(t)$ correspond to different samplers that target the same marginal path, trading off exploration versus determinism.

SSIP cleanly separates two conceptually different kinds of "randomness":

- Structured variation (model diversity): randomness in $\mathcal{I}_t(\mathbf{h}, \zeta)$ (e.g., latent-conditioned behaviors or multimodal skills). This represents genuine alternative behaviors the policy has learned.
- Sampling diffusion (exploration noise): stochasticity controlled by $\epsilon(t)$ and realized by the Wiener process $d\mathbf{W}_t$. This is an inference-time knob that enables exploration and supports principled guidance.

We can recover the deterministic streaming flow policy by using a drift-only sampler,

$$\frac{d\mathbf{a}_t}{dt} = v_\theta(\mathbf{a}_t, t, \mathbf{h}) - \gamma(t)\dot\gamma(t)\, s_\theta(\mathbf{a}_t, t, \mathbf{h}). \tag{15}$$

and choosing a noise-free interpolant ($\gamma(t) \equiv 0$). This is the deterministic counterpart of (14) and is the default at execution when low-noise trajectories are preferred. Then, (15) reduces to $\dot{\mathbf{a}}_t = v_\theta(\mathbf{a}_t, t, \mathbf{h})$. Specific constructions such as "Gaussian-tube" stabilizing dynamics correspond to particular choices of $\mathcal{I}_t$ and v_θ (and optionally a latent ζ), but are not required by the SSIP framework. We provide an extended SSIP formulation and training details in Appendix A.

C. Derivation of the Target Conditional

SSIP operates by sampling the current action/state at time t from an implicit generative distribution. To apply the trajectory-level target in Eq. (2), we derive the induced *target conditional* over the current action given the realized past. For clarity, we write trajectories as $\tau_{0:T} = (\mathbf{a}_0, \ldots, \mathbf{a}_T)$. At decision time t, the realized history up to but *excluding* the current action is fixed, i.e., $\tau_{0:t-1}$ is given and $\mathbf{a}_t$ is the decision variable[1].

As before, the global target over complete trajectories is

$$p_{\text{target}}(\tau_{0:T} \mid \xi) = \frac{1}{Z_\xi}\, p_{\text{base}}(\tau_{0:T} \mid \xi) e^{-J(\tau_{0:T}, \xi)}. \tag{16}$$

For readability, we suppress ξ in the remainder of the derivation in this subsection. By marginalizing future trajectories $\tau_{t+1:T}$, the target conditional over the current action is

$$p_{\text{target}}(\mathbf{a}_t \mid \tau_{0:t-1}) = \int p_{\text{target}}(\tau_{t:T} \mid \tau_{0:t-1})\, d\tau_{t+1:T}. \tag{17}$$

Using (16) and dropping normalization constants that do not depend on $\mathbf{a}_t$ yields

$$p_{\text{target}}(\mathbf{a}_t \mid \tau_{0:t-1}) \propto \int p_{\text{base}}(\tau_{t:T} \mid \tau_{0:t-1}) e^{-J(\tau_{0:T})} d\tau_{t+1:T}. \tag{18}$$

We can factor the base conditional into two components,

$$p_{\text{base}}(\tau_{t:T} \mid \tau_{0:t-1}) = \underbrace{p_{\text{base}}(\mathbf{a}_t \mid \tau_{0:t-1})}_{\text{Current Action}}\, \underbrace{p_{\text{base}}(\tau_{t+1:T} \mid \tau_{0:t-1}, \mathbf{a}_t)}_{\text{Future Rollout}}. \tag{19}$$

Substituting (19) into (18) and pulling out the $\mathbf{a}_t$-dependent term gives

$$p_{\text{target}}(\mathbf{a}_t \mid \tau_{0:t-1}) \propto p_{\text{base}}(\mathbf{a}_t \mid \tau_{0:t-1})\, \mathbb{E}_{\tau_{t+1:T} \sim p_{\text{base}}(\cdot \mid \tau_{0:t-1}, \mathbf{a}_t)} \left[e^{-J(\tau_{0:T})}\right]. \tag{20}$$

[1]We will work with discrete time-steps, but the derivation for continuous time is identical if we replace $\tau_{0:t-1}$ with the filtration $\mathcal{F}_t$.

