Latent Barrier Steering: Hierarchical Safety for Generative Planning

Renhao Zhang
University of Massachusetts
renhaozhang@cs.umass.edu

Mingzhe Li
University of Massachusetts
mingzheli@cs.umass.edu

Bruno Castro da Silva
University of Massachusetts
bsilva@cs.umass.edu

Abstract

We introduce *Latent Barrier Steering* (LBS), a hierarchical semantic-to-path framework for safe flow-matching-based generative planning. Existing control-barrier-function (CBF) safety filters for diffusion and flow-matching planners act locally on generated samples, either during sampling or after prediction. Such repair is effective for small violations, but can struggle when a new constraint blocks the whole selected behavior mode rather than merely perturbing it locally. LBS decouples behavior selection from path certification through two coordinated interventions: it first steers the generator's behavior latent toward rollouts with larger safety margin, then decodes the steered latent and applies a path-space CBF quadratic program (QP) only for residual violations. We show that latent steering increases the margin available to the path-space corrector, yielding finite-time recovery guarantees and reduced correction burden. Experiments across navigation and robot end-effector planning tasks show that LBS preserves safety while improving goal reaching and reducing correction burden compared with path-level safety filters.

1 Introduction

Robotic path planning must combine expressive behavior generation with strong safety guarantees. Generative policies, including diffusion and flow-matching planners, are attractive because they model multimodal trajectory distributions and synthesize diverse high-dimensional behaviors from demonstrations [16, 10, 34, 23, 17, 5, 3, 2]. Yet deployment safety is not inherited from offline training: new obstacles, workspace limits, or feasibility constraints can shift the test-time problem away from the demonstrations used to train the generator. This has motivated safe generative planners that combine learned samplers with control barrier functions (CBFs), which encode safety through a barrier-defined set, and impose constraints through quadratic programs or related filtering steps [1, 13, 27, 39, 40, 8, 41]. Existing CBF-based generative planners instantiate this idea primarily as path-level repair: they inject barrier constraints into the denoising or flow-generation process, or apply a CBF correction phase after a trajectory has been generated [40, 8, 41]. In both cases, the intervention acts on sample or waypoint coordinates, pushing unsafe portions of the trajectory back toward the safe set while staying close to the generator's original output. This is effective when the generated trajectory is already nearly safe, so that only a short segment must be moved away from an obstacle. Under larger constraint shifts, however, the new constraint may block a substantial portion of the selected behavior mode. The corrector is then no longer making a local geometric repair: it must reshape a long connected segment of the rollout, which can increase intervention energy and leave the final trajectory safe but far from the goal.

When path-level repair has to reshape a long segment of a rollout, the desired solution may be another behavior that the generator already knows how to produce. A multimodal demonstration dataset can contain several valid ways to solve the same start-goal task, such as passing above, below, or between obstacles, and generative policies can represent these alternatives as diverse rollouts. In standard generative planners, however, this diversity is usually accessed through stochastic sampling: the initial

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

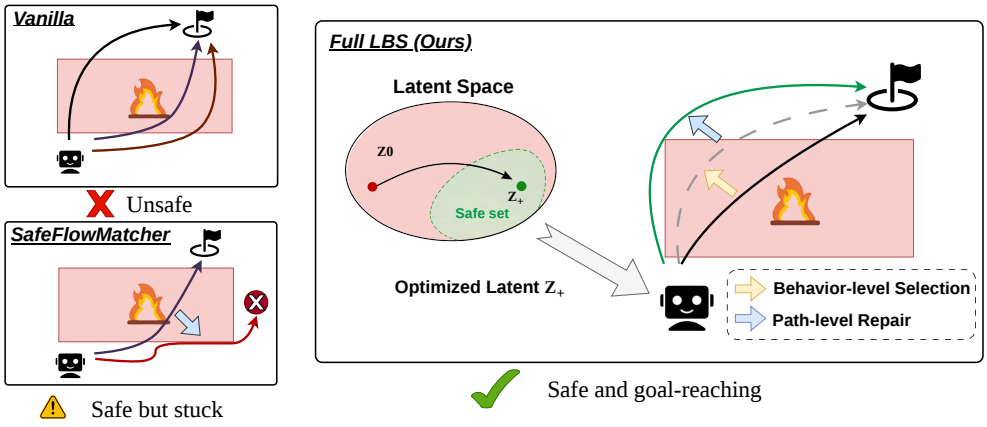


Figure 1: **Latent Barrier Steering separates behavior selection from path repair.** A vanilla flow-matching planner may generate a goal-reaching but unsafe rollout under a new constraint. A path-space CBF corrector such as SAFEFLOWMATCHER can enforce safety locally, but may produce a trajectory that is safe while no longer reaching the goal when the selected behavior mode is blocked. LBS first steers the behavior latent from z_0 to z_+ toward a safer rollout, then applies path-level CBF repair only for residual violations, yielding a trajectory that is both safe and goal-reaching.

noise or sampling seed determines which behavior appears, with no direct handle for a safety filter to choose a rollout that is easier to certify. Geometry-aligned behavior latents, as in parameterized diffusion policies (PDP), provide such a handle by learning a low-dimensional coordinate in which nearby latent values decode to similar trajectories and larger latent moves induce more substantial behavior changes [44]. This suggests a different division of labor for safe generative planning as shown in Figure 1: use the behavior latent to select or interpolate a safer rollout before asking a path-level CBF corrector to enforce the final constraints.

In this paper, we propose *Latent Barrier Steering* (LBS), a two-stage framework for safe parameterized flow matching that separates behavior selection from path certification. LBS first applies a CBF quadratic program (CBF-QP) in the behavior latent space to steer the decoded rollout toward a larger robust safety margin. It then decodes the steered latent and applies a path-space CBF-QP only to the remaining waypoint-level violations. This semantic-to-path structure lets the planner change the selected behavior before invoking local path repair. Theoretically, we show that latent steering improves the safety margin handed to the path-space corrector, yielding finite-time latent recovery when a safe latent is reachable and tighter path-level correction-time and intervention-energy certificates. Empirically, LBS improves safety, goal reaching, and correction efficiency across constrained planning tasks. Our contributions are:

- **Introduce semantic-to-path safety with parameterized flow matching.** We build on a parameterized flow-matching planner whose geometry-aligned latent indexes different rollout behaviors. LBS uses this latent as the first point of safety intervention, steering the selected behavior before applying path-space CBF certification.
- **Analyze when latent steering reduces path-space correction.** We show that increasing the latent safety margin gives the path-space corrector a less-violating warm start. This yields finite-time latent recovery when a safe latent is reachable, and tighter correction-time and intervention-energy certificates for the downstream CBF-QP.
- **Demonstrate safer and more goal-directed constrained planning.** Across constrained planning tasks, LBS preserves safety while improving goal reaching and reducing correction energy compared with path-level safety filters.

2 Related Work

Generative policies for planning. Diffusion and flow-based generative models provide expressive trajectory priors for robot planning and control, with diffusion planners modeling flexible multimodal behavior through iterative denoising and flow-matching planners improving sampling efficiency through deterministic transport [16, 10, 34, 23, 24, 17, 5, 37, 29, 3, 2]. These models have been applied

to offline RL, visuomotor imitation, navigation, manipulation, and diverse human-demonstration datasets [25, 33, 18]. Their strength is that a single learned sampler can represent many plausible trajectories for the same task context. However, the generated behavior is usually selected by conditioning variables or stochastic noise, so multimodal diversity does not by itself provide a controllable safety mechanism under new constraints.

Safety certification for generative planners. Control barrier functions (CBFs) and related invariance tools provide a standard way to enforce safety by maintaining the forward invariance of a superlevel set, often through a quadratic program that minimally modifies a nominal command [1, 13, 27, 39]. Recent safe generative planners attach this machinery to diffusion or flow sampling: SAFEDIFFUSER embeds finite-time invariance into the diffusion process, SAFEFLOW introduces flow-matching barrier functions for safe trajectory generation, and SAFEFLOWMATCHER decouples prediction from correction by applying a finite-time CBF certificate only to the final path-space correction phase [40, 8, 41]. These methods provide formal safety mechanisms at test time, but their interventions are still expressed in sample or waypoint coordinates. They therefore primarily address local path repair after the generator has already selected a behavior.

Behavior latents and controllable multimodality. Controllable multimodality has a long history in parameterized actions, options, and skills, where a policy selects both a discrete behavior and continuous parameters that adapt the behavior to a task instance [36, 20, 7, 15, 26, 32, 9, 11, 42, 14, 45]. Related work on skill discovery and latent-action imitation learns reusable behavior abstractions from demonstrations or interaction data [28, 33, 21, 31, 38]. PDP brings this idea to diffusion policies by learning a continuous behavior latent whose geometry is aligned with trajectory similarity, turning noise-driven diversity into an optimizable behavior handle [44]. These threads motivate our use of a behavior latent as the semantic coordinate for safety-aware steering before path-level certification.

3 Background

3.1 Parameterized flow matching for trajectory planning

Let $\mathcal{D}$ denote the waypoint space, and let $\tau = (\tau^0, \dots, \tau^H) \in \mathcal{D}^{H+1}$ denote a planning trajectory over horizon H under context c. Flow matching learns a time-dependent vector field $v_t(\cdot, c; \theta)$ whose ODE transports a simple prior p_0 to the demonstration distribution $q(\tau \mid c)$ [23, 24]: $\frac{d}{dt}\psi_t(\tau_0, c) = v_t(\psi_t(\tau_0, c), c; \theta)$, with $\tau_0 \sim p_0$ and $\psi_1(\tau_0, c) \sim q(\cdot \mid c)$. Following conditional flow matching, a standard objective samples $\tau_0 \sim p_0$, $\tau_1 \sim q(\cdot \mid c)$, $t \sim \mathrm{Unif}[0, 1]$, sets $\tau_t = (1-t)\tau_0 + t\tau_1$, and regresses to the optimal-transport velocity:

$$\mathcal{L}_{\mathrm{FM}}(\theta) = \mathbb{E}\big[\, \|v_t(\tau_t, c; \theta) - (\tau_1 - \tau_0)\|_2^2 \,\big]. \tag{1}$$

To expose a controllable behavior coordinate, we follow parameterized diffusion policies and learn a trajectory encoder $q_\phi(z \mid \tau)$ for a low-dimensional latent $z \in \mathcal{Z} \subset \mathbb{R}^{d_z}$ [44]:

$$q_\phi(z \mid \tau) = \mathcal{N}\big(\mu_\phi(\tau), \mathrm{diag}(\sigma_\phi^2(\tau))\big), \qquad \bar{z}(\tau) = \mu_\phi(\tau), \tag{2}$$

where $\bar{z}(\tau)$ is the deterministic code used for conditioning. The generator produces $\tau(z; \xi) = G_\theta(c, z, \xi)$, with stochastic seed ξ or deterministic decode $\hat{\tau}_\theta(z, c)$. With latent prior $p(z)$ and nonnegative weights $\lambda_{\mathrm{rec}}, \lambda_{\mathrm{KL}}, \lambda_{\mathrm{geo}}$, we write a generic parameterized flow-matching objective as

$$\begin{aligned}\mathcal{L}_{\text{PFM}}(\theta, \phi) = \mathcal{L}_{\mathrm{FM}}(\theta; z, c) &+ \lambda_{\mathrm{rec}}\, \mathbb{E}_{\tau, z \sim q_\phi}\big[\, \|\tau - \hat{\tau}_\theta(z, c)\|_2^2 \,\big] \\ &+ \lambda_{\mathrm{KL}}\, \mathbb{E}_\tau\big[D_{\mathrm{KL}}(q_\phi(z \mid \tau) \,\|\, p(z))\big] + \lambda_{\mathrm{geo}}\, \mathcal{L}_{\mathrm{geo}},\end{aligned} \tag{3}$$

where the geometry loss aligns trajectory and latent distances. For a sampled pair (τ_i, τ_j), PDP computes a trajectory distance $\delta_{ij}^X = d_{\mathrm{softDTW}}(X(\tau_i), X(\tau_j))$ between behavior-trace feature sequences $X(\tau_i)$ and $X(\tau_j)$ using soft-DTW [6], and a latent distance $\delta_{ij}^z = \|\bar{z}(\tau_i) - \bar{z}(\tau_j)\|_2$. It then penalizes $\mathcal{L}_{\mathrm{geo}} = \mathbb{E}_{(i,j)}[(\delta_{ij}^z - \kappa_{\mathrm{geo}}\delta_{ij}^X)^2]$, with scale $\kappa_{\mathrm{geo}} > 0$. This geometry term encourages z to act as a behavior coordinate: nearby latents decode to similar trajectory traces, while larger latent displacements correspond to larger behavioral changes.

3.2 Control barrier functions for path safety

For a control-affine system $\dot{x} = f(x) + g(x)u$ with state $x \in \mathcal{D}$ and input $u \in \mathcal{U}$, and a C^1 barrier $b : \mathcal{D} \to \mathbb{R}$, a control barrier function defines safety as $\mathcal{C} = \{x : b(x) \geq 0\}$ and enforces forward invariance through the inequality $L_f b(x) + L_g b(x)u + \alpha(b(x)) \geq 0$, where L_f, L_g are Lie derivatives and α is a class-$\mathcal{K}$ rate [1, 13]. Given a nominal input u^{nom}, the standard CBF-QP chooses the closest safe input:

$$u^\star = \arg\min_{u \in \mathcal{U}} \|u - u^{\mathrm{nom}}\|_2^2 \quad \text{s.t.} \quad L_f b(x) + L_g b(x)u + \alpha(b(x)) \geq 0 \tag{4}$$

For states that may start outside $\mathcal{C}$, fractional-power rates $\alpha(b) = \kappa_{\mathrm{cbf}} \operatorname{sgn}(b)|b|^\rho$ with rate constant $\kappa_{\mathrm{cbf}} > 0$ and fractional-power exponent $0 < \rho < 1$ give finite-time convergence into $\mathcal{C}$ [27, 39, 40].

