TYPICALITY CONSTRAINED LATENT REINFORCEMENT LEARNING FOR GENERATIVE POLICIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion steering via reinforcement learning (DSRL) improves a frozen diffusion or flow-matching policy by learning a latent policy over its input noise. The frozen policy, however, acts well only on noise typical of its Gaussian prior, with a tolerance that shrinks as the dimension grows. Existing steering methods bound the noise in raw units re-tuned per task and need a costly critic distilled through the frozen policy. We propose Typical DSRL, whose latent policy is the prior shifted by a learned mean, so its KL divergence from the prior is exact; bounding it in widths of the prior's typical shell lets one budget serve every dimension. Because its critic sees only typical noise, it needs no distillation. On eleven robomimic, Adroit and Gym tasks, with one budget, Typical DSRL is Pareto-optimal in performance against training time; baselines that come close train two to twenty-four times longer. Typicality is needed only in training: deploying the latent policy's mean, which lies far from the typical shell, outperforms sampling on every task.

1 INTRODUCTION

Diffusion and flow-matching policies trained by behavior cloning have become a standard policy class for robot control, and reinforcement learning is the natural route to improving them beyond their demonstrations (Chi et al., 2025; Black et al., 2025). Fine-tuning their weights is costly, however, because policy-gradient and actor-critic methods alike must optimize through the iterative sampler (Ren et al., 2025; Wang et al., 2023; Zhang et al., 2026). Diffusion steering via reinforcement learning (DSRL; Wagenmaker et al., 2025) instead freezes the weights: a standard off-policy agent learns a *latent policy* over the noise the frozen policy denoises, online, with black-box access to it.

Although any latent in $\mathbb{R}^d$ is a valid input, the frozen policy was trained only on draws from its Gaussian prior, and in dimension d these *typical* draws (MacKay, 2003) have independent coordinates and squared norm $d \pm \sqrt{2d}$, a shell whose relative width narrows as d grows. The policy carries a latent's atypicality into the action: a latent scaled by 1.5 retains 81% of the frozen policy's return at $d = 12$ and never succeeds at $d = 112$, and correlating the steps of a chunk lowers performance on all five probed tasks even on the shell (Section 3.1). Existing methods bound latents in raw units, DSRL by a per-coordinate box tuned for each task and LP-DS (Simsir & Oguz, 2026) by a threshold on the squared norm of a learned residual re-tuned for each environment, so the same setting admits a different fraction of the shell in every dimension and does not transfer. Both also depend on a latent critic distilled from an action critic through the frozen policy at every update (DSRL-NA), and fall behind on most tasks with a temporal-difference critic in its place (DSRL-SAC).

We propose Typical DSRL, which keeps the latent policy typical, with a budget in the prior's own units. Its latent policy is a Gaussian with the prior's covariance whose mean alone is learned, so each sample is a prior draw translated by a state-dependent shift, and its divergence from the prior is exactly half the squared shift. We bound it by $\delta = \eta\sqrt{2d}$ with a Lagrange multiplier in soft actor-critic (Section 3.3), so that the expected squared norm moves by at most 2η shell widths. One η thus serves every dimension, and by data processing the bound is also a likelihood-free trust region on the action policy. Since the critic is trained and queried only on typical latents, a temporal-difference critic suffices, and Typical DSRL trains at the cost of DSRL-SAC.

On eleven tasks from robomimic, Adroit and Gym ($d = 12$ to 240), against six methods that improve the same frozen policy, Typical DSRL with $\eta = 0.5$ lies on the Pareto frontier of performance against

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

training wall-clock on every task (Section 4), best or within one standard deviation of the best on eight, while the baselines that come close take two to twenty-four times as long. Of the methods with a temporal-difference critic, only Typical DSRL keeps up with those with a distilled one, and on door only its sampled latents stay typical (Figure 3). Typicality is needed only in training, however: deploying the latent policy's mean, which lies far from the typical shell, outperforms sampling on every task (Figure 5).

Contributions. (i) A probe showing that frozen diffusion policies carry a latent's atypicality into the action, with a tolerance that shrinks with dimension; (ii) Typical DSRL, which bounds the exact divergence of a mean-shift latent policy from the prior in shell widths; and (iii) an evaluation on eleven tasks, where Typical DSRL matches the strongest baselines at a fraction of their cost and typical latents make a temporal-difference critic sufficient.

2 Related Work

Reinforcement learning for diffusion and flow-matching policies. One line of work fine-tunes these models by policy gradient over the denoising steps, in image generation (Black et al., 2024; Liu et al., 2025) and in control (Zhang et al., 2025; Chen et al., 2025; McAllister et al., 2026). A second line optimizes the policy against a learned critic, backpropagating through the denoising chain or avoiding it through one-step distillation (Park et al., 2025), regression onto a noise-prediction target (Fang et al., 2025), or values of intermediate noisy actions (Selim et al., 2026). A third keeps the model fixed and acts on its outputs, by re-ranking samples with a value function (Nakamoto et al., 2024), learning a residual policy (Yuan et al., 2025; Ankile et al., 2025), or selecting among candidates with a verifier (Kwok et al., 2025) or a density estimate (Rosasco et al., 2025). Closest to our work, DSRL acts on the model's input instead, training a standard off-policy agent whose action is the latent noise, and LP-DS, observing that weakly constrained steering drives latents into low-density regions, constrains the squared norm of a learned noise perturbation.

The initial noise as a control variable. Under a deterministic sampler the initial noise determines the sample. Independently trained models map the same noise to nearly the same output (Zhang et al., 2024), a consistency largely explained by the second-order statistics of the data (Wang et al., 2026). In image generation the noise has accordingly been optimized toward valid regions of noise space (Guo et al., 2024) or against a reward model (Eyring et al., 2024), predicted by a learned network (Zhou et al., 2025), and searched as a form of inference-time scaling (Ma et al., 2025). In trajectory generation, particular noise samples bias motion toward specific semantics (Ota et al., 2026), selecting the noise makes consecutive action chunks consistent (Ho et al., 2026), and a single noise vector per task, found by black-box search from the standard Gaussian with no constraint to the prior, improves frozen robot policies on 46 of 51 task–policy pairs (Patil et al., 2026a).

Typicality of latents. Generative models are reliable only on latents that are typical under their training prior. Linear interpolation between Gaussian samples yields latents whose norm falls several standard deviations below the typical norm and that decode to visibly weaker samples (White, 2016), and sample quality degrades with the mismatch between the distribution of manipulated latents and the prior (Agustsson et al., 2019). The truncation trick (Brock et al., 2019; Karras et al., 2019) exploits the same sensitivity in the opposite direction, trading diversity for fidelity by shrinking latents toward the center, and unconstrained latent-space optimization can move into regions the decoder was never trained on (Gómez-Bombarelli et al., 2018; Griffiths & Hernández-Lobato, 2020). Most directly, Seo et al. (2026) find that optimizing the initial noise of a video world model without constraint leaves the typical set and produces implausible generations, and regularize the norm, blockwise isotropy and spectral flatness of the noise toward their values under the prior.

3 Method

3.1 Effect of Noise on Generated Actions and Closed-Loop Performance

A diffusion or flow-matching policy acts by drawing a latent $\boldsymbol{w} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$, where d is the dimension of a whole action chunk (chunk length times per-step action dimension), and denoising it at

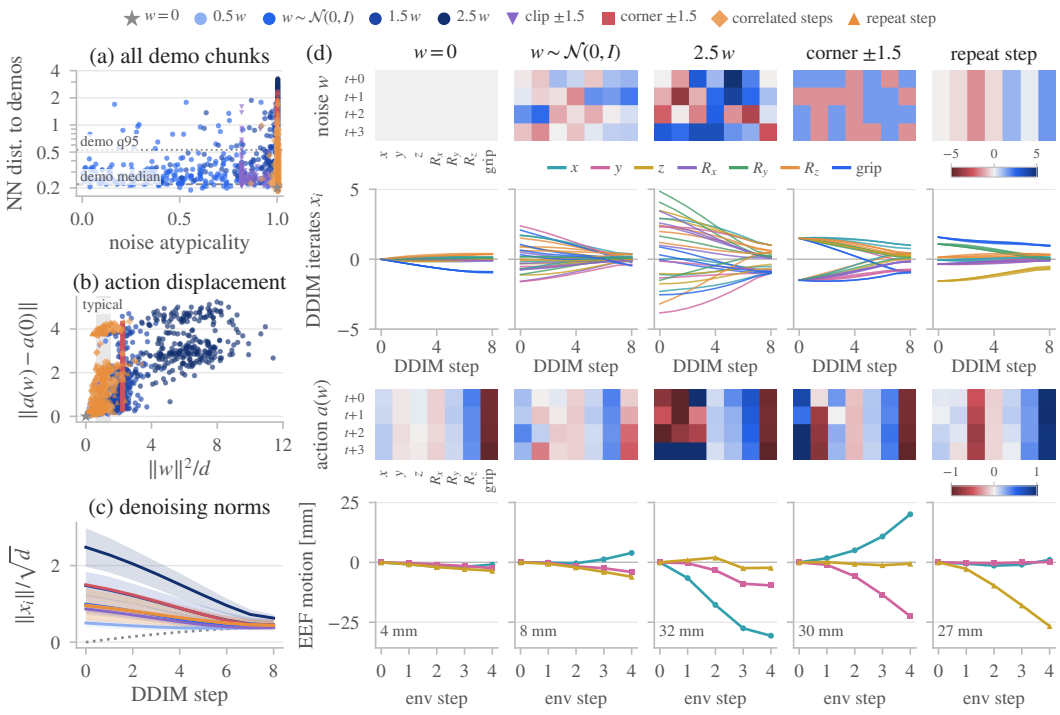

Figure 1: Open-loop probe on can. (a) Distance of the decoded chunk to the nearest demonstration chunk. (b) Displacement from the decode of $\boldsymbol{w} = \mathbf{0}$. (c) Norm of the DDIM iterates, mean and 5–95% band per family. (d) Five latents decoded at one state, followed to the end-effector motion.

the current state $\boldsymbol{s}$. With a deterministic sampler this is a map $\boldsymbol{a} = f_\beta(\boldsymbol{s}, \boldsymbol{w})$ (Section 3.2). Latent-steering methods replace the prior with a learned distribution and feed its samples to the same frozen decoder. The decoder, however, was trained only on prior draws, and in dimension d those are far from arbitrary: $\|\boldsymbol{w}\|^2$ has mean d and standard deviation $\sqrt{2d}$, so almost every draw lies on a thin shell of radius of order $\sqrt{d}$, with independent coordinates and no structure along the chunk. We call latents with these statistics *typical* (Cover & Thomas, 2006). This section asks what the decoder does with an atypical latent, and whether the kind of deviation matters. We probe the frozen diffusion policies that our experiments steer on hopper ($d = 12$), can, lift, square ($d = 28$), and transport ($d = 112$); both *open loop*, decoding 256 latents at one state; and *closed loop*, running full episodes with a fresh latent at every chunk (Section B.1 gives the protocol).

Noise families. Each family transforms a prior draw so as to violate one statistic of the typical set. Scaling ($0.5\,\boldsymbol{w}$, $1.5\,\boldsymbol{w}$, $2.5\,\boldsymbol{w}$) moves the norm off the shell. Clipping every coordinate to ± 1.5 removes the tails, while pinning every coordinate to ± 1.5 with a random sign ("corner") stays in the same box but at its vertices. "Correlated steps" (an autoregressive process with coefficient 0.9 along the chunk) and "repeat step" (one step's noise repeated across the chunk) add temporal correlation at typical norm, and $\boldsymbol{w} = \mathbf{0}$ decodes the prior's mean. We score each latent by its *atypicality*, the tail rank of its most extreme statistic (norm, mean, standard deviation, largest coordinate, step correlation, spectral flatness, isotropy) against 4096 prior draws (0.5: typical; 1: beyond all).

Open loop: the decoder propagates atypicality. On task can (Figure 1), prior draws decode to chunks at the demonstrations' own median distance from the demonstration set, while inflated and corner latents land beyond its 95th percentile at almost every draw (panel a). Two modes emerge in the typical regime (panel b): a lower norm loses this multimodality, and a higher norm generates out-of-distribution action chunks. The displacement from the decode of $\boldsymbol{w} = \mathbf{0}$ grows with $\|\boldsymbol{w}\|^2/d$ once the norm leaves the typical band, and at $2.5\,\boldsymbol{w}$ a fifth of the action coordinates saturate (panel b). Denoising does not undo an atypical start (panel c): it contracts every trajectory toward the data scale, but after eight DDIM steps the final iterate of a prior draw has $\|\boldsymbol{x}_8\|/\sqrt{d} = 0.39$ and a $2.5\times$

draw at 0.63. Followed through the pipeline at one state (Figure 1d), a prior draw moves the end effector by 8 mm, and an inflated, a corner and a repeat-step latent by 32, 30 and 27 mm. The flow-matching policies of the Adroit tasks, which this probe leaves out, pass a latent deviation into the action almost linearly (Figure 4a,b).

Closed loop: which deviations the task forgives. Table 1 pairs initial states across families, so each column compares families on the same episodes. Three regularities hold across tasks.

Table 1: Closed-loop success rate under each noise family (return on hopper), 18 episodes per cell (12 on transport). Best per column in bold.

Family	can	lift $d = 28$	square	transport $d = 112$	hopper $d = 12$
$\boldsymbol{w} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$ (prior)	0.22	**0.78**	0.44	0.25	1446
$0.5\,\boldsymbol{w}$	**0.67**	0.28	0.39	**0.42**	1427
$1.5\,\boldsymbol{w}$	0.22	0.33	0.50	0.00	1170
$2.5\,\boldsymbol{w}$	0.00	0.00	0.00	0.00	688
clip ± 1.5	0.56	**0.78**	**0.67**	**0.42**	1546
corner ± 1.5	0.17	0.22	0.22	0.00	1362
repeat step	0.06	0.28	0.17	0.00	1334
correlated steps	0.11	0.50	0.33	0.00	1349
$\boldsymbol{w} = \mathbf{0}$	0.11	0.00	0.22	0.08	**1563**

Inflating the norm fails, and more sharply as d grows. $1.5\,\boldsymbol{w}$ keeps 81% of the return at $d = 12$, succeeds on 22–50% of episodes at $d = 28$, and never succeeds at $d = 112$. A tolerance in $\|\boldsymbol{w}\|^2/d$ thus does not transfer across policies; it must shrink with the shell width $\sqrt{2/d}$.