Under additive costs (Eq. (5)), the trajectory cost splits as $J(\tau_{0:T}) = J(\tau_{0:t-1}) + J(\tau_{t:T})$, where $J(\tau_{0:t-1})$ is fully determined by the fixed past. Therefore,

$$\mathbb{E}\Big[e^{-J(\tau_{0:T})} \mid \tau_{0:t-1}, \mathbf{a}_t\Big] = e^{-J(\tau_{0:t-1})}\mathbb{E}\left[e^{-J(\tau_{t:T})} \mid \tau_{0:t-1}, \mathbf{a}_t\right], \quad (21)$$

and the prefactor $e^{-J(\tau_{0:t-1})}$ cancels in (20) because it is independent of $\mathbf{a}_t$. Define the (unnormalized) *utility* of the current action as the expected exponentiated *future* cost:

$$u(\mathbf{a}_t, t, \tau_{0:t-1}) \triangleq \mathbb{E}_{\tau_{t+1:T}\sim p_{\text{base}}(\cdot|\tau_{0:t-1},\mathbf{a}_t)}\left[e^{-J(\tau_{t:T})}\right] \quad (22)$$

Then the exact target conditional is

$$p_{\text{target}}(\mathbf{a}_t \mid \tau_{0:t-1}) \propto p_{\text{base}}(\mathbf{a}_t \mid \tau_{0:t-1})\, u(\mathbf{a}_t, t, \tau_{0:t-1}). \quad (23)$$

D. Guiding Streaming Policies

To guide SSIP, we apply the general result of Section III-A to the SSIP base sampler (14). Define the SSIP base drift

$$\mathbf{b}_\theta(\mathbf{a}, t, \mathbf{h}) \triangleq v_\theta(\mathbf{a}, t, \mathbf{h}) + \big(\epsilon(t) - \gamma(t)\dot{\gamma}(t)\big)s_\theta(\mathbf{a}, t, \mathbf{h}). \quad (24)$$

Let the streaming desirability (future utility) be

$$u(\mathbf{a}_t, t, \mathbf{h}) \triangleq \mathbb{E}_{\tau_{t+1:T}\sim p_{\text{base}}(\cdot|\mathbf{a}_t,t,\mathbf{h})}[\exp(-J(\tau_{t:T}))]. \quad (25)$$

Then the guided SSIP dynamics are obtained by adding the optimal correction $\Delta\mathbf{b} = 2\epsilon(t)\nabla_{\mathbf{a}} \log u$:

$$d\mathbf{a}_t = \Big[\mathbf{b}_\theta(\mathbf{a}_t, t, \mathbf{h}) + 2\epsilon(t)\nabla_{\mathbf{a}} \log u(\mathbf{a}_t,t,\mathbf{h})\Big]dt + \sqrt{2\epsilon(t)}\, d\mathbf{W}_t. \quad (26)$$

Reactivity is ensured by receding-horizon execution: after executing an action from the current rollout, we compute the next step based on the current action $\mathbf{a}_t$ and the utility calculated at that specific timestep, which reflects the instantaneous future trajectory cost $J(\tau_{t:T})$. In practice, the remaining challenge is estimating $\nabla_{\mathbf{a}} \log u$ efficiently at inference time, which we address in the following section.

IV. Guidance for On-the-fly Obstacles

As shown above, the optimal guidance drift is proportional to the gradient of the log-expected future utility: $\nabla_{\mathbf{a}} \log \mathbb{E}[e^{-J(\tau_{t:T})}]$. While this formulation provides a theoretical foundation for constraint satisfaction, computing this gradient exactly is often computationally intractable, as it necessitates differentiating through an expectation over a high-dimensional distribution of future trajectories. To bridge this gap, we propose two complementary strategies tailored to different deployment requirements, described below.

A. Stochastic Trajectory Ensemble Guidance (STEG)

To enable zero-shot adaptation, we propose Stochastic Trajectory Ensemble Guidance (STEG). This method leverages the differentiability of our SSIP backbone to construct a Monte Carlo estimator of the utility gradient $\nabla_{\mathbf{a}} \log \mathbb{E}[e^{-J(\tau)}]$ on-the-fly. Estimating the guidance gradient using a finite set of sample trajectories introduces two primary sources of error: *Distributional Shift* (failure to cover safe modes) and *Dynamics Mismatch* (incorrect gradient direction). In Appendix B, we provide an error decomposition of the guidance score. Our analysis suggests two critical design principles for a valid estimator:

- High-Fidelity Support ($N \uparrow$): The estimator must employ a large ensemble size to minimize the selection error arising from the multi-modal nature of the safety posterior.
- Physics-Consistent Gradients: The rollout mechanism must preserve causal derivatives ($\partial\mathbf{a}_{t+k}/\partial\mathbf{a}_t$) to ensure the guidance force respects system kinematics.

STEG implements these principles by combining parallel sampling with differentiable physics.