Safe generative planners apply such constraints during diffusion or flow sampling, or as a correction phase after unconstrained prediction [40, 8, 41]. SAFEFLOWMATCHER's hard-correction phase satisfies, at sampling time t, for corrected waypoint $\tau_t^{c,k}$ (waypoint k of the path at time t) and correction input u_t^k,

$$\nabla b(\tau_t^{c,k})^\top u_t^k + \epsilon \operatorname{sgn}(b(\tau_t^{c,k}) - \delta)\, |b(\tau_t^{c,k}) - \delta|^\rho + w_t^k r_t^k \geq 0, \tag{5}$$

where $\epsilon > 0$ is the convergence rate, $0 < \rho < 1$ is the same fractional-power exponent as in (4), $\delta \geq 0$ is the robustness margin, and $w_t^k \geq 0$ is a vanishing weight schedule paired with a per-waypoint slack variable r_t^k that together deactivate after switch time t_w. The corrector enters $\mathcal{C}_\delta = \{b \geq \delta\}$ within correction time

$$T_{\mathrm{SFM}} \leq t_w + \frac{(\delta - b(\tau_{t_w}^{c,k}))^{1-\rho}}{\epsilon(1-\rho)}. \tag{6}$$

This bound depends on the pre-correction barrier margin, the downstream quantity tightened in Section 4.

4 Latent Barrier Steering

We introduce Latent Barrier Steering (LBS), a two-stage safety framework for generative planning that separates behavior-level steering from path-level certification. Prior CBF-based generative planners apply certified correction after a plain generator has already selected a behavior mode, leaving the corrector to repair the resulting rollout in waypoint or sample space. LBS instead builds on a parameterized flow-matching planner whose geometry-aligned behavior latent indexes different task-solving rollouts. At test time, Stage I treats this latent as the first safety intervention point: given a differentiable barrier function for the deployment constraint, it steers the latent toward a decoded rollout with a larger worst-case barrier margin. Stage II then uses this decoded rollout as a safety-aware warm start for a downstream path-space CBF-QP corrector, which enforces the remaining waypoint constraints under standard barrier-certificate conditions.

4.1 LBS Objective with a Geometry-Aligned Latent

Stage I steers the behavior coordinate of a parameterized flow-matching planner. For a fixed task context c, each latent $z \in \mathcal{Z}$ decodes to a rollout

$$\tau(z) = G_\theta(c, z) = \big(\tau^0(z), \ldots, \tau^H(z)\big) \in \mathcal{D}^{H+1}.$$

We use a deterministic decode $\tau(z)$ by fixing the sampling randomness. The geometry-regularized objective in Section 3.1 encourages local latent moves to induce controlled rollout changes. Because the encoder is KL-regularized toward the prior $p(z) = \mathcal{N}(0, I)$, Stage I restricts steering to a compact region $\Omega \subseteq \mathcal{Z}$ around the prior center, implemented by latent clipping. The prior is used only to organize the latent space during training. At test time, Stage I optimizes a single latent within Ω, which contains the encoded demonstration modes and their interpolation region. On this region, we assume the decoder and barrier are locally Lipschitz. This is an analysis assumption encouraged by the geometry objective rather than guaranteed by it (Appendix B.4). Concretely, for $z, z' \in \Omega$ and $x, x' \in \mathcal{D}$,

$$\|\tau(z) - \tau(z')\| \leq L_\tau \|z - z'\|, \qquad |b(x) - b(x')| \leq L_b \|x - x'\|.$$

This local geometry is the condition that makes latent-space safety steering meaningful: a controlled update in z produces a controlled change in the decoded rollout.

We now define the latent safety objective. Using the barrier b and robustness margin δ from Section 3.2, Stage I assigns each latent a differentiable approximation to the worst shifted waypoint margin of its decoded rollout:

$$m_\beta(z) := -\frac{1}{\beta} \log \sum_{k=0}^{H} \exp\big(-\beta\big(b(\tau^k(z)) - \delta\big)\big), \qquad \beta > 0.$$

This soft-min margin satisfies

$$\min_k \big(b(\tau^k(z)) - \delta\big) - \frac{\log(H+1)}{\beta} \le m_\beta(z) \le \min_k \big(b(\tau^k(z)) - \delta\big),$$

so $\varepsilon_{\text{soft}} = \log(H+1)/\beta$ is the only gap between m_β and the true worst-waypoint margin; see Appendix B.1. Combining this construction with the local geometry above gives

$$|m_\beta(z) - m_\beta(z')| \le L_b L_\tau \|z - z'\|, \qquad z, z' \in \Omega,$$

with proof in Appendix B.2. Thus, controlled motion in the behavior latent produces controlled changes in the rollout safety margin. The strongest Stage I target is the safe-latent set $\mathcal{Z}_\delta := \{z \in \mathcal{Z} : m_\beta(z) \ge 0\}$, whose elements decode to rollouts satisfying the buffered waypoint constraint up to the soft-min gap. LBS, however, does not require Stage I to reach this set before handoff: increasing m_β already reduces the barrier deficit that Stage II must correct. This is the margin quantity used in Section 4.2 for latent CBF steering and in Section 4.3 to relate latent progress to the path-space correction burden. Because m_β is defined over discrete waypoints, all guarantees following are stated at the waypoint level. Appendix B.3 extends these margins to continuous path segments.

4.2 Stage I: Latent CBF Steering

Stage I uses the latent margin m_β from Section 4.1 as a barrier over the behavior latent. Rather than modifying waypoints independently, it updates z so that the pretrained decoder produces a rollout with larger safety margin before path-space correction. We model latent steering as $\dot{z} = u_z$ and choose the smallest modification to a reference latent update u_z^{ref}. In our implementation, $u_z^{\text{ref}}(z) = -\gamma_z z$ with $\gamma_z > 0$, which mildly contracts the latent toward the prior center and discourages off-support drift. At each latent z, LBS solves

$$u_z^\star(z) = \arg\min_{u \in \mathbb{R}^{d_z}} \|u - u_z^{\text{ref}}(z)\|_2^2 \quad \text{s.t.} \quad \nabla m_\beta(z)^\top u + \kappa \, \text{sgn}(m_\beta(z)) |m_\beta(z)|^\rho \ge 0, \tag{7}$$

where $\kappa > 0$ and $0 < \rho < 1$ are the CBF rate and exponent. Because (7) has one affine constraint, it is the projection of u_z^{ref} onto a half-space. Let $\psi(z) := \nabla m_\beta(z)^\top u_z^{\text{ref}}(z) + \kappa \, \text{sgn}(m_\beta(z)) |m_\beta(z)|^\rho$. Then, whenever $\nabla m_\beta(z) \neq 0$,

$$u_z^\star(z) = u_z^{\text{ref}}(z) + \frac{[-\psi(z)]_+}{\|\nabla m_\beta(z)\|_2^2} \nabla m_\beta(z). \tag{8}$$

Thus, Stage I follows the reference update when it already satisfies the latent barrier condition, and otherwise adds the minimum correction along ∇m_β; the derivation is in Appendix C.1.

Proposition 1 (Latent margin decrease)**.** *Let $V(z) := [-m_\beta(z)]_+$. Along the Stage I flow $\dot{z} = u_z^\star(z)$, whenever $m_\beta(z) < 0$ and $\nabla m_\beta(z) \neq 0$,*

$$\dot{V}(z(t)) \le -\kappa V(z(t))^\rho.$$

Proposition 1 says that Stage I decreases the margin deficit whenever the unsafe decoded rollout has a first-order latent direction for improvement. The proof is a direct consequence of the CBF constraint and is given in Appendix C.2. When this first-order sensitivity is uniformly nonzero on the unsafe part of Ω, the same fractional-power rate gives finite-time recovery.

Theorem 1 (Finite-time Stage-I recovery)**.** *Suppose the latent flow $\dot{z} = u_z^\star(z)$ remains in Ω before reaching $\mathcal{Z}_\delta$, and suppose there exists $g_m > 0$ such that $\|\nabla m_\beta(z)\|_2 \ge g_m$ for all $z \in \Omega$ with $m_\beta(z) < 0$. Then, for any $z_0 \in \Omega$ with $m_\beta(z_0) < 0$, $z(t) \in \mathcal{Z}_\delta$ for all $t \ge T_z := |m_\beta(z_0)|^{1-\rho}/(\kappa(1-\rho))$. Moreover, $\mathcal{Z}_\delta$ is forward-invariant.*

The proof is in Appendix C.3. In practice, Stage I uses projected Euler steps with either backpropagated or finite-difference estimates of ∇m_β; implementation details and generator-call cost are given in Appendix E.

4.3 Stage II Correction and Handoff

Stage I returns a steered latent z_+ and decoded rollout $\tau(z_+)$. Stage II then applies the path-space CBF-QP from Section 3.2 to certify the remaining waypoint-level violations. The key question is whether improving the latent margin m_β actually reduces the path-space violation.

Let $\tilde{b}_{t_w}(z) := \min_k(b(\tau^{c,k}_{t_w}(z)) - \delta)$ denote the shifted path margin at the start of hard correction, where t_w is the switch time used by the path-space corrector. For predictor–corrector methods, the switch-time path may differ from the decoded rollout; write this deviation as $\varepsilon_{\text{pred}}(z) := \max_k \|\tau^{c,k}_{t_w}(z) - \tau^k(z)\|_2$. For a post-hoc corrector applied directly to $\tau(z)$, $\varepsilon_{\text{pred}}(z) = 0$.

Proposition 2 (Latent-to-path margin transfer)**.** *Under the local regularity condition in Section 4.1, for all $z \in \Omega$,*

$$|\tilde{b}_{t_w}(z) - m_\beta(z)| \le \varepsilon_{\text{soft}} + L_b\varepsilon_{\text{pred}}(z), \qquad \varepsilon_{\text{soft}} = \frac{\log(H+1)}{\beta}.$$

Proposition 2 is the formal handoff from latent steering to path correction: increasing $m_\beta(z)$ improves the initial path-space margin up to the soft-min gap and any predictor deviation. The proof is in Appendix D.1. Thus, Stage I does not need to certify the trajectory by itself; it only needs to reduce the deficit that Stage II starts from.

Corollary 1 (Reduced Stage-II correction burden)**.** *Let $V_+ := [-\tilde{b}_{t_w}(z_+)]_+$ be the initial Stage-II barrier deficit. With Stage-II CBF rate $\epsilon > 0$ and exponent $\rho \in (0,1)$, the path-space corrector satisfies*

$$T_{\text{corr}}(z_+) \le t_w + \frac{V_+^{1-\rho}}{\epsilon(1-\rho)}.$$

If the squared intervention magnitude is bounded by $c_{II}V_{\text{path}}^{2\rho}$ during correction, then

$$J_{II}(z_+) \le \frac{c_{II}V_+^{1+\rho}}{\epsilon(1+\rho)}.$$

Corollary 1 shows why the latent stage is useful even when it does not reach $\mathcal{Z}_\delta$: by decreasing V_+, Stage I tightens both the certified correction-time bound and the intervention-energy bound. The proof follows the standard fractional-power CBF argument and is given in Appendix D.2.

It remains to choose when to stop latent steering. Running Stage I longer reduces the terminal deficit but incurs additional latent-steering cost. In the exact-gradient limit, the descent law from Proposition 1 gives $V_+^{1-\rho} = V_0^{1-\rho} - \eta\kappa(1-\rho)N_z$, where $V_0 := [-m_\beta(z_0)]_+$. Combining this decrease with a per-step Stage-I cost c_I and the Stage-II energy bound above yields a cost-sensitive stopping rule.

Theorem 2 (Cost-sensitive Stage-I handoff)**.** *Suppose the Stage-I decrease law holds along the latent trajectory and the margin-transfer slack in Proposition 2 is absorbed into the Stage-II deficit. For $V_0 > 0$, the resulting upper bound on total cost is minimized at $V_+^\star = (c_I\epsilon/(\eta\kappa c_{II}))^{1/(2\rho)}$. The corresponding handoff iteration is $N_z^\star = (V_0^{1-\rho} - (V_+^\star)^{1-\rho})/(\eta\kappa(1-\rho))$ when $V_+^\star < V_0$; otherwise, LBS hands off to Stage II immediately.*

Theorem 2 says that Stage I should stop once another latent update is expected to cost more than the correction energy it saves downstream. The convexity argument and the explicit total-cost expression are deferred to Appendix D.3.

5 Experiments

We evaluate whether combining behavior-latent steering with path-space CBF certification improves constrained flow-matching planning. The experiments are organized around four questions. **Q1:** How does LBS compare with existing safe generative planners across different domains and test-time safety constraints? **Q2:** How does LBS change the generated behavior before path-space correction? **Q3:** How effective is Stage I's latent steering design compared with alternative ways of selecting the behavior latent? **Q4:** What does Stage I achieve on its own, and what does it hand off to Stage II?