Removing tails doesn't hurt. Clip ± 1.5 matches or beats the prior in all five settings; on can, a test in which both arms of an episode consume the same Gaussian stream turns 11 failures into successes against one reversal ($p = 0.006$; Section B.1). Clipping leaves the median distance to the demonstrations unchanged and cuts its 99th percentile by 40%. An episode draws about 75 chunks, so success depends on the worst, which the 99th percentile tracks and the median does not.

Structure fails in closed loop even when it decodes well. Correlated-step and repeat-step chunks lie only $1.2\times$ farther from the demonstrations than prior draws, yet sampled afresh at every chunk they are among the worst families in manipulation, because each chunk commits the arm to a new random direction. Corners are harmful everywhere: corner ± 1.5 is below clip ± 1.5 in every setting. At $d = 112$ every family that inflates the norm, pins the corners or correlates the steps scores zero.

Two requirements follow. (i) The policy may push the latent outward from the shell only within a budget, and since the same inflation costs more as d grows, the budget cannot be a fixed fraction of the norm; we state it in shell widths. (ii) The policy must not correlate the steps within a chunk at random. Neither requirement asks the policy to keep the prior's noise as it is: removing the tails didn't hurt, and scaling the latent down helped or hurt depending on the task.

3.2 Problem Setup

Reinforcement learning. We consider an infinite-horizon discounted Markov decision process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, \rho_0, r, \gamma)$ with state space $\mathcal{S}$, action space $\mathcal{A} \subseteq \mathbb{R}^d$, transition kernel $P(\cdot \mid \boldsymbol{s}, \boldsymbol{a})$, initial-state distribution ρ_0, reward $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and discount $\gamma \in [0, 1)$. A policy $\pi(\cdot \mid \boldsymbol{s})$ is scored by its expected return $J(\pi) = \mathbb{E}_\pi\big[\sum_{t \geq 0} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t)\big]$, and $Q^\pi(\boldsymbol{s}, \boldsymbol{a})$ is its state–action value. As is standard for generative policies, an action $\boldsymbol{a} = (\boldsymbol{a}^{(1)}, \ldots, \boldsymbol{a}^{(H)})$ is a *chunk* of H consecutive low-level commands $\boldsymbol{a}^{(h)} \in \mathbb{R}^m$: one step of $\mathcal{M}$ executes the whole chunk, r is the reward accrued over it, and $d = Hm$ is the dimension of the chunk.

Generative policies as deterministic decoders. We are given a *generative policy* $\pi_\beta(\cdot \mid \boldsymbol{s})$, a diffusion (Ho et al., 2020; Chi et al., 2025) or flow-matching (Lipman et al., 2023; Liu et al., 2023) model over action chunks, pretrained by behavior cloning and frozen. It acts by drawing a latent $\boldsymbol{w} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$ in the *latent-noise space* $\mathcal{W} = \mathbb{R}^d$ and refining it over K denoising or integration steps. With a deterministic sampler—DDIM (Song et al., 2021) without its stochastic term, or a fixed-step ODE solver for flow matching—the sampler is a deterministic *decoder* $f_\beta : \mathcal{S} \times \mathcal{W} \to \mathcal{A}$, $\boldsymbol{a} = f_\beta(\boldsymbol{s}, \boldsymbol{w})$, and $\pi_\beta(\cdot \mid \boldsymbol{s})$ is the distribution of $f_\beta(\boldsymbol{s}, \boldsymbol{w})$ for $\boldsymbol{w} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$. We assume only black-box access to f_β, without gradients.

Latent-action MDP. Because f_β is deterministic, choosing a latent is choosing an action. Following DSRL, we lift $\mathcal{M}$ to the *latent-action MDP* $\mathcal{M}^{\mathcal{W}} = (\mathcal{S}, \mathcal{W}, P^{\mathcal{W}}, \rho_0, r^{\mathcal{W}}, \gamma)$ with $P^{\mathcal{W}}(\cdot \mid$

Algorithm 1 Typical DSRL: constrained soft actor–critic in the latent-action MDP $\mathcal{M}^{\mathcal{W}}$.

1: **Input:** decoder f_β, budget η, temperature α; $\delta = \eta\sqrt{2d}$, $\boldsymbol{\Delta}_\theta \equiv \mathbf{0}$, $\lambda = 1$, $\mathcal{B} \leftarrow$ rollouts of π_β.
2: **for** each environment step **do**
3: Draw $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$, set $\boldsymbol{w} = \boldsymbol{\Delta}_\theta(\boldsymbol{s}) + \boldsymbol{\epsilon}$, execute $f_\beta(\boldsymbol{s}, \boldsymbol{w})$, store $(\boldsymbol{s}, \boldsymbol{w}, r^{\mathcal{W}}, \boldsymbol{s}')$ in $\mathcal{B}$.
4: **for** g gradient steps on minibatches from $\mathcal{B}$ **do**
5: Critics: regress onto Equation (6); Polyak-update the targets.
6: Actor: descend Equation (7) at fixed λ; ascend $\log \lambda$ on the violation of Equation (5).
7: **Deploy:** $\boldsymbol{a} = f_\beta(\boldsymbol{s}, \boldsymbol{\Delta}_\theta(\boldsymbol{s}))$.

$\boldsymbol{s}, \boldsymbol{w}) = P(\cdot \mid \boldsymbol{s}, f_\beta(\boldsymbol{s}, \boldsymbol{w}))$ and $r^{\mathcal{W}}(\boldsymbol{s}, \boldsymbol{w}) = r(\boldsymbol{s}, f_\beta(\boldsymbol{s}, \boldsymbol{w}))$. A *latent policy* $\pi_\theta^{\mathcal{W}}(\cdot \mid \boldsymbol{s})$ induces the action policy $\pi_\theta(\cdot \mid \boldsymbol{s})$, the distribution of $f_\beta(\boldsymbol{s}, \boldsymbol{w})$ for $\boldsymbol{w} \sim \pi_\theta^{\mathcal{W}}(\cdot \mid \boldsymbol{s})$, with the same return, $J(\pi_\theta) = J^{\mathcal{W}}(\pi_\theta^{\mathcal{W}})$, so improving π_β reduces to reinforcement learning in $\mathcal{M}^{\mathcal{W}}$ with a latent critic $Q^{\mathcal{W}} : \mathcal{S} \times \mathcal{W} \to \mathbb{R}$, the frozen policy being part of the environment. The prior is the latent policy that induces π_β, and each noise family of Section 3.1 is a state-independent latent policy.

Typical latents. For $\boldsymbol{w} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$, $\|\boldsymbol{w}\|^2$ is χ^2_d-distributed with relative standard deviation $\sqrt{2/d}$, so $\boldsymbol{w}$ concentrates on a shell of radius $\approx \sqrt{d}$ that thins with dimension (Laurent & Massart, 2000). We call the latents within c standard deviations of this shell the *typical set* of the prior,

$$\mathcal{T}_d(c) = \left\{\boldsymbol{w} \in \mathcal{W} : \ |\|\boldsymbol{w}\|^2 - d| \le c\sqrt{2d}\right\}. \tag{1}$$

The origin, the corners of a box $[-b, b]^d$ and any latent rescaled by a factor other than one may fall outside it once d is moderately large. For latent policies with the prior's covariance, the divergence from the prior measures how far they leave the typical set: for $\pi_\theta^{\mathcal{W}}(\cdot \mid \boldsymbol{s}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{I}_d)$, $D_{\mathrm{KL}}\big(\pi_\theta^{\mathcal{W}}(\cdot \mid \boldsymbol{s}) \,\|\, \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)\big) = \frac{1}{2}\|\boldsymbol{\mu}\|^2$ while $\mathbb{E}\|\boldsymbol{w}\|^2 = d + \|\boldsymbol{\mu}\|^2$, so a divergence of δ nats moves the expected squared norm by $2\delta/\sqrt{2d}$ shell widths, the unit requirement (i) asks for.

Problem statement. Given black-box access to f_β and online interaction with $\mathcal{M}$, we learn a latent policy that maximizes return while remaining typical:

$$\max_\theta \ J^{\mathcal{W}}(\pi_\theta^{\mathcal{W}}) \quad \text{s.t.} \quad \mathbb{E}_{\boldsymbol{s}}\Big[D_{\mathrm{KL}}\big(\pi_\theta^{\mathcal{W}}(\cdot \mid \boldsymbol{s}) \,\|\, \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)\big)\Big] \le \delta, \tag{2}$$

where the expectation is over the states the learner encounters and $\delta > 0$ is the typicality budget. Two properties make Equation (2) the right object. First, the constraint is exact and cheap: for a Gaussian latent policy the divergence has a closed form, whereas the divergence between π_θ and π_β in action space is not computable for a diffusion policy. Second, it is still a trust region on actions: π_θ and π_β are $\pi_\theta^{\mathcal{W}}$ and $\mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$ passed through the same map $f_\beta(\boldsymbol{s}, \cdot)$, so the data-processing inequality gives, for every state,

$$D_{\mathrm{KL}}\big(\pi_\theta(\cdot \mid \boldsymbol{s}) \,\|\, \pi_\beta(\cdot \mid \boldsymbol{s})\big) \ \le \ D_{\mathrm{KL}}\big(\pi_\theta^{\mathcal{W}}(\cdot \mid \boldsymbol{s}) \,\|\, \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)\big), \tag{3}$$

LP-DS introduced a constraint of this form as a small-perturbation approximation; for the policy class of Section 3.3 it is exact.

3.3 TYPICAL DSRL

Typical DSRL solves Equation (2) by soft actor–critic in $\mathcal{M}^{\mathcal{W}}$ with three choices that answer the requirements of Section 3.1. The latent policy keeps the prior's noise and learns only a state-dependent shift of its mean (requirement ii); the budget on that shift is stated in shell widths (requirement i); and the budget is enforced by a Lagrange multiplier. Algorithm 1 summarizes the procedure.

A mean-shift latent policy. The latent policy is a Gaussian with the prior's covariance whose mean alone is learned,

$$\pi_\theta^{\mathcal{W}}(\cdot \mid \boldsymbol{s}) = \mathcal{N}\big(\boldsymbol{\Delta}_\theta(\boldsymbol{s}), \boldsymbol{I}_d\big), \quad \text{i.e.} \quad \boldsymbol{w} = \boldsymbol{\Delta}_\theta(\boldsymbol{s}) + \boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d), \tag{4}$$

with $\boldsymbol{\Delta}_\theta : \mathcal{S} \to \mathcal{W}$ a network whose output layer is initialized to zero, so that learning starts exactly at π_β. A sample is a prior draw translated by $\boldsymbol{\Delta}_\theta(\boldsymbol{s})$: its coordinates stay independent, of unit variance and Gaussian-tailed, with no correlation along the chunk beyond what the state-dependent mean introduces. Requirement (ii) thus holds by construction; only the mean is left to constrain.[1]

[1] A $[-6, 6]^d$ latent clamp counts as part of f_β, so Equation (4) stays exact (Section B.2).

A budget in shell widths. For Equation (4) the divergence is exactly $D_{\mathrm{KL}}(\pi_\theta^{\mathcal{W}}(\cdot \mid \boldsymbol{s}) \,\|\, \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)) = \frac{1}{2}\|\boldsymbol{\Delta}_\theta(\boldsymbol{s})\|^2$, so the constraint is a single scalar, which we impose on average over the replay buffer $\mathcal{B}$:

$$\mathbb{E}_{\boldsymbol{s}\sim\mathcal{B}}\Big[\tfrac{1}{2}\|\boldsymbol{\Delta}_\theta(\boldsymbol{s})\|^2\Big] \;\le\; \delta \;=\; \eta\sqrt{2d}. \tag{5}$$

The budget lets $\mathbb{E}\|\boldsymbol{w}\|^2$ move by at most 2η shell widths, whatever d is (requirement (i)). The dimensionless η is the constraint's only hyperparameter; we use $\eta = 0.5$ on every task, and the budget is not tuned per task (Figure 4c,d). By Equation (3), δ also bounds the divergence of the action policy from π_β: a trust region (Schulman et al., 2015) that needs no likelihood from the generative model. The budget also keeps the critic's inputs in distribution: every latent it scores is a prior draw translated by a shift of RMS norm at most $\sqrt{2\delta}$, a fraction of the shell radius that falls as $d^{-1/4}$, rather than an arbitrary point of a box whose corners lie $b\sqrt{d}$ from the origin.

Constrained soft actor–critic. We run soft actor–critic (Haarnoja et al., 2018a) in $\mathcal{M}^{\mathcal{W}}$: the actor emits a latent, the environment decodes and executes it, and the transition $(\boldsymbol{s}, \boldsymbol{w}, r^{\mathcal{W}}, \boldsymbol{s}')$ enters $\mathcal{B}$ with the reward summed over the chunk. Critics $Q_{\phi_1}^{\mathcal{W}}, Q_{\phi_2}^{\mathcal{W}}$ regress onto the soft target

$$y = r^{\mathcal{W}} + \gamma\Big[\min_j Q_{\bar{\phi}_j}^{\mathcal{W}}(\boldsymbol{s}', \boldsymbol{w}') - \alpha \log \pi_\theta^{\mathcal{W}}(\boldsymbol{w}' \mid \boldsymbol{s}')\Big], \qquad \boldsymbol{w}' \sim \pi_\theta^{\mathcal{W}}(\cdot \mid \boldsymbol{s}'), \tag{6}$$

with $\bar{\phi}_j$ Polyak averages of ϕ_j and the bracket dropped at terminal transitions. The constraint enters the actor through a Lagrange multiplier (Altman, 1999): with $\boldsymbol{w}$ reparameterized as in Equation (4) and λ held fixed, the actor minimizes

$$\mathcal{L}_{\mathrm{actor}}(\theta) = \mathbb{E}_{\boldsymbol{s}\sim\mathcal{B},\, \boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I}_d)}\Big[-\min_j Q_{\phi_j}^{\mathcal{W}}\big(\boldsymbol{s}, \boldsymbol{\Delta}_\theta(\boldsymbol{s}) + \boldsymbol{\epsilon}\big)\Big] + \lambda\, \mathbb{E}_{\boldsymbol{s}\sim\mathcal{B}}\Big[\tfrac{1}{2}\|\boldsymbol{\Delta}_\theta(\boldsymbol{s})\|^2\Big], \tag{7}$$

and $\log\lambda$ ascends the violation after every actor step (Algorithm 1). In the checkpoints we inspect, the constraint is active at the end of training (Table 11), so the budget, not the critic, decides how far the policy leaves the prior. The update ratio g and other constants are in Section B.2.

Temperature. With the variance fixed, the entropy of $\pi_\theta^{\mathcal{W}}$ is constant: the entropy term of the actor objective has no gradient in θ and is absent from Equation (7), and α acts only through the critic target Equation (6). Tuning α against an entropy target (Haarnoja et al., 2018b) is therefore not meaningful, and in practice it drives α far from the values we use. We fix α per task (Section B.2; Figure 9 varies it on square).