At inference time, STEG approximates the intractable expectation $\mathbb{E}_\tau[e^{-J(\tau)}]$ by simulating a batch of N parallel chains. We use the learned SDE dynamics itself as the transition model, conditioning on the snapshot context $\mathbf{h}$ and holding it fixed throughout the K-step rollout. The process consists of three steps:

1) Stochastic Ensemble Rollout. We replicate the current state $\mathbf{a}_t$ into an ensemble $\{\mathbf{a}_t^{(i)}\}_{i=1}^N$. We then unroll the dynamics for K steps using the Euler-Maruyama discretization of the base SSIP generative SDE.
2) Utility Aggregation. During the rollout, we accumulate the running cost for each trajectory $J(\tau^{(i)}) = \sum_{k=1}^K c(\mathbf{a}_k^{(i)})\Delta t$, where $c(\cdot)$ is the additive cost function (e.g., repulsive potential). We then estimate the log-expected utility by employing the Log-Sum-Exp (LSE) estimator:
$$\hat{\mathcal{V}}(\mathbf{a}_t) = \text{LogSumExp}\left(\{-J(\tau^{(i)})\}_{i=1}^N\right) - \log N \approx \log \mathbb{E}[e^{-J}] \quad (27)$$
3) Backpropagation Through Time (BPTT). The optimal guidance drift is the gradient of this value estimate. Leveraging automatic differentiation, we compute the gradient w.r.t. the current action $\mathbf{a}_t$ by backpropagating through the unrolled graph:
$$\Delta\mathbf{b}(\mathbf{a}_t, t, \mathbf{h}) = 2\epsilon(t) \cdot \nabla_{\mathbf{a}_t}\hat{\mathcal{V}}(\mathbf{a}_t) \quad (28)$$

In summary, STEG provides a principled, training-free mechanism for inference-time guidance by directly estimating the utility gradient through differentiable stochastic rollouts. STEG enables robust zero-shot adaptation to novel objectives while preserving the smoothness and reactivity of the underlying SSIP dynamics. This makes STEG well-suited for unstructured and rapidly changing environments, where safety constraints and task objectives cannot be anticipated at training time.

B. Conditional Critic Guidance (CCG)

To achieve real-time reactivity, we propose Conditional Critic Guidance (CCG), where we train a parameterized critic network V_ψ to approximate the guidance potential directly. Since the base SSIP policy generates actions autoregressively, the distribution of future trajectories is inherently conditioned

on the robot's history. Consequently, the value function must account for this temporal context. We parameterize the critic as $V_\psi(\mathbf{a}_t, t, \mathbf{h}, \phi)$, where $\mathbf{h}$ represents the history context (e.g., observation history) used by the base policy, and ϕ represents the parameters defining the cost (e.g., human preferences or obstacle positions). In the following sections, we specifically use ϕ to denote obstacle parameters. The network is trained to approximate a scalar value y_{target} derived from future rollouts:

$$V_\psi(\mathbf{a}_t, t, \mathbf{h}, \phi) \approx y_{\text{target}}(\mathbf{a}_t, \mathbf{h}, \phi) \tag{29}$$

Here, y_{target} estimates the log-desirability $\log u(\mathbf{a}_t, t, \mathbf{h})$ in Eq. (25), whose gradient yields the guidance correction in Eq. (26) for inducing the target conditional in Eq. (23).

In practice, CCG is most effective in structured environments where obstacle classes and cost functions are known at training time. By amortizing the guidance computation into a learned critic, CCG enables low-latency inference when prior knowledge is available.

C. Summary of the Main Method

In summary, Guided SSIP proceeds in three steps. First, we instantiate the streaming SI sampler with the base drift in Eq. (24). Second, at each execution step, we estimate the log-desirability $\log u(\mathbf{a}_t, t, \mathbf{h})$: STEG estimates it online through short-horizon stochastic rollouts, while CCG amortizes it with a trained critic. Third, we apply the guidance correction in Eq. (26) and execute the resulting action in a receding-horizon manner. The full inference procedures are given in Alg. 1 for STEG and Alg. 2 for CCG.

V. Experiments

We designed our experiments to investigate the following research questions:

- (**Q1**) How does our method compare to conventional guidance techniques in static environments?
- (**Q2**) Does our method demonstrate adaptability when encountering dynamic or unexpectedly appearing obstacles compared to baseline techniques?
- (**Q3**) What is the computational overhead of STEG compared to CCG with pre-trained critics, and what are the respective strengths and weaknesses of each approach in real-time scenarios?