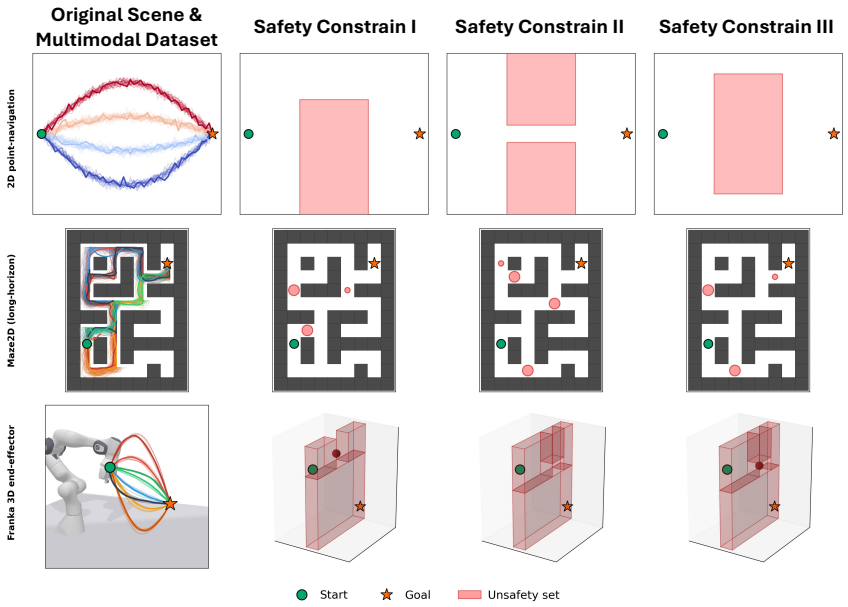


Figure 2: **Experimental domains and test-time safety constraints.** Each row shows one planning domain: 2D point-navigation, Maze2D, and Franka 3D end-effector planning. The first column visualizes the original scene and multimodal demonstration dataset, where multiple behaviors can reach the same goal. The remaining columns show three test-time safety constraints imposed on the same pretrained planner. Green circles denote starts, orange stars denote goals, and red shaded regions denote unsafe sets. These constraints test whether a method can preserve safety and goal reaching when certification requires more than local path repair.

Setup. We evaluate on constrained planning tasks spanning low-dimensional navigation, long-horizon maze navigation, and robot end-effector planning. In each domain, the pretrained flow-matching planner is trained on a multimodal dataset with several feasible ways to reach the goal. At test time, we keep the pretrained planner fixed and introduce three different safety constraints, shown in Figure 2. These constraints vary how much the originally generated rollout must change before it can be certified, but the method is not given this structure explicitly. Full domain construction, obstacle layouts, model architectures, and training details are provided in Appendix F.

Baselines and Metrics. We compare LBS against vanilla flow-matching planner, SAFEDIFFUSER [40], SAFEFLOW [8], and SAFEFLOWMATCHER [41], which corresponds to using Stage II without latent steering. We report Barrier Safety (referred to as Safety below, with Maze2D walls handled as described in Appendix F.3), Reach, final goal distance (ED), CBF intervention Energy, curvature smoothness (CS), and per-episode Stage-I/Stage-II runtime. Together, these metrics distinguish whether a method is safe, goal-reaching, or both, with low correction burden.

5.1 Performance Comparison Across Domains and Constraints

Table 1 reports Safety, Reach, and intervention Energy for all three constraints in each domain. Final goal distance, curvature smoothness, and runtime are reported in Appendix I. This comparison evaluates whether LBS improves the overall safety–performance tradeoff relative to existing path-level safety filters under the same pretrained planner and test-time constraints.

Across domains and constraints, the table shows a clear separation between goal-directed generation and certified safety. The unconstrained flow-matching planner often preserves the task behavior learned from demonstrations, so its rollouts remain close to the goal, but it has no mechanism to adapt that behavior to new safety constraints. This leads to goal-reaching trajectories that are frequently unsafe. Existing CBF-based safety filters correct this failure in the opposite direction: they reliably improve safety, but because the intervention is applied in sample or waypoint space after a behavior has already been selected, the corrected trajectory can lose the structure needed to complete the

Table 1: **Performance comparison across domains and test-time safety constraints.** Each domain uses the same pretrained flow-matching planner across methods. Final goal distance, curvature smoothness, and runtime are reported in Appendix I.

		2D			Franka			Maze2D		
Constraint	Method	Safety↑	Reach↑	Energy↓	Safety↑	Reach↑	Energy↓	Safety↑	Reach↑	Energy↓
I	Vanilla FM	12%	**100%**	–	12%	**100%**	–	0%	50%	–
	SAFEDIFFUSER	**100%**	10%	1.226	**100%**	0%	0.0023	**100%**	0%	0.0288
	SAFEFLOW	**100%**	18%	1.077	**100%**	16%	0.0020	**100%**	0%	0.1208
	SAFEFLOWMATCHER	**100%**	18%	1.166	**100%**	14%	0.0075	**100%**	0%	0.2668
	Full LBS	**100%**	**100%**	**0.037**	**100%**	**100%**	**0.0004**	**100%**	**95%**	**0.010**
II	Vanilla FM	0%	**100%**	–	0%	**100%**	–	38%	50%	–
	SAFEDIFFUSER	**100%**	34%	0.728	**100%**	0%	0.0023	**100%**	7%	**0.0010**
	SAFEFLOW	**100%**	20%	0.976	**100%**	0%	0.0023	**100%**	25%	0.0624
	SAFEFLOWMATCHER	**100%**	22%	1.051	**100%**	0%	0.0086	98%	18%	0.1404
	Full LBS	**100%**	90%	**0.160**	**100%**	**100%**	**0.0004**	**100%**	**90%**	0.009
III	Vanilla FM	0%	**100%**	–	0%	**100%**	–	28%	50%	–
	SAFEDIFFUSER	**100%**	0%	1.321	**100%**	0%	**0.0023**	**100%**	7%	0.0072
	SAFEFLOW	**100%**	4%	1.262	**100%**	0%	**0.0023**	**100%**	38%	0.0345
	SAFEFLOWMATCHER	**100%**	4%	1.366	**100%**	0%	0.0086	**100%**	53%	0.0740
	Full LBS	**100%**	82%	**0.303**	**100%**	98%	0.0028	**100%**	**93%**	**0.004**

task. This is reflected not only in reduced reach, but also in larger final distance (Table 6) and higher correction energy.

Full LBS changes this pattern by intervening before path-space certification: the latent stage can change the selected behavior, giving Stage II a rollout that is closer to both the safe set and the task solution. This explains why LBS preserves goal reaching while maintaining high safety, and why its CBF intervention energy is lower in most settings. Because all methods use the same pretrained planner within each domain, the gains come from where safety intervention is applied at test time rather than from a stronger generator or additional task-specific training. The runtime columns in Table 6 reflect the current implementation tradeoff: Stage I adds overhead due to zeroth-order finite-difference gradients, while Stage II remains comparable to path-level baselines. This overhead grows linearly with the behavior-latent dimension d_z (Appendix E.3), which is set by the number of demonstrated behavior modes rather than by the horizon or robot degrees of freedom ($d_z \leq 4$ in all domains), and the resulting 0.35–$4.50\,\mathrm{s}$ of Stage I is incurred once before execution. Thus, under the present implementation, the main efficiency gain appears as reduced correction burden rather than lower wall-clock time; cheaper latent-gradient estimators should reduce the Stage I cost while preserving the qualitative advantage.

5.2 Ablation: Latent Steering Provides the Warm Start

The main comparison shows that LBS improves the safety–goal-reaching tradeoff, but the table alone does not explain how the two stages interact. We therefore visualize a representative episode in Figure 3, comparing path-space correction with and without the latent-steering stage. The figure is organized as two rows with five columns. The first row shows Full LBS: the rollout is first decoded from the initial behavior latent, then updated through Stage I, and finally passed to Stage II for path-space CBF correction. The second row removes Stage I and applies the same Stage II corrector directly to the original decoded rollout. The first panel of Full LBS is therefore the same rollout as the Stage-II-only warm start shown in the third panel of the second row.

This visualization isolates the role of Stage I. Before path-space correction, the initial rollout is unsafe and poorly positioned for local repair. When Stage II starts directly from this cold rollout, the CBF-QP can push the trajectory toward the safe set, but the correction remains tied to the originally selected behavior and fails to recover a goal-reaching path. In contrast, Stage I changes the decoded behavior before certification: by the third column of the LBS row, the rollout is not necessarily fully safe, but it is already in a much better region for downstream correction. Starting from this latent-steered warm start, Stage II only needs to remove residual waypoint-level violations, and the final trajectory remains both safe and task-completing.

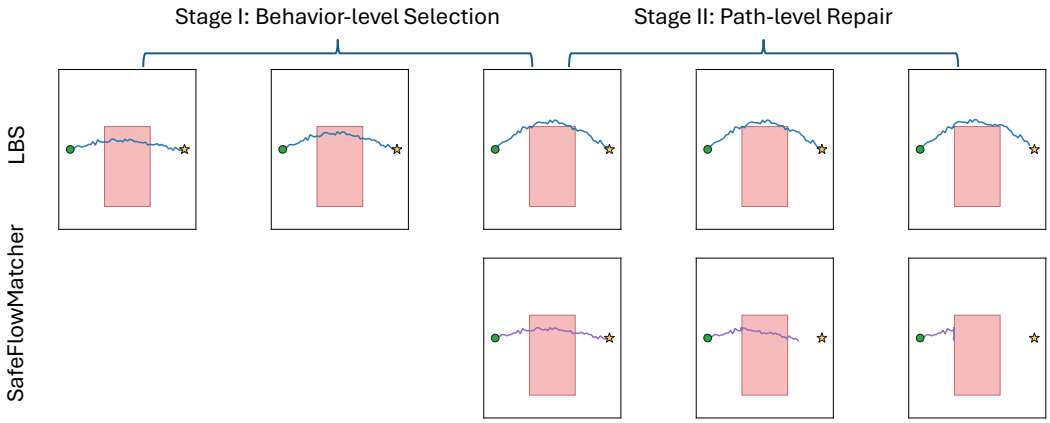


Figure 3: **Latent steering provides the warm start needed for path-space repair.** Stage I performs behavior-level selection by steering the latent and changing the decoded rollout before certification; Stage II then applies path-level CBF repair to remove residual violations. The second row removes Stage I and applies the same path-space correction directly, corresponding to SAFEFLOWMATCHER. Without the latent-steered warm start, the local corrector remains tied to the original rollout and fails to produce a goal-reaching safe path. With Stage I, the corrected trajectory starts from a better behavior and Stage II only needs to repair the remaining local violation.

This ablation supports the central semantic-to-path design of LBS. Stage I is not intended to replace CBF certification, and Stage II is not sufficient when initialized from a behavior that is incompatible with the new constraint. The improvement comes from their composition: latent steering changes the selected behavior at the level of the generator, while path-space correction certifies the remaining local violations. The comparison also explains the lower intervention energy observed in Table 1: after Stage I, the corrector starts from a less-violating and more goal-directed rollout, so it does not need to perform a large deformation in waypoint space.

5.3 Ablation Study of the Stage-I Latent Steering

Table 2 compares ways of selecting the latent before path-space correction. All methods use the same pretrained generator, the same safety margin $m_\beta(z)$, and the identical Stage-II corrector:

- **Best-of-N (BoN):** sample N latents from the prior and select the decoded rollout with the largest m_β. We report $N \in \{1, 4, 16, 64\}$ (BoN-N).
- **Plain latent gradient ascent (PLGA):** directly ascend $m_\beta(z)$, with the best step size per domain from $\{10^{-3}, \ldots, 1\}$.

PLGA and LBS share the same initial latent, gradient estimator, clipping region Ω, early-stopping criterion, and Stage-I wall-clock budget: the update rule is the only difference. We also report *Stage I only*, which decodes the steered latent without Stage-II correction. Table 2 supports three observations:

- **Neither stage suffices alone.** Stage II alone is safe but rarely reaches the goal, whereas Stage I alone preserves reach but leaves residual violations. Stage I does not need to reach the buffered safe-latent set for the two to combine well: its terminal margin is still negative in most episodes, yet it shrinks the deficit handed to Stage II (Section 5.4).
- **Increasing N does not recover goal reaching.** BoN reaches the goal in at most 48% of episodes for any N, and Reach is non-monotonic in N (in Maze2D, 48% at $N = 4$ but 0% at $N = 64$). BoN can only choose among behaviors exposed by prior sampling, and selecting by safety margin among them favors conservative behaviors that do not complete the task.
- **The CBF-QP update outperforms plain gradient ascent.** PLGA, which does optimize the latent, closes most of the gap, but LBS achieves higher Reach in all three domains, with the largest gap in Maze2D, where preserving task-level structure during safety optimization is hardest. PLGA always moves along $\nabla_z m_\beta$, so it keeps pushing the latent toward higher-margin, often

more conservative behaviors even when the margin is already improving fast enough. The CBF-QP instead follows the reference update u_z^{ref} whenever it already satisfies the prescribed margin-improvement condition, and otherwise adds only the smallest correction along $\nabla_z m_\beta$ needed to satisfy it (Eq. (8)). This better preserves goal-directed behavior when aggressive margin ascent conflicts with progress toward the goal.

Table 2: **Alternatives to Stage I** (Constraint III). All methods share the pretrained generator, the safety margin m_β, and the Stage-II corrector.