Deployment. At test time we act with the maximum-likelihood latent action $\boldsymbol{w} = \boldsymbol{\Delta}_\theta(\boldsymbol{s})$, the mode of Equation (4), and execute its decode $\boldsymbol{a} = f_\beta(\boldsymbol{s}, \boldsymbol{\Delta}_\theta(\boldsymbol{s}))$; unless $f_\beta(\boldsymbol{s}, \cdot)$ is affine, this decode is neither the mode nor the mean of $\pi_\theta(\cdot \mid \boldsymbol{s})$. The noise $\boldsymbol{\epsilon}$ serves exploration; removing it raises success on every task (Figure 5). This does not contradict Section 3.1, where decoding $\boldsymbol{w} = \mathbf{0}$ was best on hopper and worst on lift: that latent ignores the state, whereas $\boldsymbol{\Delta}_\theta(\boldsymbol{s})$ is chosen per state.

Relation to DSRL and LP-DS. Typical DSRL is DSRL with its box on latents replaced by Equation (5), trained with plain soft actor–critic on a temporal-difference latent critic, where DSRL's noise-aliased variant, and LP-DS after it, distill the latent critic from an action critic through the decoder. It constrains the same quantity as LP-DS, the squared mean shift. LP-DS, however, conditions its perturbation on the base noise, $\boldsymbol{w} = \boldsymbol{\epsilon} + \boldsymbol{\Delta}_\theta(\boldsymbol{s}, \boldsymbol{\epsilon})$, so its latent policy is not Gaussian, the squared shift only approximates the divergence, and its threshold is set in raw units per environment. Here half the squared shift is the divergence, and $\delta = \eta\sqrt{2d}$ absorbs the dimension so that one η serves from $d = 12$ to $d = 240$.

4 Experiments

4.1 Experimental Setup

Tasks. We evaluate on eleven state-based tasks whose latent dimension spans $d = 12$ to 240: *lift*, *can*, *square* and *transport* from robomimic (Mandlekar et al., 2021) and *door*, *hammer*, *pen* and *relocate* from Adroit (Rajeswaran et al., 2018), scored by sparse success, and *hopper*, *walker2d* and

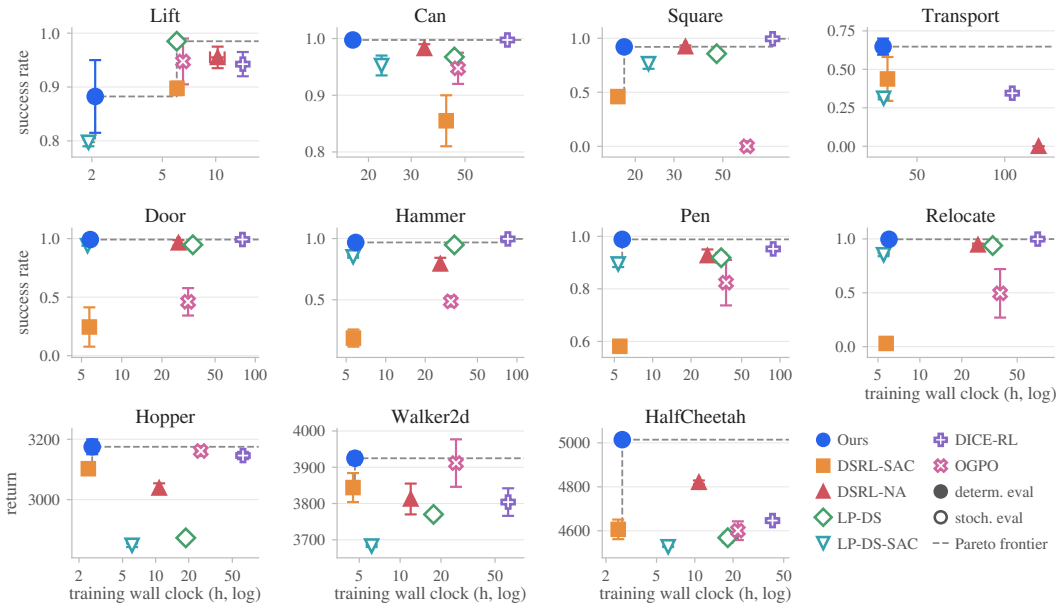

Figure 2: Plateau of each method as it is deployed, mean ± standard deviation over seeds.

halfcheetah from Gym (Brockman et al., 2016; Todorov et al., 2012), scored by return. Each task has one frozen policy that every method steers: the DDIM-sampled diffusion policies of DSRL on robomimic and Gym, and flow-matching policies that we trained on the D4RL (Fu et al., 2020) `expert` data on Adroit (Table 5).

Protocol. Every method starts from the same frozen policy and learns online, with no demonstrations, for a fixed budget of 50k to 500k chunk-level environment steps (Table 6). We report the *plateau* of a run, the mean of the last quarter of its evaluation points, as mean ± standard deviation over two seeds, together with its training wall-clock, since the methods differ widely in cost per step. Latent policies are evaluated *stochastically*, sampling $\boldsymbol{w} \sim \pi_\theta^{\mathcal{W}}(\cdot \mid \boldsymbol{s})$ as in training, and *deterministically*, with one latent per state, for Typical DSRL the maximum-likelihood latent action $\boldsymbol{w} = \boldsymbol{\Delta}_\theta(\boldsymbol{s})$ with which it is deployed. Typical DSRL uses one configuration on every task, $\eta = 0.5$ and a temperature α fixed per task, with every other setting inherited from DSRL-SAC (Section B.2).

Baselines. Six methods improve the same frozen policy from the same interaction. Four steer the latent: the two variants of DSRL, whose latent critic is trained by temporal differences (DSRL-SAC) or distilled from an action critic through the decoder (DSRL-NA, for noise aliasing); LP-DS, which bounds a noise-conditioned residual and trains its critics as DSRL-NA does; and LP-DS-SAC, our ablation of LP-DS with a temporal-difference critic, so that LP-DS/LP-DS-SAC isolates the distilled critic as DSRL-NA/DSRL-SAC does. Two act on the action: DICE-RL (Sun & Song, 2026) adds a learned residual to the decoded chunk and OGPO (Patil et al., 2026b) fine-tunes the generator itself, and both act by best-of-16 and best-of-8 search under their critic. Section B.3 gives every baseline's settings and the choices that porting DICE-RL and OGPO to our protocol required.

4.2 Results

Typical DSRL matches the strongest baselines at a fraction of their cost. Typical DSRL lies on the Pareto frontier of Figure 2 on all eleven tasks. Its plateau is the best, or within a standard deviation of the best, on eight of them; it trails DICE-RL on square and hammer and LP-DS on lift (Table 8). Part of this lead is the maximum-likelihood latent action with which Typical DSRL is deployed (Figure 5). It shares the code, networks and update ratios of DSRL-SAC and trains at most ten percent longer on any task (Table 10). The baselines that come close take two to twenty-four times as long, because they train their latent critic through the frozen policy (DSRL-NA, LP-DS) or

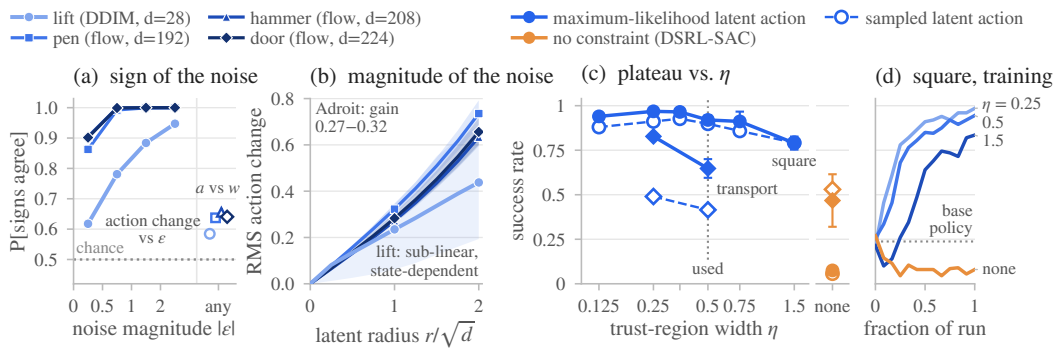

Figure 4: (a,b) Decoders of Typical DSRL's final checkpoints at visited states. (a) Sign agreement of a noise coordinate with the action change it causes, and of the latent with the action (right). (b) Action change along random directions from $\boldsymbol{\Delta}_\theta(s)$; band 10–90%. (c,d) Varying the trust-region width η of Typical DSRL, all else fixed. (c) Whiskers: one standard deviation over seeds. (d) Success of the maximum-likelihood latent action, mean over seeds.

act by test-time search over candidate actions (DICE-RL, OGPO). OGPO collapses on square; we omit it on transport, where one seed would take an estimated 178 hours.

Typical latents make a temporal-difference critic sufficient. Of the three methods that train their latent critic by temporal differences, DSRL-SAC, LP-DS-SAC and Typical DSRL, only Typical DSRL keeps up with the methods that distill it. DSRL-SAC falls behind as the latent grows, to 0.58 or below on the four Adroit tasks ($d \geq 192$) where Typical DSRL reaches 0.97 or above. LP-DS-SAC is below LP-DS on every task. For DSRL and LP-DS, then, the distilled critic is load-bearing, and it is what their extra wall-clock pays for. Typical DSRL does without it and still exceeds DSRL-NA and LP-DS on every completed task except lift, where both lead, and square, where it ties DSRL-NA. On door ($d = 224$), its critic, trained only on typical latents, has the scale and shape of DSRL-NA's distilled one (Figure 3).

Only Typical DSRL's latents are typical. We record the latents that the final checkpoints of Typical DSRL, DSRL-SAC and DSRL-NA execute on door and the values their critics assign (Section C.3). A sample of Typical DSRL looks like a shifted prior draw: its energy, largest coordinate and atypicality score are close to a prior draw's (Table 11, Figure 3a). Its divergence from the prior ends at 9.9 nats against a budget of 10.6, so the constraint is still active at the end of training. Both DSRL actors are atypical, in opposite directions. DSRL-NA shrinks its latents and clips them at the box. DSRL-SAC inflates them to 2.5 times the prior's energy and succeeds on only 0.17 of episodes, the inflated-norm failure of Table 1 produced here by a learner. Its critic is the only one that values these latents above prior draws. Relative to the return its policy earns, it is also over ten times higher than the critics of DSRL-NA and Typical DSRL, which agree with each other (Section C.3).

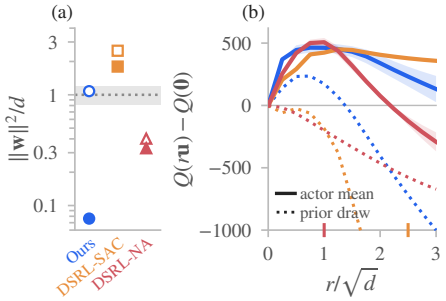

Figure 3: Final checkpoints on door. (a) Energy of the executed latent, stochastic (hollow) and deterministic (filled), one checkpoint per method; band: $\mathcal{T}_d(2)$. (b) Latent critic along a ray, mean ± SD over seeds; ticks: box half-width b of the DSRL actors.

The trust region prevents collapse. The budget $\delta = \eta\sqrt{2d}$ has one free number, η, and every result above uses $\eta = 0.5$. Figure 4c,d vary it on square and transport. Without the constraint, DSRL-SAC collapses during training on square: success falls below the frozen policy's and never recovers (Figure 4d). The tighter the constraint, the smaller the drop: at $\eta = 1.5$ success still dips below the

frozen policy before it climbs, and at narrower widths it does not dip at all. The plateau is flat for $\eta \leq 0.75$ and lower at $\eta = 1.5$ (Figure 4c). At every width the divergence ends at its budget, so the constraint stays active. Although $\eta = 0.25$ performs better on both tasks, we use $\eta = 0.5$ everywhere, so Figure 2 is not tuned to η.

How the latent maps to the action. To see what the latent does to the action, we decode perturbed latents at visited states of lift, door, hammer and pen through the final checkpoints of Typical DSRL (Section C.4). On the three Adroit tasks the frozen policy is almost exactly affine in the latent, with a nearly diagonal Jacobian: each latent coordinate moves its own action coordinate, in proportion, and the sign of the noise sets the direction of the action change, while the state sets the action itself (Figure 4a,b). These policies are therefore close to Gaussian at every state, steering their latent adds a residual to the action, and Equation (3) holds nearly with equality, so the latent budget is in effect a budget on the action policy. Lift's policy is far from affine, low-rank and bimodal in the gripper, but there too the noise pushes the action in its own direction. On both, as on the diffusion policies of Section 3.1, a deviation of the latent reaches the action rather than being corrected. The Adroit and lift policies differ in both generator and demonstrations, so we do not attribute the difference to either. The budget holds only on average over states, so the actor spends it where the task needs it (Section C.4): the divergence peaks at the grasp on lift and the strike on hammer, and grows once the door opens.

Maximum-likelihood latent action. Deploying Typical DSRL with the maximum-likelihood latent action $\boldsymbol{w} = \boldsymbol{\Delta}_\theta(\boldsymbol{s})$ instead of a sampled one raises success on every task (Figure 5), because it removes action noise. Decoded, the prior's noise scatters the action around the maximum-likelihood action by more than the learned shift moves it (Figure 6). Test-time interventions on the latent confirm that this scatter costs success: success falls as the noise grows, and averaging a few decoded samples recovers the maximum-likelihood success rate (Figure 7). The gain is largest on transport and large on door and hammer, which tolerate little action noise, but small on pen. During training, however, the noise is needed: it keeps the sampled latents typical, and these are the latents the critic is trained and queried on (Figure 3a). Typical DSRL therefore removes it only at deployment, whereas DSRL-NA reduces its noise during training, at the cost of atypical latents, and gains almost nothing from deterministic evaluation (Tables 8 and 9). Since the deployed latent is far from typical yet raises success, typicality is a requirement on the latents the learner trains on and its critic scores, not on the latent it deploys.

5 Conclusion

We asked how far a latent policy may leave the prior of the frozen generative policy it steers. Frozen diffusion policies carry a latent's atypicality into the action, with a tolerance that shrinks as the dimension grows, so Typical DSRL learns only the mean of a latent policy with the prior's covariance and bounds its exact divergence from the prior in shell widths. With $\eta = 0.5$ on all eleven tasks, it matches the strongest baselines at the cost of DSRL-SAC, and typical latents make its temporal-difference critic sufficient. Typicality is needed only during training, however: at deployment, the latent policy's mean lies far from the typical shell and still outperforms sampling on every task.