A. Experimental Setup

To answer these questions, we integrate our streaming guidance techniques with the SSIP and evaluate them on obstacle avoidance in two simulation domains: the Push-T task and simulated robot manipulation in Robosuite. All base policies are trained on datasets containing *only collision-free* demonstrations. During evaluation, we introduce unmodeled static and dynamic obstacles to assess the capabilities of our proposed guidance framework.

Baselines and SSIP Variants. We compare our methods against two baselines: conventional action chunk-based diffusion policy (DP) with energy-based guidance [13] and flow-policy (FP) with flow-guidance [8]. For SSIP, we compare the STEG and CCG guidance methods against a simple repulsion method (a simple additive residual control based on the distance to obstacles that works well in practice). For STEG guidance, we use $K = 3$ rollout steps with $N = 64$ parallel rollouts for Push-T and $K = 4$ rollout steps with $N = 32$ parallel rollouts for Robomimic. For CCG guidance, we employ two different critics trained with the following costs:

1) Distance Potential Cost (CCG-D): the regression target approximates the log-expected future utility $\log \mathbb{E}[e^{-J}]$ by aggregating cumulative distance costs from rollouts.
2) Collision Proxy Cost (CCG-P): the target is a collision-risk proxy computed from future rollouts (based on *minimum* distance and approach angle). This relaxes the additive-cost assumption required for the exact Doob-transform guidance, but provides a practical, reactive guidance field.

TABLE I: Experimental results on the Push-T task with static obstacles. We compare Task Success Rate (SR) and Inference Latency. Our guided SSIP methods outperform the baselines in terms of success rate with lower latency. **Bold** indicates the best performance.

Category	**Method**	**SR%** (↑)	**Latency** (ms)
Baseline	DP	58.5	91.52
	FP	56.0	**7.78**
SSIP	Repulsion	68.6	11.37
	STEG	66.7	18.49
	CCG-D	65.7	13.72
	CCG-P	**74.3**	14.75

Evaluation Protocols. Guided policies exhibit varying sensitivity to the guidance scale λ, which controls the trade-off between task performance and safety. To ensure a fair and comprehensive comparison, we adopt two complementary evaluation protocols:

- Peak Performance Metrics. We perform a grid search over λ and report results at the configuration that achieves the highest task success rate (SR). A trajectory is considered successful if its final reward exceeds a task-specific threshold (85%, corresponding to a slight relaxation to account for necessary avoidance maneuvers). We additionally report inference latency, defined as the average wall-clock time required to generate a single action step[2].
- Safety-Reward Trade-off. We sweep λ and report the Pareto frontier of average reward versus safety rate ($1 -$ Collision Rate). Methods closer to the top-right corner exhibit stronger task performance while maintaining higher safety.

B. Push-T with Static Obstacles

On this task, the robot has to complete the task (push the T-block into a particular goal configuration) without colliding

[2]All evaluations were conducted on a workstation with an AMD EPYC 7543 processor and a single A5000 GPU.

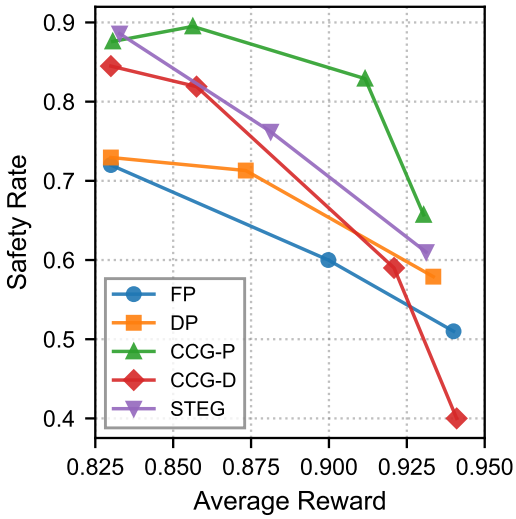


Fig. 2: Safety-Reward trade-off. SSIP consistently achieves more favorable trade-offs compared to the baselines, with its Pareto frontiers closer to the top-right corner of the reward–safety plane. Among the SSIP variants, CCG-P dominates across the evaluated range, while STEG outperforms the CCG-D variant.