	2D		Franka		Maze2D	
Method	Safety↑	Reach↑	Safety↑	Reach↑	Safety↑	Reach↑
Stage II only (SAFEFLOWMATCHER)	**100%**	4%	**100%**	0%	**100%**	53%
Stage I only	10%	**100%**	10%	**100%**	20%	**95%**
BoN-1 + Stage II	**100%**	0%	**100%**	0%	**100%**	0%
BoN-4 + Stage II	**100%**	0%	**100%**	0%	90%	48%
BoN-16 + Stage II	**100%**	0%	**100%**	32%	**100%**	5%
BoN-64 + Stage II	**100%**	0%	**100%**	36%	98%	0%
PLGA + Stage II	**100%**	76%	**100%**	92%	**100%**	68%
Full LBS	**100%**	82%	**100%**	98%	**100%**	**93%**

5.4 What Stage-I Latent Steering Hands Off to Stage II

Section 5.3 shows that Stage-I latent steering yields the best safety–reach trade-off once combined with Stage II, but not what the steering itself changes before the handoff. We therefore track the Stage-I latent trajectory directly (Table 3): how much the latent margin improves, whether the steering gradient remains informative, and how far from the buffered safe-latent set Stage I leaves the latent when Stage II takes over. The gradient norms show no systematic collapse in the evaluated regions, and the latent margin improves substantially in every domain, including 2D point-navigation, where clipping activates in every episode. The terminal margin nevertheless remains negative in most episodes: Stage I typically does not reach the buffered safe-latent set on its own, but hands Stage II a much smaller deficit. This is the warm-start role analyzed in Proposition 2 and Corollary 1.

Table 3: **Stage-I diagnostics** for Constraint III in each domain. *Initial* $\to$ *terminal* m_β: median buffered latent margin before and after Stage I. $\|\widehat{\nabla m_\beta}\|_2$: median norm of the estimated pulled-back gradient along the Stage-I iterates. *Clip frequency*: fraction of episodes in which the projection onto Ω activates at least once. *Terminal* $m_\beta < 0$: fraction of episodes whose buffered margin is still negative when Stage I hands off to Stage II.

Domain	Initial $\to$ terminal m_β	$\|\widehat{\nabla m_\beta}\|_2$	Clip frequency	Terminal $m_\beta < 0$
2D	$-1.03 \to -0.08$	0.46	100%	90%
Franka	$-0.21 \to -0.006$	0.09	8%	84%
Maze2D	$-0.63 \to -0.11$	0.18	22%	88%

6 Limitations and Conclusion

We presented *Latent Barrier Steering* (LBS), a two-stage framework for safe flow-matching planning that steers a geometry-aligned behavior latent before path-space CBF certification. LBS gives the path-space corrector a safer, more goal-directed warm start, reducing the need to reshape an unsuitable rollout locally. Our analysis shows that latent margin improvement reduces the Stage II deficit, yielding finite-time recovery conditions and tighter correction-time and intervention-energy certificates. Empirically, LBS improves the safety–goal-reaching tradeoff over path-level safety filters across constrained planning tasks. The safety guarantee covers the planned path certified by Stage II. For Franka, this is the planned end-effector path, so tracking error and full-body collisions are outside its scope. The finite-time Stage I result depends on local regularity and non-vanishing pulled-back gradients of the learned latent-to-trajectory map, which the training objective encourages but does not certify. The current zeroth-order Stage I implementation also adds runtime overhead. Cheaper gradient estimators or learned margin critics are natural next steps.

Acknowledgments and Disclosure of Funding

This work used computational resources and services provided by the Unity Research Computing Platform. We also thank the anonymous reviewers for their comments and suggestions, which helped strengthen and clarify this work.

References

[1] Aaron D. Ames, Xiangru Xu, Jessy W. Grizzle, and Paulo Tabuada. Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control*, 62(8):3861–3876, 2017.

[2] Max Braun, Noémie Jaquier, Leonel Rozo, and Tamim Asfour. Riemannian flow matching policy for robot motion learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024. arXiv:2403.10672.

[3] João Carvalho, An T. Le, Mark Baierl, Dorothea Koert, and Jan Peters. Motion planning diffusion: Learning and planning of robot motions with diffusion models. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023.

[4] Shreyas Chaudhari, Renhao Zhang, Philip S. Thomas, and Bruno Castro da Silva. Data selection for reward labeling in limited-feedback reinforcement learning. *Transactions on Machine Learning Research*, 2026.

[5] Cheng Chi, Siyuan Feng, Yilun Du, Zhenghao Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems*, 2023.

[6] Marco Cuturi and Mathieu Blondel. Soft-DTW: a differentiable loss function for time-series. In *International Conference on Machine Learning*, 2017.

[7] Bruno C. Da Silva, George Konidaris, and Andrew G. Barto. Learning parameterized skills. In *International Conference on Machine Learning*, 2012.

[8] Xiaobing Dai, Zewen Yang, Dian Yu, Fangzhou Liu, Hamid Sadeghian, Sami Haddadin, and Sandra Hirche. Safeflow: Safe robot motion planning with flow matching via control barrier functions. *arXiv preprint arXiv:2504.08661*, 2025.

[9] Murtaza Dalal, Deepak Pathak, and Ruslan Salakhutdinov. Accelerating robotic reinforcement learning via parameterized action primitives. In *Advances in Neural Information Processing Systems*, 2021.

[10] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat GANs on image synthesis. In *Advances in Neural Information Processing Systems*, 2021.

[11] Haotian Fu, Sam Yu, Siddharth Tiwari, Michael L. Littman, and George Konidaris. Meta-learning parameterized skills. In *International Conference on Machine Learning*, 2023.

[12] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4RL: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[13] Paul Glotfelter, Jorge Cortés, and Magnus Egerstedt. Nonsmooth barrier functions with applications to multi-robot systems. *IEEE Control Systems Letters*, 1(2):310–315, 2017.

[14] Vedant Gupta, Haotian Fu, Calvin Luo, Yiding Jiang, and George Konidaris. Learning parameterized skills from demonstrations. In *Advances in Neural Information Processing Systems*, 2025.

[15] Matthew J. Hausknecht and Peter Stone. Deep reinforcement learning in parameterized action space. In *International Conference on Learning Representations*, 2016.

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 2020.

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

[18] Xiaogang Jia, Denis Blessing, Xinkai Jiang, Moritz Reuss, Atalay Donat, Rudolf Lioutikov, and Gerhard Neumann. Towards diverse behaviors: A benchmark for imitation learning with human demonstrations. In *International Conference on Learning Representations*, 2024.

[19] Hassan K. Khalil. *Nonlinear Systems*. Prentice Hall, 3rd edition, 2002.

[20] George D. Konidaris and Andrew G. Barto. Skill discovery in continuous reinforcement learning domains using skill chaining. In *Advances in Neural Information Processing Systems*, 2009.

[21] Seungjae Lee, Yibin Wang, Haritheja Etukuru, H. Jin Kim, Nur Muhammad Mahi Shafiullah, and Lerrel Pinto. Behavior generation with latent actions. In *International Conference on Machine Learning*, 2024.

[22] Mingzhe Li, Renhao Zhang, Zhiyang Wen, Siqi Pan, Bruno Castro da Silva, Juan Zhai, and Shiqing Ma. PROMPTMINER: Black-box prompt stealing against text-to-image generative models via reinforcement learning and VLM-guided optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7795–7804, 2026.

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

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

[25] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *Conference on Robot Learning*, 2021.

[26] Warwick Masson, Pravesh Ranchod, and George Konidaris. Reinforcement learning with parameterized actions. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2016.

[27] Quan Nguyen and Koushil Sreenath. Exponential control barrier functions for enforcing high relative-degree safety-critical constraints. In *American Control Conference*, pages 322–328. IEEE, 2016.

[28] Scott Niekum, Sarah Osentoski, George Konidaris, Sachin Chitta, Bhaskara Marthi, and Andrew G. Barto. Learning grounded finite-state representations from unstructured demonstrations. *The International Journal of Robotics Research*, 34(2):131–157, 2015.

[29] Tim Pearce, Tabish Rashid, Anssi Kanervisto, Dave Bignell, Mingfei Sun, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Ida Momennejad, Katja Hofmann, and Sam Devlin. Imitating human behaviour with diffusion models. In *International Conference on Learning Representations*, 2023.

[30] Yuefeng Peng, Mingzhe Li, Kejing Xia, Renhao Zhang, and Amir Houmansadr. Membership inference attacks on vision-language-action models. *arXiv preprint arXiv:2605.07088*, 2026.

[31] Ruixi Qiao, Jie Cheng, Xingyuan Dai, Yonglin Tian, and Yisheng Lv. Offline reinforcement learning with discrete diffusion skills. *arXiv preprint arXiv:2503.20176*, 2025.

[32] Jeffrey F. Queisser and Jochen J. Steil. Bootstrapping of parameterized skills through hybrid optimization in task and policy spaces. *Frontiers in Robotics and AI*, 5:49, 2018.

[33] Nur Muhammad Mahi Shafiullah, Zichen Jeff Cui, Ariuntuya Altanzaya, and Lerrel Pinto. Behavior transformers: Cloning k modes with one stone. In *Advances in Neural Information Processing Systems*, 2022.

[34] 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.

[35] James C. Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37(3):332–341, 1992.

[36] Richard S. Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1–2): 181–211, 1999.

[37] Zhendong Wang, Jonathan J. Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *International Conference on Learning Representations*, 2023.

[38] Kun Wu, Yichen Zhu, Jinming Li, Junjie Wen, Ning Liu, Zhiyuan Xu, and Jian Tang. Discrete policy: Learning disentangled action space for multi-task robotic manipulation. In *IEEE International Conference on Robotics and Automation*, 2025.

[39] Wei Xiao and Calin Belta. Control barrier functions for systems with high relative degree. In *IEEE Conference on Decision and Control*, pages 474–479. IEEE, 2019.

[40] Wei Xiao, Tsun-Hsuan Wang, Chuang Gan, Ramin Hasani, Mathias Lechner, and Daniela Rus. Safediffuser: Safe planning with diffusion probabilistic models. In *International Conference on Learning Representations*, 2025.

[41] Jeongyong Yang, Seunghwan Jang, and SooJean Han. SafeFlowMatcher: Safe and fast planning using flow matching with control barrier functions. In *International Conference on Learning Representations*, 2026. arXiv:2509.24243.

[42] Renhao Zhang, Haotian Fu, Yilin Miao, and George Konidaris. Model-based reinforcement learning for parameterized action spaces. In *International Conference on Machine Learning*, 2024.

[43] Renhao Zhang, Shreyas Chaudhari, and Bruno Castro da Silva. Uncertainty-guided reward labeling for reinforcement learning under limited feedback. In *Advances in Neural Information Processing Systems*, 2026.

[44] Renhao Zhang, Haotian Fu, Mingxi Jia, George Konidaris, Yilun Du, and Bruno Castro da Silva. From noise to control: Parameterized diffusion policies. In *International Conference on Machine Learning*, 2026.

[45] Renhao Zhang, Mingzhe Li, Yuli Li, Jiayu Zheng, and Yilin Miao. Learning what to keep: Evidence-gated parameterized skills for frozen LLM agents. In *NeurIPS 2026 Workshop on Towards Test-Time Continual Learning Agents*, 2026.

A Notation

- $\mathcal{Z} \subset \mathbb{R}^{d_z}$: latent space; $\Omega \subset \mathcal{Z}$: compact region of validity for the latent CBF analysis.
- $\tau(z) = (\tau^0(z), \dots, \tau^H(z)) \in \mathcal{D}^{H+1}$: rollout at latent z with context and noise fixed or averaged out.
- $h_k(z) \triangleq b(\tau^k(z)) - \delta$: per-waypoint shifted barrier; $m_\beta(z) = -\beta^{-1} \log \sum_k e^{-\beta h_k(z)}$: softmin margin.
- $\mathcal{Z}_\delta = \{z \in \mathcal{Z} : m_\beta(z) \geq 0\}$: robust safe latent set.
- $u_z^{\mathrm{ref}} : \mathcal{Z} \to \mathbb{R}^{d_z}$: locally bounded reference update; $u_z^\star$: solution of (7).
- $J_{\tau^k}(z) \in \mathbb{R}^{d_{\mathcal{D}} \times d_z}$: Jacobian of τ^k at z; $H_{\tau^k}(z)$: Hessian tensor.
- $[x]_+ = \max(x, 0)$, applied componentwise on vectors.
- $U_{\mathrm{ref}} = \sup_{z \in \Omega} \left\| u_z^{\mathrm{ref}}(z) \right\|_2$, $\|\nabla m_\beta\|_\infty = \sup_{z \in \Omega} \|\nabla m_\beta(z)\|_2$, $M_m = \sup_{z \in \Omega} |m_\beta(z)|$.
- g_m is the non-degeneracy lower bound on $\|\nabla m_\beta\|_2$ on the unsafe set (hypothesis of Theorem 1); L_b, L_τ are the Lipschitz constants of the barrier and decoder from Section 4.1; $L_m \leq L_b L_\tau$ is the resulting Lipschitz constant of m_β (Appendix B.2); $L_{\nabla m}$ is the Lipschitz constant of ∇m_β used in the optional discrete-time analysis (Appendix C.4).

B Auxiliary Soft-Min and Geometry Facts

This appendix collects three small derivations and one geometry remark that the main text invokes without proof. Sections B.1–B.3 are referenced from Section 4.1; Section B.4 comments on the modeling status of the local geometry condition.