Limitations. Typical DSRL refines the frozen policy rather than replacing it: by Equation (3), its action policy stays within δ nats of the frozen one on average over states, so behavior the frozen policy rarely produces stays out of reach. The temperature α is still set per task. Results average at most two seeds per cell on state-based, simulated tasks; image-based and large pretrained policies on real robots are the natural next step.

AI use statement

In this work, we used generative AI tools to help develop conceptual frameworks, formulate mathematical claims, propose or refine hypotheses, design and provide feedback on experiments, implement methods, and interpret results. We have not used generative AI tools to assist with translation or to clean and reformat datasets, and generating synthetic data sets, proving mathematical claims or writing proofs, and qualitative and thematic data analysis are not applicable to this work. Additionally, we used generative AI tools to create scientific figures, suggest experimental parameters,

create and edit software code, brainstorm, summarize existing literature, suggest the structure of the paper, draft parts of the paper, edit it for readability, and format references. We have reviewed all AI-assisted work: generated code was tested with unit tests and smoke runs before use, and the conclusions of the paper were drawn by the authors. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

Ethics statement

This work uses only simulated benchmarks and publicly available demonstration data; it involves no human subjects or personal data and releases no new dataset. Typical DSRL fine-tunes robot control policies, and a policy fine-tuned by reinforcement learning can behave unexpectedly, so deploying one on a physical robot would need the usual safety measures. We are not aware of other ethical concerns raised by this work.

Reproducibility statement

We will release our code upon acceptance. Typical DSRL is specified in Section 3.3 and Algorithm 1, and Section B gives the experiment details: the noise probe settings (Section B.1); the tasks, frozen policies, and training and evaluation protocol (Section B.2); and the baseline settings (Section B.3).

References

Eirikur Agustsson, Alexander Sage, Radu Timofte, and Luc Van Gool. Optimal transport maps for distribution preserving operations on latent spaces of Generative Models. In *International Conference on Learning Representations (ICLR)*, 2019. arXiv:1711.01970.

Eitan Altman. *Constrained Markov Decision Processes*. Chapman & Hall/CRC, 1999.

Lars Ankile, Zhenyu Jiang, Rocky Duan, Guanya Shi, Pieter Abbeel, and Anusha Nagabandi. Residual Off-Policy RL for Finetuning Behavior Cloning Policies. *arXiv preprint arXiv:2509.19301*, 2025.

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

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training Diffusion Models with Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2305.13301.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. In *Proceedings of Robotics: Science and Systems (RSS)*, 2025.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large Scale GAN Training for High Fidelity Natural Image Synthesis. In *International Conference on Learning Representations (ICLR)*, 2019. arXiv:1809.11096.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI gym. *arXiv preprint arXiv:1606.01540*, 2016.

Kang Chen, Zhihao Liu, Tonghe Zhang, Zhen Guo, Si Xu, Hao Lin, Hongzhi Zang, Xiang Li, Quanlu Zhang, Zhaofei Yu, Guoliang Fan, Tiejun Huang, Yu Wang, and Chao Yu. π_{RL}: Online RL Fine-tuning for Flow-based Vision-Language-Action Models. *arXiv preprint arXiv:2510.25889*, 2025.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10–11):1684–1704, 2025.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley-Interscience, second edition, 2006.

Luca Eyring, Shyamgopal Karthik, Karsten Roth, Alexey Dosovitskiy, and Zeynep Akata. ReNO: Enhancing One-step Text-to-Image Models through Reward-based Noise Optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. arXiv:2406.04312.

Linjiajie Fang, Ruoxue Liu, Jing Zhang, Wenjia Wang, and Bing-Yi Jing. Diffusion Actor-Critic: Formulating Constrained Policy Iteration as Diffusion Noise Regression for Offline Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2405.20555.

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.

Ryan-Rhys Griffiths and José Miguel Hernández-Lobato. Constrained Bayesian Optimization for Automatic Chemical Design using Variational Autoencoders. *Chemical Science*, 11(2):577–586, 2020. arXiv:1709.05501.

Xiefan Guo, Jinlin Liu, Miaomiao Cui, Jiankai Li, Hongyu Yang, and Di Huang. InitNO: Boosting Text-to-Image Diffusion Models via Initial Noise Optimization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. arXiv:2404.04650.

Rafael Gómez-Bombarelli, Jennifer N. Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D. Hirzel, Ryan P. Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS Central Science*, 2018. arXiv:1610.02415.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning (ICML)*, 2018a.

Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018b.

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

Trong-Bao Ho, Quang-Tan Nguyen, Thien-Loc Ha, Gia-Binh Nguyen, Viet-Thanh Nguyen, Long Dinh, Minh N. Vu, Duy M. H. Nguyen, An Thai Le, and Ngo Anh Vien. Start Right, Arrive Right: Asynchronous Execution via Initial Noise Selection. *arXiv preprint arXiv:2606.19774*, 2026.

Tero Karras, Samuli Laine, and Timo Aila. A Style-Based Generator Architecture for Generative Adversarial Networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. arXiv:1812.04948.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

Jacky Kwok, Christopher Agia, Rohan Sinha, Matt Foutter, Shulu Li, Ion Stoica, Azalia Mirhoseini, and Marco Pavone. RoboMonkey: Scaling Test-Time Sampling and Verification for Vision-Language-Action Models. In *Conference on Robot Learning (CoRL)*, 2025. arXiv:2506.17811.

Béatrice Laurent and Pascal Massart. Adaptive estimation of a quadratic functional by model selection. *The Annals of Statistics*, 28(5):1302–1338, 2000.

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 (ICLR)*, 2023. URL `https://arxiv.org/abs/2210.02747`.

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-GRPO: Training Flow Matching Models via Online RL. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2505.05470.

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 (ICLR)*, 2023. URL https://arxiv.org/abs/2209.03003.

Nanye Ma, Shangyuan Tong, Haolin Jia, Hexiang Hu, Yu-Chuan Su, Mingda Zhang, Xuan Yang, Yandong Li, Tommi Jaakkola, Xuhui Jia, and Saining Xie. Inference-Time Scaling for Diffusion Models beyond Scaling Denoising Steps. *arXiv preprint arXiv:2501.09732*, 2025.

David J. C. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

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 (CoRL)*, 2021.

David McAllister, Songwei Ge, Brent Yi, Chung Min Kim, Ethan Weber, Hongsuk Choi, Haiwen Feng, and Angjoo Kanazawa. Flow Matching Policy Gradients. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2507.21053.

Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, 1947.

Mitsuhiko Nakamoto, Oier Mees, Aviral Kumar, and Sergey Levine. Steering Your Generalists: Improving Robotic Foundation Models via Value Guidance. In *Conference on Robot Learning (CoRL)*, 2024. arXiv:2410.13816.

Sakuya Ota, Qing Yu, Kent Fujiwara, Satoshi Ikehata, and Ikuro Sato. Retrieving and Refining Winning Noise Tickets for Diffusion-Based Motion Generation. In *European Conference on Computer Vision (ECCV)*, 2026. arXiv:2607.06843.

Seohong Park, Qiyang Li, and Sergey Levine. Flow Q-Learning. In *International Conference on Machine Learning (ICML)*, 2025. arXiv:2502.02538.

Omkar Patil, Ondrej Biza, Thomas Weng, Karl Schmeckpeper, Wil Thomason, Xiaohan Zhang, Kausik Sivakumar, Robin Walters, Nakul Gopalan, Sebastian Castro, Stephen Hart, and Eric Rosen. You've Got a Golden Ticket: Improving Generative Robot Policies With A Single Noise Vector. *arXiv preprint arXiv:2603.15757*, 2026a.

Sarvesh Patil, Mitsuhiko Nakamoto, Manan Agarwal, Shashwat Saxena, Jesse Zhang, Giri Anantharaman, Cleah Winston, Chaoyi Pan, Douglas Chen, Nai-Chieh Huang, Zeynep Temel, Oliver Kroemer, Sergey Levine, Abhishek Gupta, Hongkai Dai, Paarth Shah, and Max Simchowitz. OGPO: Sample efficient full-finetuning of generative control policies. *arXiv preprint arXiv:2605.03065*, 2026b.

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. In *Robotics: Science and Systems (RSS)*, 2018.

Allen Z Ren, Justin Lidard, Lars L Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2409.00588.

Andrea Rosasco, Federico Ceola, Giulia Pasquale, and Lorenzo Natale. KDPE: A Kernel Density Estimation Strategy for Diffusion Policy Trajectory Selection. In *Conference on Robot Learning (CoRL)*, 2025. arXiv:2508.10511.

John Schulman, Sergey Levine, Pieter Abbeel, Michael I. Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning (ICML)*, pp. 1889–1897, 2015.

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

Mahmoud Selim, Cristina Cipriani, and Karl H. Johansson. Noisy-Space Policy Gradient for Diffusion Policies in Offline Reinforcement Learning. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2609.06882.

Junwon Seo, Sushant Veer, Ran Tian, Wenhao Ding, Apoorva Sharma, Karen Leung, Edward Schmerling, Marco Pavone, and Andrea Bajcsy. StressDream: Steering video world models for robust policy evaluation and improvement. In *Conference on Robot Learning (CoRL)*, 2026. arXiv:2606.00267.

Hikmet Simsir and Ozgur S. Oguz. Lagrangian perturbation diffusion steering: Latent reinforcement learning for generative policies. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2606.01151.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations (ICLR)*, 2021. arXiv:2010.02502.

Zhanyi Sun and Shuran Song. From prior to pro: Efficient skill mastery via distribution contractive RL finetuning. *arXiv preprint arXiv:2603.10263*, 2026.

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

Andrew Wagenmaker, Mitsuhiko Nakamoto, Yunchu Zhang, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering your diffusion policy with latent space reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2025. arXiv:2506.15799.

Binxu Wang, Jacob Zavatone-Veth, and Cengiz Pehlevan. A Random Matrix Theory Perspective on the Consistency of Diffusion Models. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2602.02908.

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 (ICLR)*, 2023. arXiv:2208.06193.

Tom White. Sampling Generative Networks. *arXiv preprint arXiv:1609.04468*, 2016.

Xiu Yuan, Tongzhou Mu, Stone Tao, Yunhao Fang, Mengke Zhang, and Hao Su. Policy Decorator: Model-Agnostic Online Refinement for Large Policy Model. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2412.13630.

Huijie Zhang, Jinfan Zhou, Yifu Lu, Minzhe Guo, Peng Wang, Liyue Shen, and Qing Qu. The Emergence of Reproducibility and Generalizability in Diffusion Models. In *International Conference on Machine Learning (ICML)*, 2024. arXiv:2310.05264.

Tonghe Zhang, Chao Yu, Sichang Su, and Yu Wang. ReinFlow: Fine-tuning Flow Matching Policy with Online Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025. arXiv:2505.22094.

Yixian Zhang, Shu'ang Yu, Tonghe Zhang, Mo Guang, Haojia Hui, Kaiwen Long, Yu Wang, Chao Yu, and Wenbo Ding. SAC Flow: Sample-Efficient Reinforcement Learning of Flow-Based Policies via Velocity-Reparameterized Sequential Modeling. In *International Conference on Learning Representations (ICLR)*, 2026. arXiv:2509.25756.

Zikai Zhou, Shitong Shao, Lichen Bai, Shufei Zhang, Zhiqiang Xu, Bo Han, and Zeke Xie. Golden Noise for Diffusion Models: A Learning Framework. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2025. arXiv:2411.09502.

A NOTATION

Symbol	Meaning
$\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, \rho_0, r, \gamma)$	environment MDP; $\boldsymbol{a} \in \mathcal{A} \subseteq \mathbb{R}^d$ is a chunk of H commands
$H, m, d = Hm$	chunk length, per-step action dimension, chunk and latent dimension
$\pi_\beta(\cdot \mid \boldsymbol{s}), \mathcal{D}$	frozen generative policy and the demonstrations it was cloned from
$\mathcal{W} = \mathbb{R}^d, \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$	latent-noise space and its prior
$f_\beta : \mathcal{S} \times \mathcal{W} \to \mathcal{A}$	deterministic decoder of π_β: $f_\beta(\boldsymbol{s}, \boldsymbol{w}) \sim \pi_\beta(\cdot \mid \boldsymbol{s})$ for $\boldsymbol{w} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$
$\mathcal{M}^{\mathcal{W}} = (\mathcal{S}, \mathcal{W}, P^{\mathcal{W}}, \rho_0, r^{\mathcal{W}}, \gamma)$	latent-action MDP
$\pi_\theta^{\mathcal{W}}, \pi_\theta$	latent policy and its action policy: $f_\beta(\boldsymbol{s}, \boldsymbol{w}) \sim \pi_\theta$ for $\boldsymbol{w} \sim \pi_\theta^{\mathcal{W}}$
$J^{\mathcal{W}}; Q_{\phi_j}^{\mathcal{W}}, Q_{\bar{\phi}_j}^{\mathcal{W}}$	return in $\mathcal{M}^{\mathcal{W}}$; twin critics and their Polyak targets
$\boldsymbol{\Delta}_\theta(\boldsymbol{s}), \boldsymbol{\epsilon}$	mean shift of the latent policy; base-noise draw
$\mathcal{T}_d(c)$	typical set of $\mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$: $\lvert\lVert\boldsymbol{w}\rVert^2 - d\rvert \le c\sqrt{2d}$
δ, η, λ	KL budget (nats), budget in shell widths ($\delta = \eta\sqrt{2d}$), its multiplier
$\alpha, \mathcal{B}, g$	SAC temperature (fixed per task); replay buffer; updates per transition

Table 2: Notation.

B IMPLEMENTATION DETAILS

B.1 NOISE PROBE (SECTION 3.1)

This section gives the protocol behind Section 3.1 in full: the frozen policies and the decoder, the noise families, the atypicality score, the open-loop probe behind Figure 1, and the closed-loop evaluations behind Table 1 and the paired-stream test. Every probe is deterministic given its seeds, and the scripts will be released with the code.