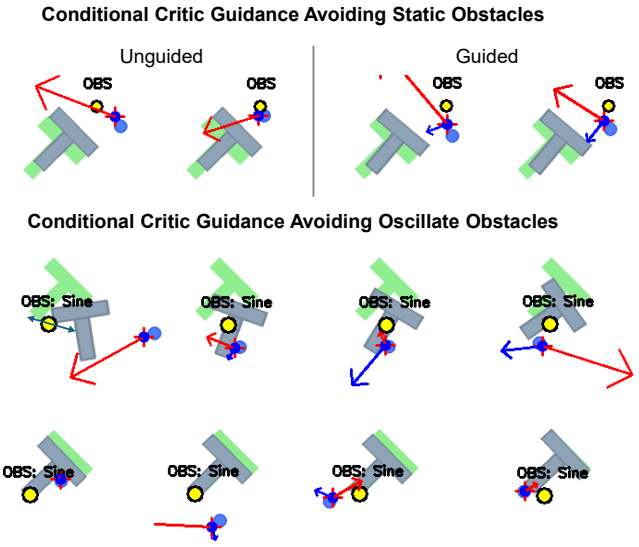


Fig. 3: SSIP with CCG-P guidance in static and dynamic obstacle tasks. Red arrows (→) represent the base policy while blue arrows (→) represent guidance. We can observe the guidance acts to push the end-effector away from the obstacles.

with any of eight static obstacles. We execute the base policy and filter for instances where it achieves task success but experiences collisions, yielding a set of approximately 100 cases per method.

Results. Table I reports success rate and inference latency on Push-T with static obstacles. Overall, SSIP improves task success relative to conventional chunk-based guidance baselines, addressing **Q1**. In particular, CCG-P achieves the highest success rate (74.3%) while maintaining low per-step latency (14.75 ms), outperforming classifier-guided DP (58.5%, 91.52 ms) and flow-guided FP (56.0%, 7.78 ms). Notably, a simple repulsion residual already performs strongly in this static setting (68.6%, 11.37 ms), indicating that local distance-based avoidance is often sufficient when obstacles are fixed.

Crucially, the per-step latency reported for conventional baselines represents an amortized metric. To reflect the true control latency, this value must be multiplied by the execution chunk size ($H_{execute}$=8), as these methods generate the entire action sequence simultaneously. Consequently, the robot faces a "wait-then-act" bottleneck resulting in unsmooth execution, a limitation also identified in SFP [14].

The distance-based instantiations of our principled guidance—CCG-D (65.7%, 13.72 ms) and STEG with short-horizon rollouts K=3 (66.7%, 18.49 ms)—are comparable to, and slightly below, repulsion: they optimize essentially the same distance signal but incur additional approximation error from critic regression (CCG-D) or finite-sample, short-horizon Monte Carlo estimation (STEG). This highlights a key trade-off for **Q3**: CCG amortizes guidance into a critic for lower-latency inference, whereas STEG adds overhead due to online rollouts but remains training-free and broadly applicable.

The trade-off curves shown in Fig. 2 are consistent with this view. SSIP yields a more favorable safety–reward trade-off than the chunk-based baselines, with Pareto frontiers consistently closer to the top-right region of the reward–safety plane. Among SSIP variants, CCG-P dominates across the swept guidance scales, maintaining high safety at comparable (or higher) reward. STEG improves over conventional baselines and generally exceeds CCG-D, but exhibits a less favorable trade-off than CCG-P in the high-reward regime, indicating a sharper safety degradation as reward is pushed upward.

C. Push-T with Dynamic Obstacles

To test reactivity, we introduce dynamic obstacles during the robot's execution. The obstacles follow three distinct movement patterns, increasing in kinematic difficulty: *Intercept* moves predictably toward the trajectory midpoint, *Oscillate* moves back and forth around the robot's nominal path, and *Chase* continuously follows the end-effector, making within-horizon adaptation most challenging.

Results. Table II shows that dynamic obstacles substantially amplify the limitations of chunk-based baselines, highlighting the benefit of streaming reactivity. In the simplest *Intercept* setting, DP and FP remain competitive (72–76%), and SSIP with CCG-P matches the best baseline (76%), suggesting that when obstacle motion is predictable and brief, chunk-based guidance can still recover.

As obstacle dynamics become more adversarial, SSIP variants separate more clearly from the baselines. Under *Oscillate*, CCG-P achieves the highest success (68%), outperforming both DP (38%) and FP (50%), indicating that continuous,

TABLE II: Experimental results on the Push-T task with dynamic obstacles. We evaluate the Task Success Rate (SR) across three distinct obstacle motion patterns: moving towards the trajectory midpoint (Intercept), oscillating around the midpoint (Oscillate), and chasing the end-effector (Chase).

		Success Rate (SR%) ↑		
Category	**Method**	**Intercept**	**Oscillate**	**Chase**
Baseline	DP	72.0	38.0	12.0
	FP	**76.0**	50.0	8.0
SSIP	Repulsion	64.0	54.0	30.0
	CCG-P	**76.0**	**68.0**	30.0
	STEG	68.0	56.0	**36.0**

TABLE III: **Robomimic Simulation Results (Success Rate).** We report the Success Rate (SR %) across three tasks.