B.1 Soft-min margin and soft-min gap

The soft-min surrogate m_β in main-text eq. (4.1) sandwiches the true worst-waypoint margin: for any $\beta > 0$ and any finite collection $\{h_k\}_{k=0}^H \subset \mathbb{R}$,

$$\min_k h_k - \tfrac{\log(H+1)}{\beta} \;\leq\; m_\beta = -\beta^{-1} \log \sum_{k=0}^{H} e^{-\beta h_k} \;\leq\; \min_k h_k. \tag{9}$$

This is the bound stated in main-text eq. (4.1); the soft-min gap is $\varepsilon_{\mathrm{soft}} = \beta^{-1} \log(H+1)$.

Proof. Let $h_\star = \min_k h_k$. Since $e^{-\beta h_\star} \leq \sum_k e^{-\beta h_k} \leq (H+1) e^{-\beta h_\star}$, applying $-\beta^{-1} \log(\cdot)$ (which reverses inequalities) gives $h_\star - \beta^{-1} \log(H+1) \leq m_\beta \leq h_\star$, which is (9). □

The soft-min gradient is the softmax distribution: $\partial_{h_k} m_\beta = w_k \triangleq e^{-\beta h_k} / \sum_j e^{-\beta h_j} \in [0, 1]$ with $\sum_k w_k = 1$. Hence m_β is 1-Lipschitz in $\mathbf{h}$ under $\|\cdot\|_\infty$, since $|m_\beta(\mathbf{h}) - m_\beta(\mathbf{h}')| \leq \|w\|_1 \|\mathbf{h} - \mathbf{h}'\|_\infty = \|\mathbf{h} - \mathbf{h}'\|_\infty$ by Hölder's inequality.

B.2 Lipschitz latent margin

This subsection derives main-text eq. (4.1): the Lipschitz constant $L_m \leq L_b L_\tau$ for $z \mapsto m_\beta(z)$.

Derivation of eq. (4.1). Write $h_k(z) := b(\tau^k(z)) - \delta$ and $\mathbf{h}(z) := (h_0(z), \dots, h_H(z))$. By main-text eq. (4.1), each h_k is $L_b L_\tau$-Lipschitz on Ω:

$$|h_k(z) - h_k(z')| \leq L_b \|\tau^k(z) - \tau^k(z')\|_2 \leq L_b L_\tau \|z - z'\|_2.$$

Combined with the 1-Lipschitzness of m_β in $\mathbf{h}$ under $\|\cdot\|_\infty$ (Section B.1),

$$|m_\beta(z) - m_\beta(z')| \leq \|\mathbf{h}(z) - \mathbf{h}(z')\|_\infty = \max_k |h_k(z) - h_k(z')| \leq L_b L_\tau \|z - z'\|_2 .$$

The constant $L_m \leq L_b L_\tau$ does not depend on β: the soft-min weights only redistribute Lipschitz mass across waypoints, never amplify it. □

B.3 Waypoint-to-segment buffer

This subsection extends the waypoint-level margin to safety on continuous path segments, as referenced from Section 4.1. Suppose consecutive decoded waypoints are at most ℓ apart and b is L_b-Lipschitz. We show

$$\min_k b(\tau^k(z)) \geq \tfrac{L_b \ell}{2} \quad \Longrightarrow \quad [\tau^k(z), \tau^{k+1}(z)] \subseteq \{x : b(x) \geq 0\} \quad \forall k, \tag{10}$$

a one-step Lipschitz bridge using the elementary geometric fact that any point on a line segment is at most half the segment length from at least one endpoint.

Derivation of (10). Fix k and $x \in [\tau^k(z), \tau^{k+1}(z)]$. The two distances $\|x - \tau^k\|_2$ and $\|x - \tau^{k+1}\|_2$ sum to $\|\tau^{k+1} - \tau^k\|_2 \leq \ell$, so at least one is $\leq \ell/2$. Calling the corresponding waypoint y, L_b-Lipschitzness of b gives

$$b(x) \geq b(y) - L_b \|x - y\|_2 \geq \min_k b(\tau^k(z)) - \tfrac{L_b \ell}{2} \geq 0$$

by hypothesis. Thus $b(x) \geq 0$ for every x on every inter-waypoint segment. □

The factor of $\frac{1}{2}$ comes entirely from the geometric step "the nearer endpoint is within $\ell/2$"; without it the buffer would have to be $L_b \ell$. Choosing $\delta \geq L_b \ell/2$ in the main text is the tightest robustness margin that makes a buffered *waypoint* certificate sufficient for *segment* safety.

B.4 Remarks on the latent safe set and learned latent geometry

Interpretation of the latent safe set. The safe-latent set $\mathcal{Z}_\delta = \{z : m_\beta(z) \geq 0\}$ is the preimage, through the learned decoder, of planned trajectories that satisfy the buffered waypoint condition. It is not a physical forward-invariance certificate for the robot dynamics. Stage I also does not optimize a notion of safety defined by Euclidean latent geometry alone: for every candidate z, LBS decodes $\tau(z)$, evaluates the trajectory-space barriers $b(\tau^k(z))$, and updates z through the pulled-back gradient $\nabla_z m_\beta(z)$, so the update is determined by the safety of the decoded trajectory. Accordingly, the forward invariance of $\mathcal{Z}_\delta$ in Theorem 1 refers to the latent flow $\dot{z} = u_z^\star(z)$, and the final safety claim of LBS is the path-space certificate provided by Stage II. This is consistent with existing safe generative planners [40, 8, 41], which also impose safety on the trajectories generated by a diffusion or flow-matching model. LBS adds a behavior-selection step before path-space correction. Throughout the paper, safety refers to the satisfaction of these constraints by the generated plan. It is distinct from the security and privacy of the generative model itself [30, 22].

Learned latent geometry. The local Lipschitz condition (main-text eq. (4.1)) is an analysis assumption, not a global consequence of the soft-DTW objective. The geometry loss [6] encourages nearby latents to decode to nearby behavior traces, but global bi-Lipschitz guarantees would require stronger conditions such as full training support, smooth non-collapsed decoder, and exclusion of off-manifold extrapolation. We therefore restrict Stage I to a clipped region Ω around the prior center and rely on the local condition only inside Ω.

Relation to parameterized diffusion policies. The geometry objective is inherited from parameterized diffusion policies (PDP) [44], which show that, trained on multimodal demonstrations, it yields a low-dimensional behavior manifold that supports smooth interpolation between demonstrated strategies and the synthesis of intermediate behaviors. PDP also ablates this term: removing it produces a geometrically warped latent space and substantially degrades gradient-based adaptation, while a discrete behavior representation cannot synthesize intermediate strategies. Meaningful latent interpolation therefore does not come from flow matching alone, but from the combination of multimodal demonstrations, the geometry-alignment objective, and a continuous behavior-conditioned generator. Constraint II in our experiments tests this property directly: every demonstrated mode is blocked, so success requires decoding an unseen intermediate behavior rather than selecting a demonstrated one. LBS reaches the goal in 90%, 100%, and 90% of episodes in 2D point-navigation, Franka, and Maze2D, respectively, at 100% Safety (Table 1). We do not claim robustness to arbitrarily misaligned latent spaces. LBS relies on this learned local structure and uses it only within Ω.

C Proofs for Stage-I Latent Steering

This appendix proves the Stage-I results stated in Section 4.2: the closed-form projection (Section C.1), Proposition 1 (Section C.2), and Theorem 1 (Section C.3). A discrete-time and inexact-gradient version is given as an optional extension (Section C.4).

C.1 Closed-form projection for the latent CBF-QP

This subsection derives the closed-form solution (8). With $\psi(z) := \nabla m_\beta(z)^\top u_z^{\mathrm{ref}}(z) + \kappa \operatorname{sgn}(m_\beta(z))|m_\beta(z)|^\rho$ as defined in the main text, (7) has a single linear inequality constraint, so its solution is the Euclidean projection of u_z^{ref} onto a closed half-space.

Derivation of (8). Form the Lagrangian

$$\mathcal{L}(u, \mu) = \|u - u_z^{\mathrm{ref}}\|_2^2 - \mu\big(\nabla m_\beta^\top u + \kappa \operatorname{sgn}(m_\beta)|m_\beta|^\rho\big), \quad \mu \geq 0.$$

Stationarity gives $u - u_z^{\mathrm{ref}} = \frac{\mu}{2}\nabla m_\beta$, i.e., $u = u_z^{\mathrm{ref}} + \lambda \nabla m_\beta$ with $\lambda := \mu/2 \geq 0$. If $\psi(z) \geq 0$, the reference is feasible and complementary slackness gives $\lambda = 0$. If $\psi(z) < 0$, the constraint binds at the optimum, so $\psi(z) + \lambda \|\nabla m_\beta\|_2^2 = 0$, giving $\lambda = -\psi(z)/\|\nabla m_\beta\|_2^2 > 0$. Combining the two cases yields $\lambda(z) = [-\psi(z)]_+ / \|\nabla m_\beta(z)\|_2^2$ and (8). □

The projection has no free parameter: once u_z^{ref} and the CBF rate (κ, ρ) are fixed, $u_z^\star$ is uniquely determined wherever $\nabla m_\beta(z) \neq 0$.

C.2 Proof of Proposition 1: latent margin decrease

Let $V(z) := [-m_\beta(z)]_+$. We show that whenever the CBF constraint in (7) is satisfied at z with $m_\beta(z) < 0$, the latent flow $\dot{z} = u_z^\star(z)$ produces $\dot{V}(z) \leq -\kappa V(z)^\rho$.

Proof of Proposition 1. On $\{m_\beta(z) < 0\}$, $V(z) = -m_\beta(z) > 0$ and $\operatorname{sgn}(m_\beta) = -1$. The CBF constraint in (7) becomes $\nabla m_\beta(z)^\top u_z^\star(z) \geq \kappa V(z)^\rho$. Since $V(z) = -m_\beta(z)$ on this set, $\dot{V}(z(t)) = -\dot{m}_\beta(z(t)) = -\nabla m_\beta(z(t))^\top u_z^\star(z(t))$. Combining,

$$\dot{V}(z(t)) \leq -\kappa\, V(z(t))^\rho.$$

□

C.3 Proof of Theorem 1: finite-time recovery

Theorem 1 adds a non-degeneracy hypothesis $\|\nabla m_\beta(z)\|_2 \geq g_m > 0$ on the unsafe set, under which the descent inequality from Proposition 1 integrates to a finite-time bound. Before the proof we record well-posedness of the Stage-I flow.

Lemma 1 (Well-posedness of the latent flow)**.** *Under the local Lipschitz condition of Section 4.1, the non-degeneracy hypothesis, and locally Lipschitz u_z^{ref}, the closed-form solution $u_z^\star$ in* (8) *is locally bounded and locally Lipschitz on $\Omega \setminus \mathcal{Z}_\delta$. Hence $\dot{z} = u_z^\star(z)$ admits a unique Carathéodory solution from any $z(0) \in \Omega$ until it enters $\mathcal{Z}_\delta$ or exits Ω.*

Proof. On $\Omega \setminus \mathcal{Z}_\delta$, the denominator $\|\nabla m_\beta\|_2^2$ is bounded below by $g_m^2 > 0$. Since ∇m_β, m_β, and u_z^{ref} are locally Lipschitz on Ω, and $\psi \mapsto [-\psi]_+$ is 1-Lipschitz, (8) is a rational function of locally Lipschitz quantities with bounded-below denominator, so $u_z^\star$ is locally Lipschitz. Picard–Lindelöf applies on $\Omega \setminus \mathcal{Z}_\delta$. □

Proof of Theorem 1. On $\Omega \setminus \mathcal{Z}_\delta$ and under non-degeneracy, Proposition 1 gives the strict descent

$$\dot{V}(z(t)) \leq -\kappa\, V(z(t))^\rho, \quad V(z(t)) > 0. \tag{11}$$

The scalar comparison ODE $\dot{y} = -\kappa y^\rho$ with $y(0) = V(z_0)$ separates as $y^{-\rho}\, dy = -\kappa\, dt$ and integrates to $y(t)^{1-\rho} = y(0)^{1-\rho} - \kappa(1-\rho)t$, vanishing at $t^\star = y(0)^{1-\rho}/(\kappa(1-\rho))$. By the comparison lemma [19, Lemma 3.4], $V(z(t)) \leq y(t)$ on $[0, t^\star]$, so $V(z(t)) = 0$ for $t \geq T_z = |m_\beta(z_0)|^{1-\rho}/(\kappa(1-\rho))$. Forward invariance of $\mathcal{Z}_\delta$ follows because at any boundary point $V = 0$ and $\dot{V} \leq 0$ by (11), interpreted as zero at $V = 0$. □

C.4 Optional: discrete-time and inexact-gradient steering

This subsection states a robustness guarantee for the discrete-time projected Euler iterate of Section 4.2 when the latent gradient is replaced by a finite-difference estimate. It is not used by the main-text claims but supports implementation analysis. We first record an auxiliary Lipschitz bound on ∇m_β used by Taylor's theorem along the Euler step.