Policies and decoder. All probes use, without modification, the frozen diffusion policies that the experiments steer (Section B.2): an MLP ϵ-prediction network conditioned on the current low-dimensional state (a single observation, no history), trained by DDPM (Ho et al., 2020) on the robomimic multi-human demonstrations for can, lift, square and transport, and on the D4RL `hopper-medium-v2` dataset for hopper. Actions are normalized per dimension to $[-1, 1]$, and a policy outputs a chunk of H consecutive commands in $\mathbb{R}^m$, so $d = Hm$. Sampling is deterministic DDIM, started from the latent $\boldsymbol{w}$ that we supply in place of a fresh Gaussian; the denoised estimate is clamped to the action box after every step, as in DPPO, so a decoded chunk lies in $[-1, 1]^d$ and a coordinate *saturates* when it reaches the box. The map $\boldsymbol{w} \mapsto f_\beta(\boldsymbol{s}, \boldsymbol{w})$ is therefore a fixed deterministic function, and it is the same decoder that the latent policy of Section 3.3 drives. Table 3 lists the per-task settings; sampler, step limit and success criterion are those of each task's training configuration, and nothing is tuned per family.

Table 3: Per-task settings of the noise probe; T is the number of DDPM training steps.

Task	H	m	d	T	DDIM steps	episode length	chunks per episode	score
can	4	7	28	20	8	300	75	success
lift	4	7	28	20	8	300	75	success
square	4	7	28	100	8	400	100	success
transport	8	14	112	100	100	800	100	success
hopper	4	3	12	20	5	1000	250	return

Noise families. Write $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$ for a fresh prior draw, viewed as H step vectors $\boldsymbol{\epsilon}_1, \ldots, \boldsymbol{\epsilon}_H \in \mathbb{R}^m$, and $\boldsymbol{w}_t$ for the t-th step of the latent. Table 4 defines the families of Section 3.1, with $s \in \{0.5, 1.5, 2.5\}$, $b = 1.5$ and $\rho = 0.9$. The correlated-step recursion keeps unit marginal variance, so it and repeat step differ from the prior only in the dependence between steps; clip and corner share the box $[-b, b]^d$ but put their mass on its faces and at its vertices respectively. A family is always sampled from a fresh $\boldsymbol{\epsilon}$, so in closed loop every chunk receives an independent draw, and the mean family is a single point.

Table 4: Noise families; $\|\boldsymbol{w}\|^2/d$ values are medians at $d = 28$.

Family	Latent	Deviation from the prior
prior	$\boldsymbol{w} = \boldsymbol{\epsilon}$	none
scale s	$\boldsymbol{w} = s\,\boldsymbol{\epsilon}$	norm, $\|\boldsymbol{w}\|^2/d \approx s^2$
clip b	$w_i = \max(-b, \min(b, \epsilon_i))$	no tails; $\|\boldsymbol{w}\|^2/d \approx 0.77$
corner b	$w_i = b\,\mathrm{sign}(\epsilon_i)$	norm, $\|\boldsymbol{w}\|^2/d = b^2$; no tails
repeat step	$\boldsymbol{w}_t = \boldsymbol{\epsilon}_1$ for all t	step correlation 1
correlated steps	$\boldsymbol{w}_1 = \boldsymbol{\epsilon}_1$, $\boldsymbol{w}_t = \rho\,\boldsymbol{w}_{t-1} + \sqrt{1-\rho^2}\,\boldsymbol{\epsilon}_t$	step correlation ρ
mean	$\boldsymbol{w} = \mathbf{0}$	norm 0

Atypicality score. Seven statistics of a latent, adapted from the typicality regularizers that StressDream (Seo et al., 2026) applies to video-model latents (its spatial axes become the per-step action vector and its time axis the chunk axis), are compared with their distributions under the prior: $\|\boldsymbol{w}\|^2/d$; the mean and the standard deviation of its d coordinates; $\max_i |w_i|$; the step correlation, i.e. the mean absolute cosine similarity between the H centred step vectors (1 for repeat step, about 0.34 for a prior draw at $H = 4$); the spectral flatness along the chunk axis, i.e. the mean squared log-power of the discrete Fourier transform of $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_H$ with the constant bin dropped, which is near zero for white noise; and an isotropy statistic that permutes the coordinates at random (32 permutations), reshapes them into groups of k coordinates (k the largest divisor of d not exceeding 16), and sums the squared deviation of each group's variance from one, the squared group means and the squared cross-group correlations. The reference distribution of every statistic is estimated from 4096 prior draws at the same (H, m). A latent's tail rank for a statistic is $|2r - 1|$, with r its empirical mid-rank in the reference sample; its atypicality is the largest of its seven tail ranks, re-ranked against the distribution of that maximum on the reference sample itself (without this step a typical latent would score about $0.5^{1/7} \approx 0.9$ rather than 0.5). The score thus reads 0.5 for a median prior draw and 1 for a latent that is more extreme, in at least one statistic, than every reference draw, and the statistic attaining the maximum labels the kind of deviation.

Open-loop probe (Figure 1a–c). The probe state is the first state of the first can demonstration. For every stochastic family we draw 256 latents from one seeded generator and decode them at that state in a batch; the mean family gives the single chunk $\boldsymbol{a}_0 = f_\beta(\boldsymbol{s}, \mathbf{0})$. For each decoded chunk $\boldsymbol{a}$ we record the displacement $\|\boldsymbol{a} - \boldsymbol{a}_0\|$ (Figure 1b), the fraction of its d coordinates with $|a_i| \geq 0.995$ (saturation), and its distance to the demonstrations, i.e. the smallest Euclidean distance in $\mathbb{R}^d$ from $\boldsymbol{a}$ to any of the $61{,}856$ windows of H consecutive actions in the 300 can demonstrations (Figure 1a). The demonstrations' own distances, which give the reference lines of panel (a), are the same quantity for 512 random demonstration windows measured against the windows of all *other* demonstrations (median 0.22, 95th percentile 0.53). The chunk-level quantities quoted in Section 3.1, the median and 99th percentile of the distance and the chunk diversity (mean pairwise distance between decoded chunks), are computed over the 256 decodes of a family. For panel (c) we record the DDIM iterates $\boldsymbol{x}_0 = \boldsymbol{w}, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_8 = \boldsymbol{a}$ of every latent and plot $\|\boldsymbol{x}_i\|/\sqrt{d}$, its mean and 5–95% band per family; the band of panel (b) is the 5–95% range of $\|\boldsymbol{w}\|^2/d$ under the prior at $d = 28$. The tail figure of Section 3.1 is analytic: a prior coordinate exceeds 2.5 in magnitude with probability 0.0124, so $1 - (1 - 0.0124)^{28} = 0.30$ of prior chunks at $d = 28$ contain one, and a 300-step episode draws 75 chunks.

One chunk under the microscope (Figure 1d). The environment is reset with a fixed seed and its simulator state is saved. Five latents, one draw each from the mean, prior, $2.5\,\boldsymbol{w}$, corner ± 1.5 and repeat-step families, are decoded at the reset observation with their DDIM iterates recorded as above. For each latent the simulator is restored to the saved state and the $H = 4$ commands of its chunk are executed one environment step at a time; the end-effector position is read from the unnormalized observation after every step, and the displacement printed in the last row of the figure is the distance between the end-effector positions before the first and after the last command, in millimetres.

Closed-loop evaluation (Table 1). Episodes are run in synchronized rounds over a pool of environments. The initial-state seed of episode j is j itself, so every family meets the same initial states

($j = 0, \ldots, 17$, and $0, \ldots, 11$ on transport). At every chunk each environment receives an independent latent from the family, drawn from a generator seeded by the family and the round; the pool is decoded in one batch at the current observations, and each environment executes its H commands. A robomimic episode counts as a success if the task's sparse success reward is obtained at any chunk before the step limit of Table 3; hopper is scored by its undiscounted return, with the usual early termination when the hopper falls. The clip thresholds of Section 3.1 (± 1.0, ± 1.5, ± 2.0, ± 2.5) and the mild cooling $0.88\,\boldsymbol{w}$ were evaluated with the same protocol on the same 18 can episodes.

Paired-stream test. The test runs 48 can episodes (seeds 0–47) twice each. Both runs start from the same initial state and consume the identical sequence of prior draws $\boldsymbol{\epsilon}_1, \boldsymbol{\epsilon}_2, \ldots$, one per chunk, generated from a seed that depends only on the round; one arm feeds $\boldsymbol{\epsilon}_t$ to the decoder and the other feeds $\boldsymbol{\epsilon}_t$ clipped to ± 1.5, so any difference in outcome is attributable to the clipped coordinates alone. Of the 48 pairs, 20 succeed in both arms, 16 fail in both, 11 succeed only when clipped and 1 only when not; the exact McNemar test (McNemar, 1947) is the two-sided binomial test on the 12 discordant pairs, $p = 2\Pr[X \le 1 \mid n = 12,\ \frac{1}{2}] = 0.006$. We also record the first chunk at which the two arms' observations differ by more than 0.1 in Euclidean norm: the median pair forks at chunk 14 of 75, and the magnitude clipped away at the fork sits at the 73rd percentile of that episode's own clipped magnitudes, so forks follow an accumulation of small clipped tails rather than a single large event.

B.2 Experimental Setup (Section 4.1)

Table 5 lists the tasks and their frozen policies, and Table 6 the training and evaluation protocol together with the per-task settings of Typical DSRL. Unless stated otherwise, everything below applies to every method that learns in the latent space; the baselines' own settings, and where they depart from their releases, are given in Section B.3.

Table 5: Tasks and frozen generative policies. $\dim \mathcal{S}$ is the observation dimension, $T_{\max}$ the episode cap in low-level steps, T the number of DDPM training steps and K the number of DDIM or midpoint sampling steps; MH is the robomimic multi-human data.

Task	Suite	Data	Policy	K	H	m	d	$\dim \mathcal{S}$	$T_{\max}$	Score
lift	robomimic	MH	DDPM T=20, 3×512	8	4	7	28	19	300	success
can	robomimic	MH	DDPM T=20, 3×512	8	4	7	28	23	300	success
square	robomimic	MH	DDPM T=100, 3×1024	8	4	7	28	23	400	success
transport	robomimic	MH	DDPM T=100, 3×2048	100	8	14	112	59	800	success
door	Adroit	`expert-v1`	flow, 3×256	50	8	28	224	39	200	success
hammer	Adroit	`expert-v1`	flow, 3×256	50	8	26	208	46	200	success
pen	Adroit	`expert-v1`	flow, 3×256	50	8	24	192	45	100	success
relocate	Adroit	`expert-v1`	flow, 3×256	50	8	30	240	39	200	success
hopper	Gym	`medium-v2`	DDPM T=20, 3×512	5	4	3	12	11	1000	return
walker2d	Gym	`medium-v2`	DDPM T=20, 3×512	5	4	6	24	17	1000	return
halfcheetah	Gym	`medium-v2`	DDPM T=20, 3×512	5	4	6	24	17	1000	return

Table 6: Training and evaluation protocol per task, and the settings of Typical DSRL. B is the budget, N_0 the transitions of the frozen policy before training and E the evaluation interval, all in environment steps; n_{ev} is the number of evaluation episodes.

Task	B	N_0	E	n_{ev}	width	g	γ	α	η	δ
lift	50k	6k	12k	200	2048	7.5	0.99	0	0.5	3.74
can	500k	6k	80k	100	2048	5	0.99	0.05	0.5	3.74
square	500k	8k	40k	100	2048	5	0.999	0.05	0.5	3.74
transport	500k	80k	80k	100	128	5	0.99	0.05	0.5	7.48
door	500k	8k	40k	100	1024	2.5	0.99	0.05	0.5	10.58
hammer	500k	8k	40k	100	1024	2.5	0.99	0.1	0.5	10.20
pen	500k	8k	40k	100	1024	2.5	0.99	0.05	0.5	9.80
relocate	500k	8k	40k	100	1024	2.5	0.99	0.05	0.5	10.95
hopper	300k	8k	40k	100	128	2.5	0.99	0	0.5	2.45
walker2d	300k	8k	40k	100	2048	2.5	0.99	0.1	0.5	3.46
halfcheetah	300k	8k	40k	100	1024	2.5	0.99	0	0.5	3.46

Frozen policies. The robomimic and Gym policies are those used by DSRL. On lift, can and the three Gym tasks they are the checkpoints released with DPPO (Ren et al., 2025); on square and transport they are policies that the DSRL authors trained with DPPO's code and $T = 100$ rather than 20 denoising steps, which they report performs somewhat better. All are DDPM ϵ-prediction MLPs with residual connections, conditioned on the current low-dimensional state (a single observation, no history) and trained on the robomimic multi-human datasets (300 demonstrations per task) and on the D4RL `hopper`, `walker2d` and `halfcheetah` `medium-v2` datasets. Sampling is deterministic DDIM with the number of steps K of Table 5, started from the latent that the latent policy supplies, and the denoised estimate is clamped to the action box after every step, as in DPPO. The Adroit policies are flow-matching MLPs (three hidden layers of width 256, sinusoidal time embedding of dimension 64) that we trained with the rectified-flow objective, i.e. on $\boldsymbol{x}_t = (1-t)\boldsymbol{x}_0 + t\boldsymbol{a}$ with $\boldsymbol{x}_0 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$, $t \sim \mathcal{U}(0,1)$ and target velocity $\boldsymbol{a} - \boldsymbol{x}_0$, on the D4RL `door`, `hammer`, `pen` and `relocate` `expert-v1` datasets: 300k steps of batch 512 with Adam (Kingma & Ba, 2015), learning rate 10^{-3} cosine-annealed to 10^{-5}, a 2% held-out split, and the observation and action normalization released with LP-DS, about 40 minutes on CPU per task. Sampling integrates the velocity field from $t = 0$ to 1 with the midpoint rule in $K = 50$ steps, so, as with DDIM, the latent is the initial iterate $\boldsymbol{x}_0$.

Environment interface. An environment step executes one chunk of H low-level commands at the simulator's control rate and returns the observation after the last command; its reward is the sum of the low-level rewards over the chunk, shifted by -1 per step on robomimic so that a chunk without success earns -1. Observations are the low-dimensional states the frozen policies were trained on (end-effector pose, gripper state and object poses on robomimic; hand and object state on Adroit; the MuJoCo state on Gym), normalized with the pretraining statistics, and actions are normalized to $[-1, 1]$ per dimension. Episodes are capped at $T_{\max}$ low-level steps (Table 5). A robomimic episode succeeds if the task's sparse success reward is obtained at any step before the cap; an Adroit episode succeeds if the environment's goal flag is raised at any step, and on pen the episode ends when the environment signals termination; a Gym episode is scored by its undiscounted return, with the standard early termination when hopper or walker2d falls. Training uses 4 parallel environments and evaluation 25 (20 on Adroit), seeded apart from the training environments.