		Success Rate (SR%) ↑		
Category	**Method**	**Lift**	**Square**	**Can**
Baseline	DP	71.0	58.0	51.0
	FP	72.0	48.0	51.0
SSIP	Repulsion	35.0	32.0	14.0
	STEG	45.0	50.0	51.0
	CCG-P	**73.0**	**79.0**	**78.0**

online guidance is particularly effective when constraints change within an execution horizon. The most challenging *Chase* regime further stresses within-horizon adaptation; both chunk-based baselines collapse (12% for DP, 8% for FP), while SSIP methods retain markedly higher success (30–36%). Notably, STEG performs best in *Chase* (36%), consistent with the advantage of training-free, on-the-fly gradient estimation when obstacle behavior deviates from training-time conditions.

Overall, these results suggest a regime shift; chunk-based guidance is brittle under rapidly changing constraints, while SSIP enables substantially higher success. CCG-P excels in moderately dynamic settings and STEG provides the strongest robustness in the most adversarial motion pattern.

D. Simulation Robot Experiment

We conduct experiments on three canonical Robomimic tasks: *Lift*, *Can*, and *Square*. We introduce unmodeled static obstacles that intersect the nominal demonstration trajectories, which requires the agent to deviate to succeed. We evaluate 100 trials per baseline per task.

Results. Table III reports success rates across the three Robomimic tasks. Overall, SSIP with CCG-P guidance achieves the strongest performance, substantially outperforming all baselines across tasks. This result highlights the benefit of amortized, history-aware guidance when reliable prior structure is available. Among the remaining SSIP variants, STEG consistently improves over the simple repulsion baseline but underperforms CCG-P. This gap is most pronounced in tasks where the obstacle directly intersects the nominal trajectory,

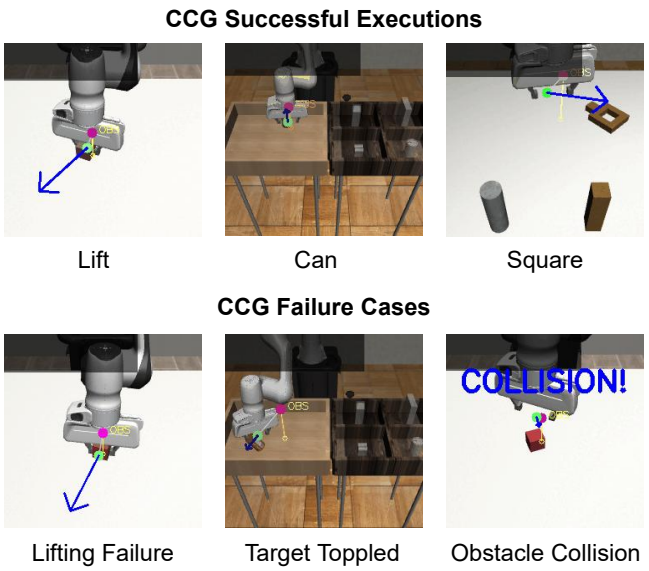


Fig. 4: Visualization of CCG execution across three Robomimic tasks, including representative failure examples. Blue arrows indicate the guidance force.

suggesting that finite-horizon stochastic rollouts can struggle to discover sufficiently long-horizon detours under coarse discretization. Nevertheless, STEG still provides meaningful gains over heuristic guidance without requiring offline training. In contrast, the repulsion-based method performs poorly across all tasks, indicating that local reactive penalties are insufficient to resolve obstacle interference while preserving task completion. This contrasts with the Push-T case, where simple repulsion did not significantly disrupt policy execution. Finally, conventional Diffusion and Flow Policies exhibit performance better than STEG, indicating that they can still provide reasonable guidance forces in high-dimensional spaces when the reward function remains static within the chunk.

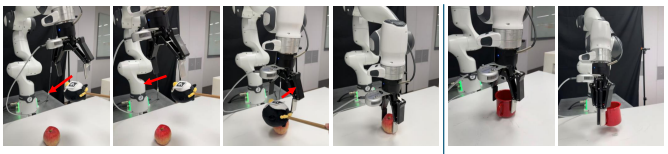
Analysis of failed episodes for CCG-P and STEG reveals two primary failure modes: *insufficient avoidance* (leading to collision) and *task interference* (leading to non-completion). Collisions (Fig. 4, bottom right) typically stem from the distributional constraints of the base policy. Since our guidance method operates by reweighting the learned trajectory distribution, it struggles to synthesize flexible avoidance behaviors if such kinematic diversity is absent from the training dataset. Conversely, task failures (e.g., failure to lift, Fig. 4, bottom left; or knocking over the can, Fig. 4, bottom middle) arise when guidance gradients directly conflict with task execution. For instance, in lifting tasks, strong repulsive forces effectively cancel out the necessary vertical actuation when the planner fails to identify a feasible lateral detour.