Lemma 2 (Lipschitz gradient of m_β)**.** *If, in addition to the conditions of Section 4.1, τ is C^2 on Ω with $\|J_\tau\|_{\mathrm{op}} \le L_\tau$, $\|H_\tau\|_{\mathrm{op}} \le M_\tau$, and ∇b is $L_{\nabla b}$-Lipschitz, then ∇m_β is Lipschitz on Ω with*

$$L_{\nabla m} \le M_\tau L_b + L_\tau^2 L_{\nabla b} + \tfrac{\beta(H+1)}{2}(L_b L_\tau)^2. \tag{12}$$

Proof. Writing $\nabla m_\beta(z) = \sum_k w_k(z)\nabla h_k(z)$ with $\nabla h_k = J_{\tau^k}^\top \nabla b \circ \tau^k$, we bound the two factors separately. Each ∇h_k is $(M_\tau L_b + L_\tau^2 L_{\nabla b})$-Lipschitz by adding/subtracting $J_{\tau^k}(z')^\top \nabla b(\tau^k(z))$. The softmax $w(\mathbf{h})$ has Jacobian $\partial_{\mathbf{h}} w = \beta(\mathrm{diag}(w) - ww^\top)$ with operator norm $\le \beta/2$ (a covariance with eigenvalues bounded by $1/2$, since $\mathrm{Var}_w(v) \le 1/2$ for unit v, with equality at two-point uniform w). Combined with $\|\partial_z \mathbf{h}\|_{\mathrm{op}} \le \sqrt{H+1}\, L_b L_\tau$ and Cauchy–Schwarz, $\|w(z) - w(z')\|_1 \le \frac{\beta(H+1)}{2} L_b L_\tau \, \|z - z'\|_2$. Combining (with $\|\nabla h_k\|_2 \le L_b L_\tau$),

$$\|\nabla m_\beta(z) - \nabla m_\beta(z')\|_2 \le L_{\nabla h} \, \|z - z'\|_2 + \|w(z) - w(z')\|_1 \, L_b L_\tau,$$

giving (12). □

Theorem 3 (Inexact-gradient discrete-time recovery)**.** *Let $\widehat{\nabla} m_\beta(z) = \nabla m_\beta(z) + e(z)$ with $\sup_\Omega \|e(z)\|_2 \le \varepsilon_g$, and let $\widehat{u}_z(z)$ be the closed-form QP solution* (8) *using $\widehat{\nabla} m_\beta$ in place of ∇m_β. Suppose Lemmas 1–2 hold and the Euler iterate $z_{i+1} = z_i + \eta \widehat{u}_z(z_i)$ remains in Ω. Then*

$$V_{i+1} \le V_i - \eta\kappa V_i^\rho + c_1 \eta \varepsilon_g + c_2 \eta^2, \qquad V_i := [-m_\beta(z_i)]_+, \tag{13}$$

with structural constants c_1, c_2 depending on $L_{\nabla m}$, $\sup_\Omega \|u_z^\star\|_2$, and the bounds in Section 4.1. For any $\varepsilon > 0$ with $\kappa\varepsilon^\rho > c_1\varepsilon_g + c_2\eta$, the iterates enter $\{z : m_\beta(z) \ge -\varepsilon\}$ in $N^\star \le V_0^{1-\rho}/[\eta(1-\rho)(\kappa - \varepsilon^{-\rho}(c_1\varepsilon_g + c_2\eta))]$ iterations, with equilibrium gap $\varepsilon = O((\varepsilon_g + \eta)^{1/\rho})$.

Proof. Step 1 bounds the perturbed barrier rate. Let $\widehat{\psi} = \widehat{\nabla} m_\beta^\top u_z^{\mathrm{ref}} + \kappa\, \mathrm{sgn}(m_\beta)|m_\beta|^\rho = \psi + e^\top u_z^{\mathrm{ref}}$, so $|\widehat{\psi} - \psi| \le \sup_\Omega \|u_z^{\mathrm{ref}}\|_2 \, \varepsilon_g$. A direct expansion of $\nabla m_\beta(z)^\top \widehat{u}_z(z)$ using $\nabla m_\beta = \widehat{\nabla} m_\beta - e$ and the closed-form $\widehat{u}_z = u_z^{\mathrm{ref}} + \widehat{\lambda}\widehat{\nabla} m_\beta$ yields, on $m_\beta < 0$, $\nabla m_\beta(z)^\top \widehat{u}_z(z) \ge \kappa V(z)^\rho - c_1\varepsilon_g$, where c_1 absorbs $|e^\top u_z^{\mathrm{ref}}|$ and the 1-Lipschitz dependence of $[-\widehat{\psi}]_+$ on $\widehat{\psi} - \psi$. Step 2 applies Taylor's theorem to m_β along the Euler step using Lemma 2: $m_\beta(z_{i+1}) \ge m_\beta(z_i) + \eta \nabla m_\beta(z_i)^\top \widehat{u}_z(z_i) - \frac{1}{2} L_{\nabla m} \eta^2 \|\widehat{u}_z(z_i)\|_2^2$, and substituting Step 1 plus $\|\widehat{u}_z\|_2 \le \sup_\Omega \|u_z^\star\|_2$ gives (13). Step 3: whenever $V_i \ge \varepsilon$, $V_i^{-\rho} \le \varepsilon^{-\rho}$, so (13) bounds $V_{i+1} \le V_i - \eta\kappa' V_i^\rho$ with $\kappa' = \kappa - \varepsilon^{-\rho}(c_1\varepsilon_g + c_2\eta) > 0$. Discrete fractional-power decay yields the $N^\star$ bound; balancing $\kappa\varepsilon^\rho = c_1\varepsilon_g + c_2\eta$ gives the equilibrium gap. □

D Proofs for Stage-II Correction and Handoff

This appendix proves the Stage-II results stated in Section 4.3: Proposition 2 (Section D.1), Corollary 1 (Section D.2), and Theorem 2 (Section D.3).

D.1 Proof of Proposition 2: latent-to-path margin transfer

Proof of Proposition 2. Decompose $\tilde{b}_{t_w}(z) - m_\beta(z)$ into a soft-min gap and a predictor gap.

Soft-min gap. By the soft-min sandwich (Section B.1) applied to $h_k(z) = b(\tau^k(z)) - \delta$, with $\hat{m}(z) := \min_k h_k(z)$, $|\hat{m}(z) - m_\beta(z)| \le \varepsilon_{\mathrm{soft}} = \beta^{-1}\log(H+1)$.

Predictor gap. By Lipschitzness of b and the definition of $\varepsilon_{\mathrm{pred}}(z)$,

$$|b(\tau_{t_w}^{c,k}(z)) - b(\tau^k(z))| \le L_b \, \|\tau_{t_w}^{c,k}(z) - \tau^k(z)\|_2 \le L_b \varepsilon_{\mathrm{pred}}(z).$$

Subtracting δ and taking minima over k (using $|\min_k f_k - \min_k g_k| \le \max_k |f_k - g_k|$) gives $|\tilde{b}_{t_w}(z) - \hat{m}(z)| \le L_b \varepsilon_{\mathrm{pred}}(z)$.

Combine. The triangle inequality yields the bound stated in Proposition 2. □

Remark 1 (Predictor-sensitivity bound)**.** *For a* SAFEFLOWMATCHER*-style corrector whose predictor is a Lipschitz ODE with field L_v-Lipschitz in state, standard ODE sensitivity gives $\varepsilon_{\mathrm{pred}}(z) \le L_\tau e^{L_v t_w} \bar{\xi} + \int_0^{t_w} w_s r_s \, ds$, where $\bar{\xi}$ controls initial perturbation and the integral captures the soft-slack term in* (5)*. For a post-hoc path-space CBF-QP applied directly to $\tau(z)$, $\varepsilon_{\mathrm{pred}}(z) \equiv 0$ and Proposition 2 reduces to the soft-min-only bound.*

D.2 Proof of Corollary 1: correction time and energy

Proof of Corollary 1. The Stage-II CBF-QP with fractional-power rate ϵ and exponent ρ enforces $\dot{V}_{\mathrm{path}} \le -\epsilon V_{\mathrm{path}}^{\rho}$ on $V_{\mathrm{path}} > 0$, parallel to (11). Integrating yields $V_{\mathrm{path}}(t)^{1-\rho} = V_+^{1-\rho} - \epsilon(1-\rho)(t - t_w)$ for $t \in [t_w, T_{\mathrm{corr}}]$, so

$$T_{\mathrm{corr}} \le t_w + \frac{V_+^{1-\rho}}{\epsilon(1-\rho)}.$$

For the energy bound, $J_{II}(z_+) \le \int_{t_w}^{T_{\mathrm{corr}}} c_{II} \, V_{\mathrm{path}}(t)^{2\rho} \, dt$. Substituting $V_{\mathrm{path}}(t)^{2\rho} = (V_+^{1-\rho} - \epsilon(1-\rho)(t-t_w))^{2\rho/(1-\rho)}$ and the change of variables $u = V_+^{1-\rho} - \epsilon(1-\rho)(t-t_w)$ gives

$$J_{II}(z_+) \le \frac{c_{II}}{\epsilon(1-\rho)} \int_0^{V_+^{1-\rho}} u^{2\rho/(1-\rho)} \, du = \frac{c_{II} \, V_+^{1+\rho}}{\epsilon(1+\rho)}.$$

□

D.3 Proof of Theorem 2: cost-sensitive handoff

Proof of Theorem 2. **Stage-I descent in continuous time.** Under the non-degeneracy hypothesis along the Stage-I trajectory, the comparison lemma applied to Proposition 1 gives $V(t)^{1-\rho} = V_0^{1-\rho} - \kappa(1-\rho)t$ for $t \in [0, V_0^{1-\rho}/(\kappa(1-\rho))]$ (same scalar argument as in Section C.3). Identifying $t = N_z \eta$ in the exact-gradient limit gives the discrete decay law

$$V_+^{1-\rho} = V_0^{1-\rho} - \eta\kappa(1-\rho) \, N_z, \tag{14}$$

so the number of Stage-I iterations needed to reach a given terminal deficit V_+ is $N_z(V_+) = (V_0^{1-\rho} - V_+^{1-\rho})/(\eta\kappa(1-\rho))$.

Joint-cost expression. Combining (14) with the Stage-I per-iteration cost $J_I(N_z) = N_z c_I$ (Appendix E.3) and the Stage-II energy bound from Corollary 1, the joint cost as a function of $V_+ \in (0, V_0]$ is

$$J_{\mathrm{tot}}(V_+) \le \frac{c_I \big(V_0^{1-\rho} - V_+^{1-\rho}\big)}{\eta\kappa(1-\rho)} + \frac{c_{II} \, V_+^{1+\rho}}{\epsilon(1+\rho)}, \tag{15}$$

the upper bound used in Theorem 2.

Convexity. Differentiating with respect to $V_+ \in (0, V_0]$,

$$\frac{dJ_{\mathrm{tot}}}{dV_+} = -\frac{c_I}{\eta\kappa} \, V_+^{-\rho} + \frac{c_{II}}{\epsilon} \, V_+^{\rho}. \tag{16}$$

The second derivative $c_I \rho V_+^{-\rho-1}/(\eta\kappa) + c_{II}\rho V_+^{\rho-1}/\epsilon$ is strictly positive for $V_+ > 0$, so J_{tot} is strictly convex on $(0, V_0]$.

Optimum. Setting (16) to zero,

$$\frac{c_{II}}{\epsilon} V_+^{2\rho} = \frac{c_I}{\eta\kappa}, \quad \Longrightarrow \quad V_+^\star = \left(\frac{c_I \, \epsilon}{\eta\kappa \, c_{II}}\right)^{1/(2\rho)},$$

the unique stationary point and, by strict convexity, the unique global minimum. The handoff iteration $N_z^\star$ follows from the descent relation when $V_+^\star \le V_0$. If $V_+^\star > V_0$, $dJ_{\mathrm{tot}}/dV_+ < 0$ on $(0, V_0]$, so J_{tot} is strictly decreasing on $(0, V_0]$ and the constrained minimum is at $V_+ = V_0$, i.e., $N_z^\star = 0$ and Stage I is skipped. □

Algorithm 1 Latent Barrier Steering (LBS)

Require: PFM generator G_θ, context c, barrier b, robustness margin δ; hyperparameters $(\kappa, \rho, \eta, \gamma_z, \beta, N_z)$; latent clipping box Ω
Ensure: Trajectory τ certified by Stage II (with optional residual-violation flag)

Stage I: latent barrier steering

$z \leftarrow \texttt{encoder_warm_start}(c)$
for $i = 1, \dots, N_z$ **do**
  $\tau \leftarrow G_\theta(c, z)$
  $m_\beta \leftarrow -\beta^{-1} \log \sum_{k=0}^{H} \exp\big(-\beta(b(\tau^k) - \delta)\big)$
  $\hat{\nabla} m_\beta \leftarrow \texttt{zeroth_order_grad}(m_\beta, z)$ ▷ $2d_z$ central finite differences
  $u_z^{\mathrm{ref}} \leftarrow -\gamma_z\, z$
  $\psi \leftarrow \hat{\nabla} m_\beta^\top u_z^{\mathrm{ref}} + \kappa\, \mathrm{sgn}(m_\beta)\, |m_\beta|^\rho$
  if $\psi < 0$ **and** $\|\hat{\nabla} m_\beta\|_2^2 > 0$ **then**
    $\lambda \leftarrow -\psi / \|\hat{\nabla} m_\beta\|_2^2$
    $u_z \leftarrow u_z^{\mathrm{ref}} + \lambda\, \hat{\nabla} m_\beta$ ▷ closed-form QP (8)
  else
    $u_z \leftarrow u_z^{\mathrm{ref}}$
  end if
  $z \leftarrow \mathrm{clip}_\Omega(z + \eta\, u_z)$
end for
$\tau \leftarrow G_\theta(c, z)$
$m_\beta \leftarrow -\beta^{-1} \log \sum_{k=0}^{H} \exp\big(-\beta(b(\tau^k) - \delta)\big)$
$\texttt{residual_violation} \leftarrow (m_\beta < 0)$ ▷ flag only; Stage II runs in either case

Stage II: certified path-space correction

$\tau \leftarrow \texttt{PathCorrector}(\tau, b, \delta)$ ▷ post-hoc path CBF-QP or SAFEFLOWMATCHER
return τ, $\texttt{residual_violation}$

Remark 2 (Effect of margin transfer slack on $V_+^\star$)**.** *The proof above absorbs the soft-min and predictor-sensitivity slack from Proposition 2 into the constant c_{II}. Retaining the slack explicitly replaces V_+ by $\widetilde{V}_+ := V_+ + \varepsilon_{\mathrm{soft}} + L_b \bar{\varepsilon}_{\mathrm{pred}}$ in the Stage-II energy bound (Corollary 1), and the optimum $V_+^\star$ shifts upward by an amount that vanishes as $\beta \to \infty$ and $\varepsilon_{\mathrm{pred}} \to 0$ (post-hoc corrector). The qualitative form of the stopping rule is unchanged.*

Remark 3 (Discrete-time and inexact-gradient corrections)**.** *Theorem 2 uses the exact-gradient continuous-time descent law from Proposition 1. Replacing it with the discrete inexact-gradient analysis of Theorem 3 adds correction terms $c_1 \eta \varepsilon_g + c_2 \eta^2$ per step, which propagate through* (14) *as a downward shift of the achievable V_+ floor. The optimum $V_+^\star$ in Theorem 2 remains valid above this floor; below it, the achievable V_+ is bounded by the equilibrium gap $\varepsilon = O((\varepsilon_g + \eta)^{1/\rho})$ from Theorem 3.*

E Algorithm and Implementation

This appendix gives the full LBS algorithm with implementation details: the pseudocode, the projected Euler update with finite-difference gradient estimation, the per-step and total Stage-I cost used by Theorem 2, and the hyperparameters used across all experiments.