Training loop. Typical DSRL is soft actor–critic in the latent-action MDP $\mathcal{M}^{\mathcal{W}}$, built on Stable-Baselines3 (Raffin et al., 2021), with the environment wrapper decoding every latent through f_β before it reaches the simulator. Actor and critics are MLPs with three hidden layers of the width given in Table 6 and layer normalization (Ba et al., 2016), trained with Adam at learning rate 3×10^{-4}, batch size 256, discount γ, Polyak averaging with coefficient 0.005, two critics with the clipped double-Q target, and a replay buffer of 2M transitions. Before any update the buffer is seeded with N_0 transitions collected by the frozen policy itself, i.e. with prior latents; these do not count towards B. Training then interleaves environment steps with g gradient steps per transition, the actor updated at every gradient step. The actor is a residual Gaussian policy over latents, $\boldsymbol{w} = \boldsymbol{\Delta}_\theta(\boldsymbol{s}) + \boldsymbol{\epsilon}$ with $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$: the mean head is zero-initialized, so that the step-0 policy is exactly the prior and the first evaluation point of every curve is the frozen policy, and the log-standard-deviation head is frozen at 0, so that $\sigma \equiv 1$ throughout. With σ fixed the policy entropy is constant, so the temperature α acts only through the soft TD target; it is held at the value of Table 6 rather than auto-tuned. The trust region of Equation (2) enters the actor loss as $\lambda\, \overline{D_{\mathrm{KL}}}$, where $\overline{D_{\mathrm{KL}}}$ is $\frac{1}{2}\|\boldsymbol{\Delta}_\theta(\boldsymbol{s})\|^2$ averaged over the replay batch, and λ is a log-parameterized Lagrange multiplier initialized at 1, updated after every actor step by dual ascent on $\overline{D_{\mathrm{KL}}} - \delta$ (Adam, step size 3×10^{-4}) and clamped to $[10^{-6}, 10^2]$. Latents are clamped to $[-6, 6]^d$ at the decoder as a numerical safeguard; we count the clamp as part of f_β, so the latent policy stays exactly Gaussian, and the clipped fraction is ≈ 0 in every run.

Selecting α, η and the budgets. α is fixed per task at the value of Table 6: 0 on lift, hopper and halfcheetah, 0.1 on walker2d and hammer, and 0.05 on the other six tasks. Square is the only task on which Typical DSRL was trained at more than one value, $\alpha \in \{0.02, 0.05, 0.08\}$ (Section C.5); every other task was trained at its value alone. The trust region uses $\eta = 0.5$ on every task. The budgets B were fixed from pilot runs of unmodified DSRL-SAC before any method comparison: 500k environment steps on robomimic and Adroit, 300k on Gym, where the dense-reward tasks plateau earlier, and 50k on lift, which the frozen policy already solves most of the time.

Evaluation and aggregation. Evaluation runs at the cadence E of Table 6 and once before training, giving between 5 (lift) and 13 (square and Adroit) evaluation points per run; the plateau averages the last quarter of them, rounded to the nearest point (1 on lift, 2 on can, transport and Gym, 3 elsewhere). Each evaluation runs n_{ev} episodes in both modes, stochastic with $\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{\Delta}_\theta(\boldsymbol{s}), \boldsymbol{I}_d)$ and deterministic with $\boldsymbol{w} = \boldsymbol{\Delta}_\theta(\boldsymbol{s})$; at step 0 these are the frozen policy and its mean decode. Success rates are fractions of evaluation episodes and returns are undiscounted sums. Every method is reported over two seeds per task, except on the two single-seed transport cells of Section B.3, and tables give their mean and population standard deviation. Wall-clock is the runtime recorded by the experiment tracker from process start to the last logged step, evaluation included. Every run trained on an AWS `g6.48xlarge` instance (NVIDIA L4, 24 GB) with up to eight runs sharing the machine, so the times are shared-machine times and compare methods at the level of hours rather than minutes.

B.3 Baselines (Section 4.1)

Every baseline improves the frozen policy of Table 5 on its task under the protocol of Table 6: the same budget B, the same N_0 transitions of the frozen policy before training, the same evaluation cadence and episode counts, four training environments, chunk-level transitions with the chunk-summed reward of the environment interface, time limits treated as terminal, a replay buffer of 2M transitions, and no demonstrations in it. The four latent-space methods run in the Stable-Baselines3 code released with DSRL, into which the LP-DS release was merged; DICE-RL and OGPO are our ports of their releases into the same training loop, so that a run of any method differs from a run of Typical DSRL in the algorithm alone. Both were released with their own data and reward conventions and for a subset of our tasks, so running them here required choices, which their paragraphs below state. Every baseline was run with two seeds per task; the transport runs of DSRL-NA and DICE-RL are single seeds, and OGPO is not run on transport (below). DSRL-SAC and DSRL-NA have a mean policy and are reported in both evaluation modes of Section 4.1, the deterministic one decoding the mean of their actor. LP-DS, LP-DS-SAC and DICE-RL draw fresh prior noise at every call and have no mean policy, so they are reported in their native stochastic protocol only. OGPO has a deterministic sampler, but we report it by its best-of-8 acting rule, the test-time search it acts with in training (see the OGPO paragraph below). Table 7 lists the per-task settings of the latent-space baselines that are not those of Table 6.

Table 7: Settings of the latent-space baselines that are not in Table 6. b and b_W are box half-widths, c the constraint level and g_{d} the distillation steps per transition.

	DSRL-SAC	LP-DS and LP-DS-SAC				LP-DS / DSRL-NA
Task	b	b_W	c	width	g	g_{d}
lift	1.5	1.5	0.05	2048	7.5	2.5
can	1.5	1.5	0.05	2048	7.5	2.5
square	1.5	1.5	0.05	2048	7.5	2.5
transport	1	1	0.35	1024	5	2.5
door	2.5	2.5	0.35	256	2.5	2.5 / 1.25
hammer	2.5	2.5	0.1	256	2.5	2.5 / 1.25
pen	2.5	2.5	0.35	256	2.5	2.5 / 1.25
relocate	2.5	2.5	0.35	256	2.5	2.5 / 1.25
hopper	1.5	1.5	0.05	2048	3.75	1.25
walker2d	2.5	1.5	0.05	2048	3.75	1.25
halfcheetah	1.5	1.5	0.05	2048	3.75	1.25

DSRL-SAC. Soft actor–critic in $\mathcal{M}^{\mathcal{W}}$, as released: a squashed Gaussian actor $\boldsymbol{w} = b\tanh\big(\boldsymbol{\mu}_\theta(\boldsymbol{s}) + \boldsymbol{\sigma}_\theta(\boldsymbol{s}) \odot \boldsymbol{\epsilon}\big)$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$, over the box $[-b, b]^d$; two latent critics trained by temporal differences on the stored latent with the clipped double-Q target; and the temperature α tuned online against a target entropy of 0 (Haarnoja et al., 2018b). The buffer is seeded with prior draws clipped to the box. The half-width b is DSRL's "action magnitude", taken from the DSRL paper for robomimic and Gym and from the Adroit settings of the LP-DS paper for Adroit (Table 7); all other settings are those of Table 6, which Typical DSRL inherits from this method, and its deterministic mode decodes $b\tanh(\boldsymbol{\mu}_\theta(\boldsymbol{s}))$. DSRL used this variant only on transport; we run it on every

task because it is the algorithmic base of Typical DSRL, which differs from it in the parameterization of the latent policy, the constraint and the fixed α (Section 3.3).

DSRL-NA. The noise-aliased instantiation: the same actor and entropy term as DSRL-SAC, but the buffer stores the decoded action, an action critic $Q(\boldsymbol{s}, \boldsymbol{a})$ is trained by temporal differences on it (its target decodes the actor's next latent), and the latent critic $Q^{\mathcal{W}}$ that the actor ascends is distilled from the action critic by regressing $Q^{\mathcal{W}}(\boldsymbol{s}, \boldsymbol{w})$ onto $Q(\boldsymbol{s}, f_\beta(\boldsymbol{s}, \boldsymbol{w}))$ on fresh prior draws $\boldsymbol{w} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$, 10 steps per update ($g_{\mathrm{d}}$ in Table 7). This is DSRL's recommended variant, used in its paper on every task but transport, and the critic design that LP-DS adopts. Every distillation step and every target decodes a batch through the frozen policy, which is what its wall-clock pays for. In the code we ran, the noise-aliased actor's box is the environment's action bounds tiled over the chunk, i.e. $b = 1$ on every task, rather than the action magnitude of the paper; its deterministic mode decodes the actor's mean. We run it on all eleven tasks, including transport, where the paper used DSRL-SAC.

LP-DS. Lagrangian perturbation diffusion steering keeps the two critics of DSRL-NA and replaces the actor by a deterministic residual on the base noise: a network $\boldsymbol{\Delta}_\theta(\boldsymbol{s}, \boldsymbol{\epsilon})$ with a zero-initialized output layer, so that training starts at the frozen policy, produces $\boldsymbol{w} = \boldsymbol{\epsilon} + \boldsymbol{\Delta}_\theta(\boldsymbol{s}, \boldsymbol{\epsilon})$ from $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$. The latent is kept in the box $[-b_W, b_W]^d$ by the release's projection: $\boldsymbol{\epsilon}$ is clipped to the box first, and the residual of each sample is then scaled by the largest factor in $[0, 1]$ that keeps all of its coordinates inside. The actor ascends $Q^{\mathcal{W}}(\boldsymbol{s}, \boldsymbol{w})$ subject to $\mathbb{E}[\|\boldsymbol{\Delta}_\theta(\boldsymbol{s}, \boldsymbol{\epsilon})\|^2]/d \le c$, the paper's small-perturbation approximation of the divergence to the prior, normalized per coordinate as in the release, through a Lagrange multiplier $\lambda = e^{\ell}$: ℓ is updated by Adam on an exponential moving average of the violation (coefficient 0.9, violation clipped to ± 0.1) and clamped to $[-10, 10]$. The actor is deterministic given $\boldsymbol{\epsilon}$, so the prior's noise is its only exploration and there is no entropy term. The action critic's target decodes a fresh latent from a slow Polyak copy of the actor (coefficient 0.995), and the latent critic is distilled on clipped prior draws. Settings follow the LP-DS release where it covers a task (lift, transport and Adroit); the remaining tasks take the release's lift settings, so $c = 0.05$ on can, square and Gym, where the paper's own table lists 0.35 to 0.5 (Table 7). On transport the paper's configuration decodes 30 batches through the 100-step sampler per update, and a seed would take about ten days on our hardware; the two runs were stopped at 9% of the budget, and only their initial evaluation, the frozen policy, is reported. Evaluation draws fresh $\boldsymbol{\epsilon}$ at every call.

LP-DS-SAC. Our ablation of LP-DS, constructed so that LP-DS/LP-DS-SAC isolates the aliased critic as DSRL-NA/DSRL-SAC does. It keeps the residual actor, the projection, the constraint and the multiplier unchanged and removes the action critic and the distillation: $Q^{\mathcal{W}}$ is trained by temporal differences on the executed latent, as in DSRL-SAC, with the target latent produced by the slow actor copy from fresh prior noise. The buffer therefore stores the projected latent rather than the decoded action, and training never calls the decoder, so an update costs what DSRL-SAC's does. Its configuration is LP-DS's on every task with the distillation removed (Table 7), and it is evaluated as LP-DS is.

DICE-RL. Distribution-contractive RL fine-tuning leaves the latent at the prior and learns an additive residual in action space, $\boldsymbol{a} = f_\beta(\boldsymbol{s}, \boldsymbol{w}) + \boldsymbol{r}_\theta(\boldsymbol{s}, \boldsymbol{w})$ with $\boldsymbol{w} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$; we port the authors' release. An ensemble of ten action critics $Q_i(\boldsymbol{s}, \boldsymbol{a})$ is trained by temporal differences on the stored chunk towards an n-step target ($n = 3$) whose bootstrap averages $\min_i Q_i$ over $K' = 4$ next-state decodes. The actor minimizes $-\min_i Q_i(\boldsymbol{s}, \boldsymbol{a}_k)$ averaged over $K = 16$ fresh latents per state and normalized by the batch mean of $|Q|$, plus $\beta\,\overline{\|\boldsymbol{r}_\theta\|^2}$ with $\beta = 100$. Acting and evaluation decode 16 candidates and execute the one with the largest $\min_i Q_i$, after a warm-up of 16k environment steps with a single sample. Networks have three layers of 1024 with GELU and layer normalization; Adam at 10^{-4} with weight decay 10^{-5} and cosine warm restarts, gradient norm clipped to 1, ten gradient steps per vectorized environment step with the actor and the targets updated on every second one, and Polyak coefficient 0.01. As in the release, each executed chunk enters the buffer as a transition starting at each of its H sub-steps, and the residual is not zero-initialized, so DICE-RL's first evaluation point is not exactly the frozen policy. We depart from the release in five places. (i) The release mixes the demonstrations into every batch at a share decaying from 0.9 to 0.1; we train online-only, so DICE-RL sees the data every other method sees. (ii) We use the reward, termination

and discount of our protocol (the chunk-summed shifted reward, no termination at success, the task's γ) in place of the release's sparse $+1$ with termination after success. (iii) The n-step return sums non-overlapping successive chunks, the rule of the release's documentation and demonstration path; its online sampler sums overlapping sub-step-shifted chunks, which is only correct when the reward is nonzero once per episode. (iv) The release covers robomimic only; lift, Adroit and Gym take its can, square and transport values, which coincide, and its Q-based BC filter is off as in every released configuration. (v) On transport, where every gradient step decodes $4096 + 1024$ latents through 100 DDIM steps, the single seed uses $K = 4$ and $K' = 1$ (106 hours, against about 420 at the released values), so that cell is not strictly comparable to the others. DICE-RL has no deterministic mode; its evaluation is the best-of-16 rule on fresh latents.