E. Real Robot Experiment

We further evaluated the methods on a real Franka Panda robot across two task types: moving-obstacle avoidance and position-preference grasping, where the robot is guided to

TABLE IV: **Real Robot Results (Success Rate).**

Method	Moving Obstacles			Position
	Slow-Steady	Slow-Diverse	Fast	Preference
DP	65	25	5	85
SSIP-Naïve	40	65	55	90
SSIP-STEG	**90**	**75**	55	**95**
SSIP-CCG	**90**	**75**	**65**	90



Moving Obstacle Avoidance | Grasp Different Position

Fig. 5: Real Robot Experiment on moving-obstacle avoidance and position-preference grasping.

grasp different parts of a mug, such as the handle or the body. For moving-obstacle avoidance, the three settings vary in the speed and diversity of the obstacle trajectories, and each setting is tested over 20 trials. As shown in Table IV, guided DP often struggles with fast-moving obstacles: it either responds only when the obstacle is already close or produces abrupt evasive motions that interfere with task execution. These results also support our claim that our guidance formulation improves over naive guidance, which uses only repulsion/attraction heuristics and does not explicitly account for the execution of the underlying task. The position-preference experiment further demonstrates that our framework can support inference-time objectives beyond obstacle avoidance, enabling the robot to grasp different parts of the mug according to the specified preference. In addition, we observe the same trend as reported in SFP [14]: streaming policies tend to produce smoother trajectories than diffusion-policy-style generation.

VI. Conclusion, Limitations, and Future Work

This paper presented a principled framework for inference-time steering of generative robot policies together with a streaming execution substrate. We derived the optimal guidance law for *Stochastic Interpolants* and instantiated it in a *Streaming Stochastic Interpolant Policy (SSIP)* that generates actions incrementally, avoiding the "wait-then-act" bottleneck inherent to chunk-based architectures. Empirically, SSIP improves reactivity relative to chunk-based diffusion and flow-policy baselines across both static and dynamic obstacle settings. The proposed guidance instantiations exhibit complementary strengths: CCG provides low-latency, amortized steering when task and constraint structure is available at training time, whereas STEG enables training-free adaptation via on-the-fly gradient estimation under distribution shift.

Across benchmarks, we find that effective inference-time steering is regime-dependent and shaped jointly by the execution architecture and the objective surrogate. On Robomimic tasks with unmodeled obstacles intersecting nominal demonstrations, simple local repulsion performs poorly, indicating that successful avoidance often requires *task-aware* deviations rather than purely reactive distance penalties. In contrast, SSIP with amortized guidance (CCG-P) substantially improves success, suggesting that history- and cost-conditioned critics can learn structured, goal-preserving avoidance behaviors when representative deployment conditions are observed during training. Training-free STEG also improves over repulsion, but underperforms CCG-P in settings requiring longer-horizon detours, consistent with limitations from finite-horizon rollouts and coarse discretization. Complementing these static-obstacle results, Push-T with dynamic obstacles highlights the brittleness of chunk-based policies under within-horizon constraint changes, while streaming SSIP maintains higher success; notably, STEG provides the strongest robustness in the most adversarial *Chase* regime, consistent with the benefits of online adaptation.

Limitations and Future Work. We identify two primary limitations. First, the policy is constrained by the support of the training trajectory distribution, which can restrict kinematic flexibility during close-range avoidance. Second, STEG operates over a finite prediction horizon, limiting its ability to anticipate obstacles that require longer-term planning. These effects compound: limited horizon discourages early detours, while distributional constraints hinder late-stage maneuvering. Future work will focus on extending the effective look-ahead horizon of streaming guidance, for example through longer-horizon rollouts or hierarchical planning interfaces.

Acknowledgements

This research / project is supported by the National Research Foundation, Singapore, under its Thematic Competitive Research Programme 2025 (NRF-T-CRP-2025-0003)

References

[1] Michael Albergo, Nicholas M Boffi, and Eric Vanden-Eijnden. Stochastic interpolants: A unifying framework for flows and diffusions. *Journal of Machine Learning Research*, 26(209):1–80, 2025.

[2] Kevin Black, Manuel Y Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=UkR2zO5uww.

