arXiv:2605.08733v1 [cs.LG] 9 May 2026

PREPRINT

Generative Actor-Critic with Soft Bridge Policies

Ke He[1] Le He[1] Shunpu Tang[2] Yafei Wang[3] Lisheng Fan[1]
[1] Guangzhou University [2] Zhejiang University [3] Southeast University

Abstract. Expressive generative policies such as diffusion and flow models are appealing for maximum-entropy (MaxEnt) online reinforcement learning because of their ability to model multimodal and highly non-Gaussian action distributions. However, training effective soft generative policies faces two obstacles that often arise together. First, marginal action densities are often unavailable, so existing methods typically rely on entropy bounds, heuristic proxies or approximations. Second, iterative shared-parameter samplers raise inference cost and require backpropagation through time over repeated network evaluations, increasing memory cost and destabilizing policy optimization. These obstacles motivate us to seek a generative policy that exposes a tractable MaxEnt objective while requiring only a single sampled actor forward pass for action generation. To this end, we propose *soft generative actor-critic* (`SoftGAC`), whose actor defines a stochastic bridge from a fixed base latent to a terminal action latent in pre-tanh space. This structured bridge allows us to lift the MaxEnt objective as an analytically tractable path-wise relative-entropy objective against a high-entropy reference process. In practical finite-step implementation, this relative entropy reduces exactly to sampled transition control energy and thus provides principled soft regularization. Moreover, we keep the single-pass actor lightweight by using small step-specific bridge transitions, each evaluated only once per sampled action, while maintaining a parameter budget comparable to strong actor baselines. Extensive experiments on challenging continuous-control benchmarks show that `SoftGAC` attains higher or competitive returns than strong generative policy baselines, including diffusion and flow-matching policies, while staying in the low-latency regime of one-pass actors and avoiding the much higher cost of many-step diffusion samplers, showing considerable improvements in the compute-return tradeoff.

Code and resources https://github.com/skypitcher/soft_gac_jax

1 Introduction

Maximum-entropy (MaxEnt) online reinforcement learning (RL) has become a central approach for off-policy continuous-control because it combines value improvement with explicit stochasticity [1–4]. In soft actor-critic (SAC) [4], this stochasticity is not only an exploration device. It is part of the optimization objective, where the actor is encouraged to choose high-value actions while maintaining high entropy [3]. This principle is simple and effective when the policy has a tractable density, but the usual Gaussian policy class can be too restrictive for complex control problems that may require multimodal or highly non-Gaussian action distributions [5–8].

Expressive generative policies offer a natural way to expand the actor class [9–13]. Diffusion, score-based and flow-style policies can represent rich action distributions and have recently become attractive for reinforcement learning [14–18]. Their appeal, however, creates a tension with MaxEnt RL. The soft actor update is an endpoint-density objective that requires policy entropy or an equivalent density-based regularizer. For many generative actors, the marginal action density is unavailable or expensive to evaluate because the action is obtained after marginalizing over latent variables or generation paths. Existing methods therefore often introduce entropy bounds [19], entropy estimators and approximations [20, 15], noise-augmented path likelihoods [21] or Wasserstein-geometric proxies [22, 23], which make the connection to MaxEnt RL indirect.

A second difficulty comes from how many generative policies spend computation. Diffusion or flow-style samplers with many denoising or refinement steps usually reuse the same network across time [13, 19, 14, 15]. This raises inference cost at deployment and it also makes the number of function evaluations (NFE) a training-time cost, because the actor update differentiates $Q(s, a)$ through the sampled action and every refinement step that produced it [24]. Increasing NFE therefore increases BPTT depth, activation memory and actor-update wall-clock [25]. It can also lengthen gradient paths through repeated uses of shared parameters, which may destabilize actor optimization. As a result, high-NFE policies may achieve stronger performance, but the added optimization burden often limits their marginal gains [15]. Conversely, reducing NFE lowers both training and inference cost, but can cause a sharp performance drop [8, 19].