OGPO. Off-policy generative policy optimization is the one baseline that changes the generator's weights: it treats the sampling chain as a stochastic policy and updates the denoiser by a clipped policy gradient (Schulman et al., 2017) with group-relative advantages. We port the authors' release onto the frozen policies of Table 5. Per gradient step, $G = 32$ stochastic chains are sampled for each of 256 replay states from an exponential-moving-average copy of the generator; a ten-head action critic, itself trained by temporal differences with one chain sample in the bootstrap and mean aggregation, scores their end points, and a chain's advantage is its score minus the group mean, computed per head and combined by the release's conservative rule (the minimum when every head agrees on the sign, zero otherwise) on robomimic and Gym and by the plain group mean on Adroit. The generator takes one Adam step on the clipped objective (clip range 0.01 on the ratio of the chain's log-density under the current and the averaged weights), plus its own behaviour-cloning loss on a batch from a *success buffer* of transitions from successful episodes (coefficient 1; the replay batch until 256 such transitions exist); on robomimic and Gym the critic takes a second step on the success batch. Acting draws 8 chains from the averaged generator and executes the one preferred by two target-critic heads drawn at random. Learning rates are 4.5×10^{-5}, the generator's under the release's repeating warm-up cosine schedule; Polyak coefficients are 0.05 for the critic and 0.005 for the averaged generator (0.05 on Adroit); the critic has four layers of 512 (five on transport). We depart from the release in five places. (i) The release trains its own flow-matching prior by behaviour cloning, stopped early at 45% success, and fine-tunes that; we fine-tune each task's frozen policy instead. On the seven diffusion tasks the stochastic chain is therefore DPPO's stochastic DDIM sampler (full DDIM noise, standard-deviation floor 0.1) over all K steps, except the last 10 of 100 on transport, DPPO's own choice; on Adroit it is the release's tapered-noise flow SDE ($\sigma = 0.05$) on our flow network with 10 Euler steps, where the release uses 3 on a prior trained for 3. OGPO's step-0 deterministic policy is the frozen policy on the diffusion tasks and differs from it on Adroit only by the sampler (10 Euler against 50 midpoint steps). (ii) Chain log-densities are normalized per fine-tuned step and coordinate, DPPO's convention, rather than by $(K+1)HA$ with the weight-independent initial term as in the release; the two differ by a constant factor and the clip range is kept. (iii) The reward, termination and data conventions of our protocol replace the release's -1 per step, $+5$ then terminate, discounting within the chunk, and behaviour-cloning warm start. (iv) Success for the buffer is our rule: a chunk reward above the offset on robomimic and the goal flag on Adroit; Gym has no success signal, so its BC term always uses the replay batch, the release's fallback. (v) The release takes one gradient step per low-level environment step; we take one per chunk step, i.e. $1/H$ as many, because each step samples $G \times 256$ chains and back-propagates their log-densities through the generator (on square, about 42 against 165 hours per seed); interaction and evaluation are unchanged. We omit transport, where a single seed is estimated to take 178 hours even at $G = 8$. We evaluate OGPO both with the deterministic sampler of the current weights and with its best-of-8 acting rule, and report the latter: it is the policy OGPO acts with, and it spends eight chains and a critic evaluation per chunk to select among them. The deterministic sampler is lower on seven of the ten tasks with finished runs, most on hammer (0.15 against 0.49), can (0.77 against 0.95) and relocate (0.37 against 0.49), and higher by at most 0.03 or 9 return on the other three (square, door, halfcheetah).

C Additional Results

C.1 Main results: plateau tables and training cost (Figure 2)

Table 8 lists the plateaus plotted in Figure 2, each method under the evaluation protocol it deploys with. Table 9 repeats it with every method under stochastic evaluation: Typical DSRL, DSRL-SAC and DSRL-NA sample $\boldsymbol{w} \sim \pi_\theta^{\mathcal{W}}(\cdot \mid \boldsymbol{s})$ as during training, and the LP-DS, LP-DS-SAC, DICE-RL and OGPO columns are those of Table 8, which already reports these four stochastically. Table 10 gives the training wall-clock of the same runs, measured as in Section B.2: from process start to the last logged step, evaluation included, on machines shared by up to eight runs, so that differences of minutes are not meaningful. Transport LP-DS carries no time because neither seed ran to completion.

Table 8: Plateau (success rate; return on Gym), mean ± standard deviation over seeds (one seed for DSRL-NA and DICE-RL on transport). Each method is evaluated as it is deployed: stochastically if marked †, deterministically otherwise. Bold: the best mean and any mean within its standard deviation. ‡: initial evaluation only.

Task	Typical DSRL	DSRL-SAC	DSRL-NA	LP-DS†	LP-DS-SAC†	DICE-RL†	OGPO†
lift	0.88 ± 0.07	0.90 ± 0.01	0.96 ± 0.02	$\mathbf{0.98 \pm 0.01}$	0.80 ± 0.01	0.94 ± 0.02	0.95 ± 0.04
can	$\mathbf{1.00 \pm 0.00}$	0.85 ± 0.04	0.98 ± 0.01	0.97 ± 0.00	0.95 ± 0.02	$\mathbf{1.00 \pm 0.00}$	0.95 ± 0.03
square	0.92 ± 0.03	0.46 ± 0.00	0.92 ± 0.01	0.86 ± 0.02	0.76 ± 0.05	$\mathbf{0.99 \pm 0.00}$	0.00 ± 0.00
transport	$\mathbf{0.65 \pm 0.05}$	0.44 ± 0.14	0.00 ± 0.00	$0.33^{\ddagger}$	0.31 ± 0.01	0.34 ± 0.00	—
door	$\mathbf{0.99 \pm 0.00}$	0.24 ± 0.17	0.96 ± 0.02	0.95 ± 0.00	0.94 ± 0.00	$\mathbf{0.99 \pm 0.01}$	0.46 ± 0.12
hammer	0.97 ± 0.01	0.19 ± 0.07	0.79 ± 0.05	0.95 ± 0.01	0.85 ± 0.01	$\mathbf{1.00 \pm 0.00}$	0.49 ± 0.04
pen	$\mathbf{0.99 \pm 0.00}$	0.58 ± 0.00	0.93 ± 0.02	0.92 ± 0.02	0.90 ± 0.01	0.95 ± 0.01	0.82 ± 0.09
relocate	$\mathbf{1.00 \pm 0.00}$	0.03 ± 0.03	0.95 ± 0.01	0.94 ± 0.01	0.85 ± 0.01	$\mathbf{1.00 \pm 0.00}$	0.49 ± 0.22
hopper	$\mathbf{3175 \pm 25}$	3102 ± 8	3038 ± 16	2874 ± 10	2849 ± 6	3146 ± 12	$\mathbf{3161 \pm 2}$
walker2d	$\mathbf{3925 \pm 17}$	3844 ± 40	3812 ± 42	3770 ± 3	3683 ± 3	3804 ± 38	$\mathbf{3912 \pm 66}$
halfcheetah	$\mathbf{5015 \pm 2}$	4606 ± 44	4820 ± 8	4568 ± 22	4528 ± 2	4646 ± 12	4600 ± 42

Table 9: Plateau of every method under stochastic evaluation; entries, bold and ‡ as in Table 8.

Task	Typical DSRL	DSRL-SAC	DSRL-NA	LP-DS	LP-DS-SAC	DICE-RL	OGPO
lift	0.77 ± 0.15	0.88 ± 0.00	0.95 ± 0.01	$\mathbf{0.98 \pm 0.01}$	0.80 ± 0.01	0.94 ± 0.02	0.95 ± 0.04
can	0.97 ± 0.02	0.86 ± 0.02	0.99 ± 0.00	0.97 ± 0.00	0.95 ± 0.02	$\mathbf{1.00 \pm 0.00}$	0.95 ± 0.03
square	0.90 ± 0.03	0.55 ± 0.00	0.93 ± 0.01	0.86 ± 0.02	0.76 ± 0.05	$\mathbf{0.99 \pm 0.00}$	0.00 ± 0.00
transport	$\mathbf{0.42 \pm 0.05}$	$\mathbf{0.40 \pm 0.12}$	0.00 ± 0.00	$0.33^{\ddagger}$	0.31 ± 0.01	0.34 ± 0.00	—
door	0.82 ± 0.02	0.11 ± 0.03	0.96 ± 0.01	0.95 ± 0.00	0.94 ± 0.00	$\mathbf{0.99 \pm 0.01}$	0.46 ± 0.12
hammer	0.78 ± 0.02	0.11 ± 0.01	0.77 ± 0.01	0.95 ± 0.01	0.85 ± 0.01	$\mathbf{1.00 \pm 0.00}$	0.49 ± 0.04
pen	$\mathbf{0.94 \pm 0.00}$	0.62 ± 0.04	$\mathbf{0.94 \pm 0.00}$	0.92 ± 0.02	0.90 ± 0.01	$\mathbf{0.95 \pm 0.01}$	0.82 ± 0.09
relocate	0.81 ± 0.01	0.03 ± 0.02	0.93 ± 0.03	0.94 ± 0.01	0.85 ± 0.01	$\mathbf{1.00 \pm 0.00}$	0.49 ± 0.22
hopper	3137 ± 18	3068 ± 44	3022 ± 6	2874 ± 10	2849 ± 6	3146 ± 12	$\mathbf{3161 \pm 2}$
walker2d	3776 ± 34	3573 ± 92	3816 ± 38	3770 ± 3	3683 ± 3	3804 ± 38	$\mathbf{3912 \pm 66}$
halfcheetah	4508 ± 0	4440 ± 70	$\mathbf{4778 \pm 12}$	4568 ± 22	4528 ± 2	4646 ± 12	4600 ± 42

Table 10: Training wall-clock in hours of the runs of Table 8.

Task	Typical DSRL	DSRL-SAC	DSRL-NA	LP-DS	LP-DS-SAC	DICE-RL	OGPO
lift	2.1 ± 0.0	6.0 ± 0.1	10.3 ± 1.0	6.0 ± 0.0	1.9 ± 0.0	14.2 ± 0.0	6.5 ± 0.1
can	17.2 ± 0.0	41.8 ± 0.5	34.0 ± 0.1	45.2 ± 0.0	22.6 ± 0.0	74.8 ± 0.6	46.8 ± 0.1
square	17.8 ± 0.0	16.7 ± 0.0	33.8 ± 0.3	46.9 ± 0.6	23.0 ± 0.0	84.2 ± 0.7	64.4 ± 0.2
transport	38.2 ± 0.0	39.6 ± 0.6	130.6 ± 0.0	—	38.4 ± 0.1	106.1 ± 0.0	—
door	5.8 ± 0.1	5.7 ± 0.0	26.6 ± 0.6	34.1 ± 0.4	5.5 ± 0.0	80.2 ± 0.8	31.4 ± 0.4
hammer	5.9 ± 0.0	5.7 ± 0.1	26.1 ± 0.0	33.4 ± 0.1	5.7 ± 0.1	85.3 ± 0.6	31.4 ± 0.4
pen	5.8 ± 0.1	5.5 ± 0.0	26.9 ± 0.1	34.5 ± 0.0	5.4 ± 0.0	88.3 ± 0.3	37.7 ± 0.0
relocate	6.0 ± 0.0	5.7 ± 0.1	26.1 ± 0.3	33.4 ± 0.0	5.5 ± 0.1	70.2 ± 1.0	37.7 ± 0.0
hopper	2.6 ± 0.0	2.4 ± 0.0	10.8 ± 0.1	18.8 ± 0.3	6.1 ± 0.0	62.5 ± 0.1	25.8 ± 0.1
walker2d	4.7 ± 0.0	4.5 ± 0.0	12.0 ± 0.0	17.6 ± 0.1	6.2 ± 0.0	62.4 ± 0.1	25.8 ± 0.1
halfcheetah	2.7 ± 0.1	2.5 ± 0.0	10.8 ± 0.2	18.2 ± 0.1	6.2 ± 0.0	40.8 ± 0.7	21.8 ± 0.1

C.2 Maximum-likelihood and sampled latent action (Section 4.2)

Figure 5 plots the Typical DSRL columns of Table 8 and Table 9: the plateau of the same checkpoints under the maximum-likelihood latent action and under the sampled latent action.



Figure 5: Plateau of Typical DSRL; labels give the gain of the maximum-likelihood latent action.

C.3 Checkpoint diagnostics (Figure 3)

Evaluation. Each final checkpoint is rebuilt from its own configuration with every parameter restored exactly, including the temperature DSRL's online tuning had reached, and evaluated with the protocol of Section B.2: 100 episodes per evaluation mode as 20 environments $\times$ 5 rounds, the environment seed of episode (round, environment) fixed, so that every checkpoint and both references start from the same initial states; success is scored as in training and an episode ends at its first termination. The references are the frozen policy with a fresh prior draw at every chunk and with the zero latent. On one T4 the wave took 7 minutes.

Per-step record. For the executed latent $\boldsymbol{w}$, in raw units (the critic of a squashed actor receives $\boldsymbol{w}/b$, as in training): the energy $\|\boldsymbol{w}\|^2/d$, the largest coordinate, the coordinate standard deviation, and the atypicality score of Section B.1 against 4096 prior draws at the same d. The divergence of the latent policy from the prior is $\frac{1}{2}\|\boldsymbol{\Delta}_\theta(\boldsymbol{s})\|^2$ for Typical DSRL and, for the squashed actors, whose density in latent space includes the tanh Jacobian, the mean of $\log \pi_\theta^{\mathcal{V}}(\boldsymbol{w}_k \mid \boldsymbol{s}) - \log \mathcal{N}(\boldsymbol{w}_k; \mathbf{0}, \boldsymbol{I}_d)$ over 16 samples $\boldsymbol{w}_k$; σ_{eff} is the standard deviation of those 16 samples per coordinate, averaged over coordinates. Table 11 gives the means on door. The displacement of the decoded action is the RMS per-coordinate difference between $f_\beta(\boldsymbol{s}, \boldsymbol{w})$ and the decode of the zero latent ($\|\boldsymbol{a} - \boldsymbol{a_0}\|$) or of a paired prior draw ($\|\boldsymbol{a} - \boldsymbol{a_\epsilon}\|$). The critic's value of prior draws, $\bar{Q}(\boldsymbol{\epsilon})$, is the minimum over the two critics at 16 paired prior draws at the visited state, and $\bar{G} = \sum_{k \geq 0} \gamma^k r_{t+k}$ is the discounted return-to-go over the remainder of the recorded episode; $\bar{G}$ is truncated at the time limit, so on door (25 chunks, up to 160 reward per chunk, a largest possible value of $160/(1-\gamma) \approx 16{,}000$) a critic that bootstraps through the time limit exceeds it for every good policy, and the ratio $\bar{Q}(\boldsymbol{\epsilon})/\bar{G}$ is comparable across critics rather than equal to one. Under deterministic evaluation it is 3.1 to 3.6 for Typical DSRL and DSRL-NA and 40 for DSRL-SAC, whose critic values prior draws at 13,500. For DSRL-NA the action critic is also evaluated at the executed action; its gap to the distilled latent critic is 180 to 270, 3 to 4% of the value.

Table 11: Final checkpoints on door, mean over seeds (one for DSRL-SAC): box half-width b (the clamp for Typical DSRL), statistics of a stochastic latent, and divergence from the prior in nats.