E.1 Pseudocode

Stage I does not need to reach the safe-latent set $\mathcal{Z}_\delta$ before handoff: increasing m_β already reduces the deficit Stage II must correct (Section 4.3). The flag `residual_violation` records whether the Stage-I terminal margin was negative; the certified safe trajectory is always produced by Stage II.

E.2 Projected Euler and gradient estimation

Stage I is implemented with projected explicit Euler steps,

$$z_{i+1} \;=\; \Pi_{\Omega}\big(z_i + \eta\, \widehat{u}_z(z_i)\big),$$

where $\eta > 0$ is the latent step size and Π_{Ω} projects onto the latent clipping box Ω. The update direction $\widehat{u}_z(z_i)$ is the closed-form latent CBF projection (8) evaluated with an estimate $\widehat{\nabla} m_\beta(z_i)$ of the latent gradient. When backpropagation through the decoder is available, the gradient can be computed directly. Otherwise, we use central finite differences,

$$[\widehat{\nabla} m_\beta(z)]_i \;=\; \frac{m_\beta(z + h e_i) - m_\beta(z - h e_i)}{2h}, \qquad i = 1, \dots, d_z,$$

which costs $2d_z$ generator evaluations per latent step. Central differences have truncation error $\varepsilon_g = O(h^2)$ for C^3 scalar fields. For larger d_z, SPSA-style random directional estimators [35] or a learned critic for m_β are natural replacements.

E.3 Cost accounting

Each Stage-I iteration consists of one rollout decode, one margin evaluation, $2d_z$ additional generator calls for the central-difference gradient, one closed-form projection, and one Euler update. We summarize the dominant cost as a per-step constant $c_I > 0$ (in our setting, c_I is dominated by the $2d_z$ generator calls, so each Stage-I iteration uses $2d_z + 1$ generator evaluations). Over N_z Stage-I iterations, the total Stage-I cost is

$$J_I(N_z) \;=\; N_z\, c_I, \tag{17}$$

which is balanced against the Stage-II correction energy in Theorem 2. The handoff iteration $N_z^\star$ in Theorem 2 selects the smallest N_z for which the marginal Stage-I cost matches the marginal reduction in Stage-II correction energy. Stage II adds one decoded rollout and the cost of the chosen path-space corrector, so the total LBS cost is

$$\mathrm{Cost}_{\mathrm{LBS}} \;=\; N_z(2d_z + 1)\, C_{\mathrm{gen}} \;+\; C_{\mathrm{gen}} \;+\; C_{\mathrm{stageII}}.$$

Per-step sampler-space methods such as SAFEDIFFUSER and SAFEFLOW instead solve CBF-QPs throughout the denoising or flow integration horizon. SAFEFLOWMATCHER solves QPs only during its correction phase, but without the Stage I semantic warm start.

E.4 Hyperparameters

All experiments use the same defaults, not per-domain tuned:

$$(\kappa,\ \rho,\ \eta,\ N_z,\ \gamma_z,\ \beta,\ \delta,\ h) \;=\; (1.0,\ 0.5,\ 0.1,\ 20,\ 0.01,\ 10,\ 0.05,\ 0.01),$$

with latent clipping box $\Omega = [-6, 6]^{d_z}$.

F Experimental Details

This appendix documents the three benchmark domains, their training data and per-domain backbone size, the three constraint setups used in each, and the evaluation protocol. Backbone training and inference hyperparameters are summarised in Appendix G and H, respectively, and we do not duplicate them here.

The three domains share a common evaluation template: each domain has a fixed start–goal pair, a multi-modal expert dataset, and three deliberately different constraint setups labelled *mode-selection* (S1), *mode-interpolation* (S2), and *CBF-correction* (S3). S1 admits at least one demonstration mode without modification; S2 blocks every demonstration mode but leaves a feasible path that requires latent interpolation between modes; S3 leaves every demonstration mode partially clipped, so a path-space CBF correction is required on top of the latent solution.

F.1 2D point-navigation

Dynamics. Single-integrator on $\mathbb{R}^2$, $s_{t+1} = s_t + a_t$, with start $s_0 = (0, 0)$ and goal $g = (10, 0)$, horizon $H = 50$. State is (x, y) and actions are (a_x, a_y) with no per-step bound (training data dictates magnitude).

Training modes. Four sine-wave modes with peak amplitude $y_{\text{peak}} \in \{-3, -1, 1, 3\}$, 30 demonstrations per mode, additive state noise $\mathcal{N}(0, 0.15^2)$ at each step with endpoints clamped to start and goal. Each demonstration interpolates start and goal along x and rides the sine in y.

Backbone. The 2D backbone is intentionally small (see Appendix G): hidden width 256, four residual blocks, behavior latent $d_z = 2$. The encoder is a transformer trajectory VAE with sinusoidal positional encoding.

Constraint setups.

S1 (mode selection). A single rectangular hazard $[3, 7] \times [-5, 2]$. Three modes ($y_{\text{peak}} \in \{-3, -1, 1\}$) intersect the rectangle; the upper mode y_{peak}=3 clears it. *Latent recovery is sufficient*: Stage I selects the upper mode and Stage II makes near-zero corrections.

S2 (mode interpolation). Two rectangles $[3, 7] \times [0.5, 6]$ and $[3, 7] \times [-6, -0.5]$ form a horizontal corridor of width 1 at $y \in [-0.5, 0.5]$ for $x \in [3, 7]$. All four training modes are blocked, but the interior $y_{\text{peak}}\approx 0$ region is open. *Latent interpolation is required*: Stage I drives z to a peak amplitude not seen in training; Stage II remains nearly inactive.

S3 (CBF correction). A single thicker hazard $[3, 7] \times [-3.5, 3.5]$. All four modes clip the band; the outer pair $y_{\text{peak}} = \pm 3$ have moderate overlap. *Stage I alone is insufficient*: the latent solution rides the band edge but still violates at corners; Stage II's path-space CBF closes the residual gap.

F.2 Franka 3D end-effector

Domain. 7-DoF Franka Panda arm in Cartesian end-effector planning mode. The trajectory variable is the EE position $(x, y, z) \in \mathbb{R}^3$; we plan in EE space and project to joint space via the standard differential-IK controller used in the Franka stack. Start $s_0 = (0.3, 0.0, 0.5)\,\text{m}$, goal $g = (0.7, 0.0, 0.3)\,\text{m}$, horizon $H = 50$. Differentiable EE-position barriers are used by every method, matching SAFEFLOW [8].

Training modes. Eight expert behaviors covering combinations of approach direction (up/down/left/right) and approach distance (near/far), 50 demonstrations per behavior. The behaviors are chosen so that each constraint setup eliminates a well-defined subset of them.

Backbone. Hidden width 256, four residual blocks, $d_z = 4$ (see Appendix G); z-projection width 32; no auxiliary start–goal context (start/goal are fixed).

Constraint setups. All three setups place a thin wall at $x \in [0.45, 0.55]$ between start and goal:

S1 (mode selection). Lower wall blocks $z < 0.58$; upper portion is solid except for a slot $y \in [-0.08, 0.08]$, $z \in [0.58, 0.85]$; a small sphere of radius 0.03 at $(0.5, 0, 0.66)$ sits inside the slot. Among the eight training modes only the up–far behavior threads the slot. Stage I selects up–far; Stage II edges around the sphere.

S2 (mode interpolation). Same wall structure, but the open region is a diagonal hole $y \in [0.04, 0.20]$, $z \in [0.48, 0.58]$ that lies *between* the up–near and right–near training behaviors. No single training mode threads the hole; latent interpolation produces an approach via $\sim (0.5, 0.12, 0.53)$.

S3 (CBF correction). Same diagonal hole as S2, plus a small sphere of radius 0.03 at $(0.5, 0.12, 0.53)$ inside the hole. Stage I produces the interpolated approach; Stage II's path-space CBF detours the EE around the sphere within the $\sim 0.16 \times 0.10\,\text{m}$ window.

F.3 Maze2D

Domain. D4RL `maze2d-large` layout [12]. The state is $(x, y, \dot{x}, \dot{y})$ and the action is the 2D velocity command; the planner outputs an action sequence and rolls it forward through the standard maze2d single-integrator dynamics. Horizon $H = 384$, matching SAFEDIFFUSER [40]. Start $s_0 = (1, 3)$, goal $g = (7, 9)$; eight start–goal context dimensions are fed to the backbone via FiLM (see Appendix G).

Training data. A custom 300-trajectory dataset on six distinct routes (50 demonstrations per route) collected by a dense-waypoint expert controller. The six routes are referenced in the setup descriptions below by their direction codes (e.g. "LD" = left then down). D4RL's native `maze2d-large` replay buffer has narrow corridors with limited mode diversity, so a custom dataset is needed to make the mode-selection vs. interpolation distinction sharp.

Backbone. Hidden width 768, 20 residual blocks, z-projection width 128, $d_z = 4$, encoder positional-encoding max length 1024. The deeper/wider model is needed for the longer horizon; see Appendix G.

Constraint setups. Each setup uses a small set of circular hazards plus a few axis-aligned rectangular wall extensions chosen to close five of the six demonstration routes:

S1 (mode selection, "LD" free). Six small circles (r=0.4) and six wall rectangles fence off the entire right half of the maze, leaving only the LD route (down the left corridor and across the bottom row). Baselines without latent recovery default to the right-arc path and trap against the right-side fence.

S2 (mode interpolation, "RUR" partly free). Hazards block the L entrance, the D branch, and the UD branch. RUR is partly open but pinched by a small obstacle; latent interpolation is needed to bend RUR into the residual corridor.

S3 (CBF correction, "RD" partly free). Hazards block the L entrance and the U branch. RD is partly open but a small additional obstacle clips the natural rollout. Stage I selects the residual mode; Stage II's path-space CBF deflects the rollout around the small obstacle.

Wall handling. Following the literature convention used by SAFEDIFFUSER, SAFEFLOW, and SAFEFLOWMATCHER, the CBF barrier and reported safety metric are defined over user-specified hazard obstacles only; the maze walls themselves are handled implicitly by the generator's learned prior and, for executable rollout, by Liang–Barsky truncation against the wall set (so a rollout that would tunnel through a wall is clipped at the wall and the post-truncation goal-error is what we report).

F.4 Evaluation protocol

Episodes and seeds. Each method is evaluated on 50 episodes per setup with a single fixed seed (default 42). Episodes within a setup vary only in the diffusion / flow-matching latent noise; start, goal, and obstacle layout are fixed by definition of the setup.

Safety metric. Segment-based safety using Liang–Barsky segment / axis-aligned-box intersection for rectangle hazards and a closed-form segment / sphere test for sphere hazards. A trajectory is *safe* iff none of its H piecewise-linear segments intersects any hazard. We report mean Safety as the fraction of safe episodes.

Reach metric. Goal-reach is determined by the post-truncation final-state distance to the goal: an episode counts as "reached" iff the truncated rollout ends within a domain-specific tolerance of the goal. The tolerance and other thresholds are fixed across methods per domain.

Other metrics. Beyond Safety and Reach, we report final goal distance (ED), CBF intervention energy, curvature smoothness (CS), and per-episode Stage-I/Stage-II wall-clock time. The full three-constraint results are reported in Appendix I.

Hardware. Wall-clock numbers are measured on a single CPU thread for the 2D domain and a single A100 GPU for Franka and Maze2D. The 2D backbone is small enough to time on CPU; Franka and Maze2D backbones use GPU at test time.

Baselines. The three CBF baselines (SAFEDIFFUSER, SAFEFLOW, SAFEFLOWMATCHER) are run from their reference open-source implementations with the same hazard set, start, goal, and horizon as our methods. We retrain each baseline's backbone on our domain datasets so that the comparison

isolates the safety wrapper rather than the prior. Their training and test-time hyperparameters follow the upstream defaults; we list only *our* method's settings in Appendices G and H.