[3] Kaiqi Chen, Eugene Lim, Kelvin Lin, Yiyang Chen, and Harold Soh. Don't Start From Scratch: Behavioral Refinement via Interpolant-based Policy Diffusion. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.122.

[4] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[5] Hyungjin Chung, Jeongsol Kim, Michael Thompson McCann, Marc Louis Klasky, and Jong Chul Ye. Diffusion

posterior sampling for general noisy inverse problems. In *The Eleventh International Conference on Learning Representations*, 2023.

[6] Haoyuan Deng, Wenkai Guo, Qianzhun Wang, Zhenyu Wu, and Ziwei Wang. Safebimanual: Diffusion-based trajectory optimization for safe bimanual manipulation. In *9th Annual Conference on Robot Learning*, 2025.

[7] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[8] Ruiqi Feng, Chenglei Yu, Wenhao Deng, Peiyan Hu, and Tailin Wu. On the guidance of flow matching. In *Forty-second International Conference on Machine Learning*, 2025.

[9] Zeyu Feng, Hao Luan, Pranav Goyal, and Harold Soh. LTLDoG: Satisfying temporally-extended symbolic constraints for safe diffusion-based planning. *IEEE Robotics and Automation Letters*, 2024.

[10] Alexandros Graikos, Nikolay Malkin, Nebojsa Jojic, and Dimitris Samaras. Diffusion models as plug-and-play priors. *Advances in Neural Information Processing Systems*, 35:14715–14728, 2022.

[11] Ce Hao, Kelvin Lin, Zhiwei Xue, Siyuan Luo, and Harold Soh. Disco: Language-guided manipulation with diffusion policies and constrained inpainting. *IEEE Robotics and Automation Letters*, 2025.

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[13] Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, 2022.

[14] 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. In *9th Annual Conference on Robot Learning*, 2025.

[15] Jeongsol Kim, Bryan Sangwoo Kim, and Jong Chul Ye. Flowdps: Flow-driven posterior sampling for inverse problems. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12328–12337, 2025.

[16] 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.

[17] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023.

[18] Aaditya Prasad, Kevin Lin, Jimmy Wu, Linqi Zhou, and Jeannette Bohg. Consistency policy: Accelerated visuomotor policies via consistency distillation. In *Robotics: Science and Systems*, 2024.

[19] Allen Z. Ren, Justin Lidard, Lars Lien Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization. In *The Thirteenth International Conference on Learning Representations*, 2025.

[20] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

[21] Luran Wang, Chaoran Cheng, Yizhen Liao, Yanru Qu, and Ge Liu. Training free guided flow-matching with optimal control. In *The Thirteenth International Conference on Learning Representations*, 2025.

[22] Yanwei Wang, Lirui Wang, Yilun Du, Balakumar Sundaralingam, Xuning Yang, Yu-Wei Chao, Claudia Pérez-D'Arpino, Dieter Fox, and Julie Shah. Inference-time policy steering through human interactions. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15626–15633. IEEE, 2025.

[23] Zhendong Wang, Max Li, Ajay Mandlekar, Zhenjia Xu, Jiaojiao Fan, Yashraj Narang, Linxi Fan, Yuke Zhu, Yogesh Balaji, Mingyuan Zhou, Ming-Yu Liu, and Yu Zeng. One-step diffusion policy: Fast visuomotor policies via diffusion distillation. In *Forty-second International Conference on Machine Learning*, 2025.

[24] Han Xue, Jieji Ren, Wendi Chen, Gu Zhang, Yuan Fang, Guoying Gu, Huazhe Xu, and Cewu Lu. Reactive diffusion policy: Slow-fast visual-tactile policy learning for contact-rich manipulation. In *Proceedings of Robotics: Science and Systems (RSS)*, 2025.

[25] Sixu Yan, Zeyu Zhang, Muzhi Han, Zaijin Wang, Qi Xie, Zhitian Li, Zhehan Li, Hangxin Liu, Xinggang Wang, and Song-Chun Zhu. M^2 diffuser: Diffusion-based trajectory optimization for mobile manipulation in 3d scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

[26] Jiwen Yu, Yinhuai Wang, Chen Zhao, Bernard Ghanem, and Jian Zhang. Freedom: Training-free energy-guided conditional diffusion model. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 23117–23127, 2023. doi: 10.1109/ICCV51070.2023.02118.

[27] Bingliang Zhang, Wenda Chu, Julius Berner, Chenlin Meng, Anima Anandkumar, and Yang Song. Improving diffusion inverse problem solving with decoupled noise annealing. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 20895–20905, 2025.

[28] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.