Policy	b	$\|\boldsymbol{w}\|^2/d$	$\max_i \|w_i\|$	atyp.	KL	success
$\boldsymbol{w} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$	—	1.00	3.01	0.50	0.0	0.69
Typical DSRL	6.0	1.08	3.13	0.59	9.9	0.99
DSRL-SAC	2.5	2.50	2.46	1.00	306.7	0.17
DSRL-NA	1.0	0.41	0.99	1.00	243.7	1.00

Fixed-state probe. A pool of about 2,000 visited states, drawn uniformly from the recorded states of every checkpoint, mode and reference, on which every actor and critic is evaluated at the same states. Figure 3b reports the latent critic along the ray $r\boldsymbol{u}$ for $r \in \{0, \frac{1}{4}, \ldots, 3\}\sqrt{d}$, with $\boldsymbol{u}$ the

direction of the actor's own mean latent at that state and, separately, one fixed prior direction, as $Q^{\mathcal{W}}(r\boldsymbol{u}) - Q^{\mathcal{W}}(\mathbf{0})$ averaged over the pool. The cross-critic comparison scores, for every ordered pair of checkpoints, the mean over the pool of the minimum over the two critics of one checkpoint at 16 samples of the other's actor, with prior draws as a reference. Every critic values its own actor's latents above prior draws, but only DSRL-SAC's values DSRL-SAC's latents above them (14,460 against 13,890); the critics of Typical DSRL and DSRL-NA value them 1 to 3% below prior draws.

C.4 How the latent maps to the action (Figures 4, 6 and 7)

Checkpoints and states. The final checkpoints of Typical DSRL on lift, door and hammer (two per task) and on pen (one), with the recorded episodes of Section C.3, which we ran on hammer and pen for this analysis. States come from the fixed-state pool of Section C.3, which mixes the visited states of every checkpoint and reference of a task: 256 of them for the decoder statistics, and the whole pool for the statistics of $\boldsymbol{\Delta}_\theta$.

Decoder (Figure 4a,b). At each state we draw 64 latents $\boldsymbol{\epsilon}_k \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_d)$, shared by the checkpoints of a task, and decode $\boldsymbol{\Delta}_\theta(\boldsymbol{s})$ and $\boldsymbol{\Delta}_\theta(\boldsymbol{s}) + \boldsymbol{\epsilon}_k$ with the deterministic samplers of Section B.2: DDIM with 8 steps on lift, whose denoised iterate is clipped to ± 1, and a 50-step midpoint solver on Adroit, whose actions the environment clips. Actions are in the environment's normalized units. Panel (a) counts, over coordinates and draws, how often the sign of $f_\beta(\boldsymbol{s}, \boldsymbol{\Delta}_\theta(\boldsymbol{s}) + \boldsymbol{\epsilon}_k) - f_\beta(\boldsymbol{s}, \boldsymbol{\Delta}_\theta(\boldsymbol{s}))$ matches that of $\boldsymbol{\epsilon}_k$, ties at a clip excluded, and at right how often the sign of the action matches that of the latent. Panel (b) decodes $\boldsymbol{\Delta}_\theta(\boldsymbol{s}) + r\boldsymbol{u}$ along 8 random unit directions $\boldsymbol{u}$ per state. The Jacobian $\boldsymbol{J} = \partial f_\beta / \partial \boldsymbol{w}$ at $\boldsymbol{\Delta}_\theta(\boldsymbol{s})$ is taken by central finite differences with step 0.05 at 24 states per checkpoint. The explained variance is that of the linear prediction $f_\beta(\boldsymbol{s}, \boldsymbol{\Delta}_\theta(\boldsymbol{s})) + \boldsymbol{J}\boldsymbol{\epsilon}_k$ over the 64 draws, median over states: 0.99 on Adroit, 0.40 and 0.24 for the two lift checkpoints. On Adroit the mean diagonal entry of $\boldsymbol{J}$ is 0.25 to 0.29, the mean off-diagonal magnitude 0.001 to 0.002, and 37 to 52% of the magnitude of $\boldsymbol{J}$ lies on its diagonal, against 0.5% for an unstructured matrix; its participation ratio is 164 to 208 of $d = 192$ to 224, and it changes by 2 to 5% between $\boldsymbol{w} = \mathbf{0}$ and $\boldsymbol{w} = \boldsymbol{\Delta}_\theta(\boldsymbol{s})$. On lift, 5 of the 28 singular values carry 90% of $\|\boldsymbol{J}\|_F^2$. The sign of a noise coordinate matches that of the action change it causes for 94 to 96% of coordinates on Adroit and 76% on lift, and the sign of the latent matches that of the action for about 60% on every task. In Figure 6 the learned shift, the spread and the distance to the mean are root-mean-square norms over states, divided by $\sqrt{d}$; the tick is the error of a 64-sample mean. On Adroit a sampled action scatters by 0.27 to 0.32 per coordinate around $f_\beta(\boldsymbol{s}, \boldsymbol{\Delta}_\theta(\boldsymbol{s}))$ and the learned shift moves it by 0.08 to 0.10; on lift the two are 0.21 and 0.26. On lift, 12 to 14% of sampled chunks command the gripper with the opposite sign to $f_\beta(\boldsymbol{s}, \boldsymbol{\Delta}_\theta(\boldsymbol{s}))$ at one step or more. The distance to the demonstrations is the root-mean-square per coordinate to the nearest window of H consecutive demonstration actions, among 200,000 random windows of the D4RL data on Adroit; it is 0.22 to 0.29 for $f_\beta(\boldsymbol{s}, \boldsymbol{\Delta}_\theta(\boldsymbol{s}))$ and 0.32 to 0.40 for sampled actions.

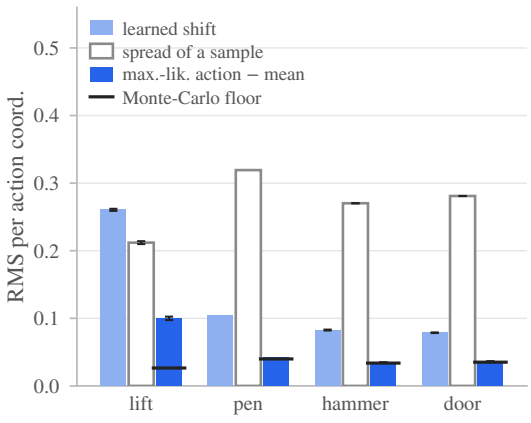

Figure 6: Through the decoders of Typical DSRL's final checkpoints: learned shift $f_\beta(\boldsymbol{s}, \boldsymbol{\Delta}_\theta(\boldsymbol{s})) - f_\beta(\boldsymbol{s}, \mathbf{0})$, spread of a sampled action around $f_\beta(\boldsymbol{s}, \boldsymbol{\Delta}_\theta(\boldsymbol{s}))$, and distance from $f_\beta(\boldsymbol{s}, \boldsymbol{\Delta}_\theta(\boldsymbol{s}))$ to the mean of sampled actions.

Divergence across states. Averaged over the pool's visited states, the divergence $\frac{1}{2}\|\boldsymbol{\Delta}_\theta(\boldsymbol{s})\|^2$ is 72 to 97% of the budget on the four tasks, and 15 to 31% of the states exceed it. Averaged at every chunk over the 100 recorded maximum-likelihood episodes of each checkpoint, and over checkpoints, it peaks at $1.5\,\delta$ around the grasp on lift and at $3.9\,\delta$ at the strike on hammer; on door it stays below $0.6\,\delta$ until the door opens and rises to $1.6\,\delta$ by the end of the episode.

Deterministic evaluation of DSRL-NA. On door the standard deviation of DSRL-NA's latent policy, per coordinate, is 0.31 to 0.35 (the σ_{eff} of Section C.3), against 1 for Typical DSRL. On every success task its deterministic and stochastic plateaus differ by at most 0.02 (Tables 8 and 9).

Interventions (Figure 7). Each rule sets the latent executed at every chunk t (Table 12), for 100 episodes per checkpoint with the environment seeds of Section C.3, so that every rule starts from the same initial states, up to lift's initial cube height, which the simulator does not reproduce across processes. The noise $\boldsymbol{\epsilon}_t$ of a given episode and chunk is the same under every rule that draws one, $\hat{\boldsymbol{\Delta}} = \boldsymbol{\Delta}_\theta(\boldsymbol{s})/\|\boldsymbol{\Delta}_\theta(\boldsymbol{s})\|$, and one latent per episode reuses the draw $\boldsymbol{\epsilon}_0$ of the first chunk. Holding one noise draw for a whole episode is worse than drawing a fresh one per chunk, so the loss of the sampled latent action is not jitter between chunks: against the sampled latent action, one $\boldsymbol{\epsilon}$ per episode loses 88, 44 and 58 episodes and wins 9, 22 and 21 on lift, door and hammer (exact McNemar $p < 10^{-4}$, 0.009 and $< 10^{-4}$), and 7 each way on pen. The mean of eight decodes wins 23, 30, 36 and 8 episodes and loses 8, 4, 2 and 1 on lift, door, hammer and pen.

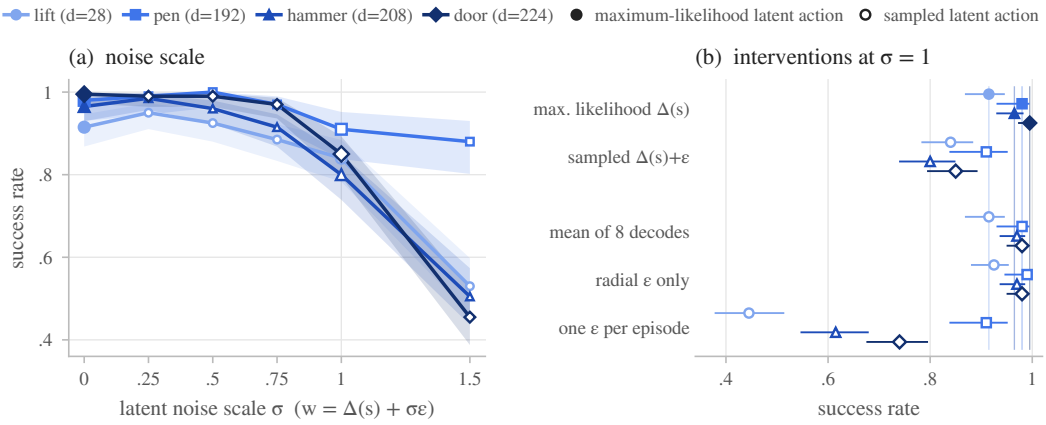

Figure 7: Test-time interventions on the latent of Typical DSRL's final checkpoints, success with 95% intervals; in (b), vertical lines mark the maximum-likelihood latent action.

Table 12: Success under each test-time rule, pooled over the checkpoints of a task.

Rule		lift $d = 28$	pen $d = 192$	hammer $d = 208$	door $d = 224$
$\boldsymbol{\Delta}$	maximum-likelihood latent action	0.92	0.98	0.96	0.99
$\boldsymbol{\Delta} + 0.25\,\boldsymbol{\epsilon}_t$		0.95	0.99	0.98	0.99
$\boldsymbol{\Delta} + 0.5\,\boldsymbol{\epsilon}_t$		0.93	1.00	0.96	0.99
$\boldsymbol{\Delta} + 0.75\,\boldsymbol{\epsilon}_t$		0.89	0.97	0.92	0.97
$\boldsymbol{\Delta} + \boldsymbol{\epsilon}_t$	sampled latent action	0.84	0.91	0.80	0.85
$\boldsymbol{\Delta} + 1.5\,\boldsymbol{\epsilon}_t$		0.53	0.88	0.51	0.46
$\frac{1}{8}\sum_k f_\beta(\boldsymbol{s}, \boldsymbol{\Delta} + \boldsymbol{\epsilon}_{t,k})$	mean of 8 decodes	0.92	0.98	0.97	0.98
$\boldsymbol{\Delta} + (\boldsymbol{\epsilon}_t^\top \hat{\boldsymbol{\Delta}})\,\hat{\boldsymbol{\Delta}}$	radial	0.93	0.99	0.97	0.98
$\boldsymbol{\Delta} + \boldsymbol{\epsilon}_t - (\boldsymbol{\epsilon}_t^\top \hat{\boldsymbol{\Delta}})\,\hat{\boldsymbol{\Delta}}$	tangential	0.85	0.97	0.82	0.87
$\boldsymbol{\Delta} + \sqrt{d}\,\boldsymbol{\epsilon}_t/\|\boldsymbol{\epsilon}_t\|$	on the shell	0.86	0.92	0.81	0.85
$\boldsymbol{\Delta} + \boldsymbol{\epsilon}_0$	one $\boldsymbol{\epsilon}$ per episode	0.45	0.91	0.61	0.74
episodes		200	100	200	200

Critic (Figure 8). On the pool's states from successful episodes, the minimum of the two latent critics at $\boldsymbol{\Delta}_\theta(\boldsymbol{s}) + \sigma\boldsymbol{\epsilon}_k$, 64 draws shared across σ, minus its value at $\boldsymbol{\Delta}_\theta(\boldsymbol{s})$, in units of its standard deviation over the draws at $\sigma = 1$. At $\sigma = 1$ the critic ranks 21 to 23% of the samples above the

maximum-likelihood latent action on lift, 26% on pen, 19 to 45% on door and 65 to 68% on hammer. The gain of the maximum-likelihood latent action is thus not the critic's: on hammer, the task with the largest gain of the four, the critic prefers the samples to it.

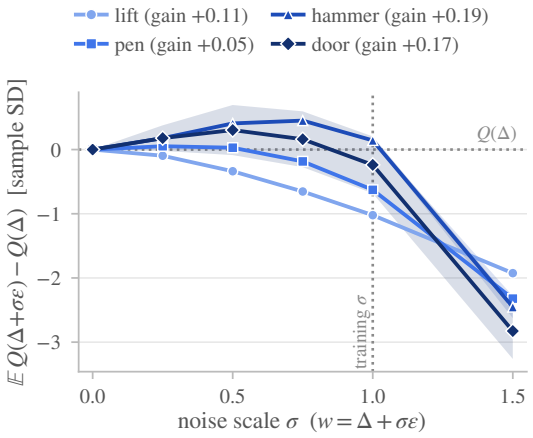

Figure 8: Latent critic of Typical DSRL, mean over states and checkpoints; the gain is that of Figure 5.

C.5 Temperature (Section 3.3)

Typical DSRL fixes the temperature α per task, at 0.05 on square. Figure 9 varies it around that value. Typical DSRL is largely robust to the choice of temperature within these values on square.

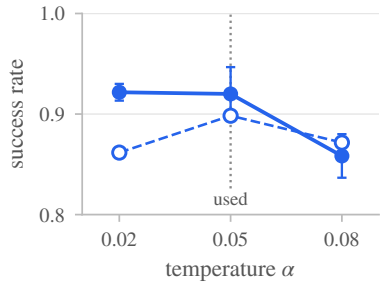

Figure 9: Temperature α on square: maximum-likelihood (filled) and sampled (hollow) latent action.