G Backbone Training Hyperparameters

This appendix records the exact hyperparameters used to train the parameterised flow-matching backbone used by Latent Barrier Steering. We list only *our* method's settings; baselines (SAFEDIFFUSER, SAFEFLOW, SAFEFLOWMATCHER) follow their respective open-source reference implementations and are not duplicated here.

The training pipeline has two stages per domain. We first train a Trajectory VAE encoder with a soft-DTW reconstruction term to produce the behavior latent z; we then freeze the encoder and train the conditional flow-matching generator on top.

G.1 Trajectory VAE encoder

The encoder is a transformer-style trajectory VAE with sinusoidal positional encoding and FiLM-style decoding. Optimiser settings are shared across all three domains: AdamW with learning rate 10^{-4}, weight decay 10^{-5}, cosine-annealed to a minimum of 10^{-6} over the full schedule, batch size 32. The reconstruction term is soft-DTW [6] with smoothing $\gamma_{\mathrm{DTW}} = 10^{-2}$. The latent dimension d_z matches the value used by the LBS hyperparameter table in Appendix E.4.

Table 4: **Encoder hyperparameters.**

	2D point-nav	Franka 3D EE	Maze2D
Hidden width	128	128	128
Latent dim d_z	2	4	4
Optimizer	AdamW	AdamW	AdamW
Learning rate	10^{-4}	10^{-4}	10^{-4}
Weight decay	10^{-5}	10^{-5}	10^{-5}
LR schedule	cosine	cosine	cosine
Batch size	32	32	32
Soft-DTW γ_{DTW}	10^{-2}	10^{-2}	10^{-2}

G.2 Conditional flow-matching generator

The generator is a z-conditioned flow-matching velocity predictor with the linear interpolant $x_t = (1-t)x_0 + t\varepsilon$ and an explicit-Euler ODE sampler. It uses FiLM injection of the encoder z and (for Maze2D only) FiLM injection of an 8-dimensional start–goal context c. We optimise with Adam, learning rate 10^{-4}, batch size 32.

Table 5: **Generator hyperparameters** for the flow-matching backbone. Maze2D uses a wider/deeper backbone because of its longer horizon and start–goal conditioning. Sampling uses the listed number of explicit-Euler integration steps.

	2D point-nav	Franka 3D EE	Maze2D
Hidden width	256	256	768
Residual blocks	4	4	20
z-projection width	32	32	128
Start/goal conditioning dim	—	—	8
FM Euler steps	50	50	50
Optimizer	Adam	Adam	Adam
Learning rate	10^{-4}	10^{-4}	10^{-4}
Batch size	32	32	32

G.3 Notes on choices

The encoder and generator hyperparameters are not tuned per constraint regime; we use one set per domain and reuse the same checkpoints across all three constraint setups in the main results. The 2D and Franka backbones are deliberately small ($\sim$0.4–0.6 M parameters) so that the central-difference cost of Stage I (Appendix E.4) stays within a few seconds per call; Maze2D is wider/deeper to handle the longer horizon. Released checkpoints are the best-validation snapshots used in all main-text tables. Both backbones are trained on demonstrations alone, without reward labels [4, 43], and a new test-time constraint requires no retraining.

H Inference Hyperparameters: Stage I and Stage II

This appendix lists the test-time hyperparameters of *our* method only (Latent Barrier Steering, Stage I + Stage II). Baseline test-time settings follow each baseline's reference implementation and are not duplicated.

H.1 Stage I (Latent Barrier Steering)

Stage I solves the latent-side finite-time recovery flow of Section 4. The same set of LBS hyperparameters is used across all three domains and all constraint setups; the values are listed in Appendix E.4. For the reader's convenience the operational parameters are:

- Finite-time rate exponent $\rho = 0.5$ and gain $\kappa = 1.0$ (linear-rate variant of the LBS QP).
- Latent Euler step size $\eta = 0.1$.
- Latent iteration cap $N_z = 20$, with early stopping as soon as the soft-min margin $m_\beta(z_i) \geq 0$.
- Reference drift $\gamma_z = 10^{-2}$ (encoder-anchor contraction; reference $u_z^{\mathrm{ref}} = -\gamma_z(z - z_0)$).
- Soft-min sharpness $\beta = 10$.
- Margin slack $\delta = 0.05$.
- Latent clip-box $[-6, 6]^{d_z}$ applied componentwise after every Euler step.

The same finite-difference estimator of $\nabla_z m_\beta$ is used in all domains. The full operational hyperparameter sextuplet $(\kappa, \rho, \eta, N_z, \gamma_z, \beta, \delta)$ is reported as a single row in Appendix E.4 and we do not duplicate it here.

H.2 Stage II (path-space CBF correction)

Stage II takes the Stage I-warm-started rollout and applies a per-step CBF QP along the executed trajectory. The corrector hyperparameters are fixed across domains and constraint setups: a fractional-power rate with tuple $(\varepsilon, \rho, \delta) = (0.3,\ 0.5,\ 0.05)$, where $\rho = 0.5$ is the fractional exponent (matching Stage I), ε scales the fractional gain, and δ is the same activation slack as in Stage I.

The corrector is solved as a small QP over the per-step action a_t, with $\|a_t - a_t^{\mathrm{ref}}\|^2$ as the QP cost and the fractional-rate CBF constraint as the only inequality. We use the unconstrained a_t^{ref} produced by the parameterised generator at the Stage-I-warm-started latent.

H.3 Sampling/integration setup at test time

At test time we use the same sampler discretisation as during training: a 50-step explicit-Euler ODE for the flow-matching generator. Stage II runs along the entire executed horizon.

I Full results across all three constraints

The main-text Table 1 reports Safety, Reach, and Energy. This appendix reports all metrics, including final goal distance, curvature smoothness, and per-episode Stage-I/Stage-II runtime, for all three constraints in each domain. Per-cell entries are means over 50 episodes (60 for Maze2D), aggregated

under the same hyperparameters reported in Appendix E.4 and the evaluation protocol of Appendix F.4. The pretrained flow-matching planner is the same across all methods within a domain. ED is reported in task units; Franka ED is in metres.

Table 6: **Full performance comparison across all test-time safety constraints.** Best per-column value within each constraint block is bolded; ties and values within 10% of the per-column optimum are also bolded. Stage-I and Stage-II runtimes are not bolded. Constraints I, II, III correspond to mode selection, mode interpolation, and residual correction, respectively.

Domain	Constraint	Method	Safety↑	Reach↑	ED↓	Energy↓	CS↓	T-I (s)↓	T-II (s)↓
2D	I	Vanilla FM	12%	**100%**	**0.21**	–	**0.88**	–	0.09
		SAFEDIFFUSER	**100%**	10%	6.38	1.226	1.56	–	0.11
		SAFEFLOW	**100%**	18%	5.57	1.077	1.42	–	0.10
		SAFEFLOWMATCHER	**100%**	18%	5.61	1.166	1.56	–	0.10
		Full LBS	**100%**	**100%**	**0.23**	**0.037**	**0.87**	0.35	0.10
	II	Vanilla FM	0%	**100%**	**0.21**	–	**0.88**	–	0.09
		SAFEDIFFUSER	**100%**	34%	3.75	0.728	1.24	–	0.12
		SAFEFLOW	**100%**	20%	4.90	0.976	1.31	–	0.11
		SAFEFLOWMATCHER	**100%**	22%	4.99	1.051	1.44	–	0.11
		Full LBS	**100%**	90%	0.89	**0.160**	1.03	0.65	0.11
	III	Vanilla FM	0%	**100%**	**0.21**	–	**0.88**	–	0.10
		SAFEDIFFUSER	**100%**	0%	6.83	1.321	1.61	–	0.12
		SAFEFLOW	**100%**	4%	6.44	1.262	1.48	–	0.10
		SAFEFLOWMATCHER	**100%**	4%	6.52	1.366	1.66	–	0.10
		Full LBS	**100%**	82%	0.97	**0.303**	**0.82**	1.80	0.11
Franka	I	Vanilla FM	12%	**100%**	**0.002**	–	**0.14**	–	0.08
		SAFEDIFFUSER	**100%**	0%	0.250	0.0023	0.18	–	0.09
		SAFEFLOW	**100%**	16%	0.214	0.0020	0.17	–	0.08
		SAFEFLOWMATCHER	**100%**	14%	0.241	0.0075	0.79	–	0.08
		Full LBS	**100%**	**100%**	0.016	**0.0004**	0.28	0.80	0.09
	II	Vanilla FM	0%	**100%**	**0.002**	–	**0.14**	–	0.08
		SAFEDIFFUSER	**100%**	0%	0.250	0.0023	0.18	–	0.10
		SAFEFLOW	**100%**	0%	0.251	0.0023	0.17	–	0.09
		SAFEFLOWMATCHER	**100%**	0%	0.276	0.0086	0.87	–	0.09
		Full LBS	**100%**	**100%**	0.017	**0.0004**	0.34	1.10	0.09
	III	Vanilla FM	0%	**100%**	**0.002**	–	**0.14**	–	0.09
		SAFEDIFFUSER	**100%**	0%	0.250	**0.0023**	0.18	–	0.11
		SAFEFLOW	**100%**	0%	0.251	**0.0023**	0.17	–	0.10
		SAFEFLOWMATCHER	**100%**	0%	0.276	0.0086	0.87	–	0.10
		Full LBS	**100%**	98%	0.042	0.0028	0.50	1.55	0.10
Maze2D	I	Vanilla FM	0%	50%	2.69	–	0.17	–	0.44
		SAFEDIFFUSER	**100%**	0%	5.13	0.0288	**0.11**	–	0.63
		SAFEFLOW	**100%**	0%	4.97	0.1208	0.21	–	0.84
		SAFEFLOWMATCHER	**100%**	0%	4.83	0.2668	0.63	–	0.84
		Full LBS	**100%**	**95%**	**0.71**	**0.010**	0.15	4.50	0.68
	II	Vanilla FM	38%	50%	2.69	–	0.17	–	0.42
		SAFEDIFFUSER	**100%**	7%	6.38	**0.0010**	**0.10**	–	0.50
		SAFEFLOW	**100%**	25%	3.31	0.0624	0.20	–	0.70
		SAFEFLOWMATCHER	98%	18%	3.76	0.1404	0.42	–	0.68
		Full LBS	**100%**	**90%**	**0.87**	0.009	0.16	3.80	0.58
	III	Vanilla FM	28%	50%	2.69	–	0.17	–	0.43
		SAFEDIFFUSER	**100%**	7%	6.37	0.0072	**0.10**	–	0.49
		SAFEFLOW	**100%**	38%	2.43	0.0345	0.19	–	0.69
		SAFEFLOWMATCHER	**100%**	53%	2.08	0.0740	0.30	–	0.69
		Full LBS	**100%**	**93%**	**0.27**	**0.004**	0.29	4.10	0.62

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state our two-stage architecture (Stage I latent CBF steering and Stage II path-space correction), the joint-energy stopping rule, and the empirical evaluation across three deployment regimes. Each claim maps to a labelled theorem in Section 4 (Theorems 1 and 2, Proposition 2) or to the experiments in Section 5.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: It's included in Section 6.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: All theoretical results are stated with their assumptions in Sections 4.1–4.3. Full proofs appear in Appendices C–D, with auxiliary derivations of the non-obvious main-text equations collected in Appendix B.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Section 5 describes the three domains, baselines, and metrics; Appendix F gives full domain details (state/action spaces, demonstration counts, horizons, obstacle geometries, model architecture, training objective). All Stage I hyperparameters ($\kappa, \rho, \eta, N_z, \gamma_z, \beta, \delta, h$) are listed in Appendix E.4.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code is attached in the submission.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Per-domain demonstration setups, model size, and training objective are reported in Appendix F. LBS hyperparameters are listed in Appendix E.4 and noted to be domain-independent.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: All main-text tables aggregate 50 episodes per method per setup (60 for Maze2D); standard deviations are computed across episodes within each cell.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix E.3 gives the per-episode generator-call budget ($N_z(2d_z + 1)$ calls for Stage I plus a path-space CBF-QP for Stage II). Section 5 reports per-episode wall-clock T-I and T-II in every main table; 2D experiments run on a single CPU thread and Franka and Maze2D on a single GPU.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work is theoretical and algorithmic safety research for robotic motion planning. It does not involve human subjects, scraped data, or models with high misuse risk, and it conforms with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

Justification: The intended impact is safer deployment of generative planners in robotics, where the proposed framework provides certified path-space safety while preserving multi-modal behavior. We do not foresee direct paths to malicious use; latent steering on top of a generative planner inherits the same dual-use profile as the underlying planner.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release pretrained language models, image generators, or scraped datasets. The released artifacts are small synthetic demonstration datasets and research code with no identified misuse risk.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All baselines (SAFEDIFFUSER, SAFEFLOW, SAFEFLOWMATCHER) and the parameterized flow-matching backbone are cited at first use. The Maze2D layout follows the D4RL `maze2d-large` environment (cited); Franka obstacle geometries follow SAFEFLOW. We use these only for evaluation, in accordance with their original terms of use.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The paper introduces a custom 6-route, 300-trajectory Maze2D demonstration dataset (described in Appendix F.3) and reference implementations of LBS for the three domains. Both will be released with documentation upon publication.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or human-subjects research. All experiments are conducted in simulation.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve human subjects, so no IRB review was required.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are not used as a component of the proposed method; LLM-based assistants were used only for general writing and editing, which does not impact the core methodology, scientific rigor, or originality of the research.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.