These two obstacles motivate us to seek a different design principle. Specifically, we seek a generative actor that retains the expressiveness of stochastic latent generation while exposing a tractable soft objective and avoiding a long shared-parameter sampler. The goal is not to replace repeated sampler evaluations with a larger network that hides the cost in parameters. Instead, we ask whether careful actor structure can match or even exceed the performance of high-NFE diffusion policies with a single sampled forward pass and a parameter budget comparable to strong actor baselines. Our key idea is to make the actor a lightweight, explicit path law over latent variables, rather than treating it only through its terminal action distribution. This path-law view addresses the soft-regularization problem by replacing endpoint entropy with path-wise relative entropy to a high-entropy reference process. Its bridge structure addresses the computation problem by using a small number of lightweight step-specific transitions that are evaluated once per action under a compact actor parameter budget, rather than repeatedly applying a shared sampler. The resulting path Kullback-Leibler (KL) divergence refines the endpoint MaxEnt principle because it contains an endpoint KL term to the reference terminal action law. It also adds a principled regularizer on how the actor reaches the terminal action.

We instantiate this idea as *soft generative actor-critic* (`SoftGAC`), an off-policy actor-critic method with soft bridge policies, as demonstrated in Figure 1. The parameter-efficient actor uses a short sequence of lightweight local Gaussian transitions to form a stochastic bridge in pre-tanh latent space from a fixed base latent to a terminal action latent. These explicit local transitions make the path-wise relative entropy to the reference bridge analytically tractable. For the finite-step bridge used in the algorithm, its trainable part reduces exactly to a sampled transition-control-energy objective. The resulting actor update directly trades off critic value against this sampled control energy, while action generation requires one sampled pass through the bridge blocks.

Our contributions are three-fold. **(i)** We formulate a path-space soft objective for generative actors, show that it contains the endpoint KL regularizer used by MaxEnt RL as a marginal component and characterize both its unrestricted endpoint-equivalent optimum and the effect of using a practical fixed actor base. **(ii)** We design a soft bridge policy that uses a short sequence of local Gaussian transitions in pre-tanh latent space, making the path-wise regularizer exactly computable as finite-step control energy and enabling a direct actor update with action generation by a single sampled bridge pass under a comparable actor parameter budget. **(iii)** We demonstrate on challenging continuous-control benchmarks that `SoftGAC` improves the compute-return tradeoff by pairing strong returns with low one-pass action-generation cost and a parameter budget comparable to strong actor baselines.

2 Preliminaries

Maximum-entropy reinforcement learning. We consider a discounted Markov decision process with state space $\mathcal{S}$, bounded continuous action space $\mathcal{A} \subset \mathbb{R}^{d_a}$, reward $r(s, a)$, transition kernel $p(s' \mid s, a)$, discount factor $\gamma \in (0, 1)$ and policy $\pi(a \mid s)$. Maximum-entropy reinforcement learning augments the expected return with an entropy bonus,

$$J(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t \left(r(s_t, a_t) + \alpha \mathcal{H}(\pi(\cdot \mid s_t)) \right) \right], \tag{1}$$

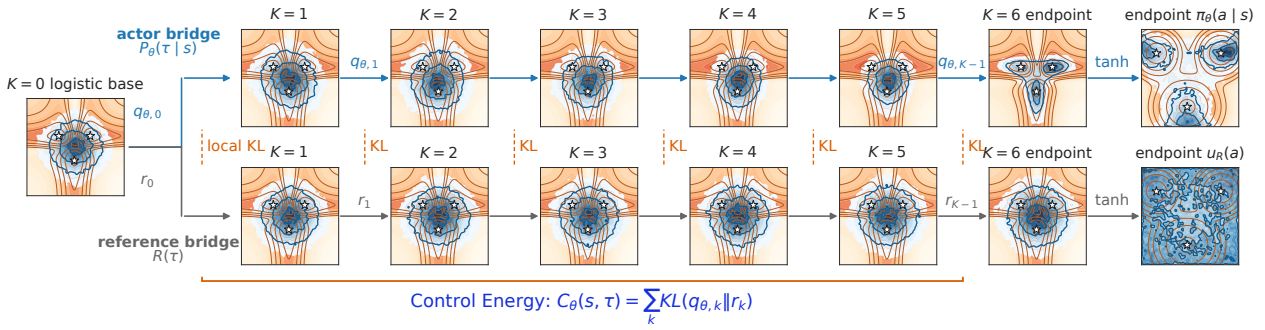

Figure 1. Overview of the proposed soft bridge policy. Local KL terms compare actor and reference transitions and sum to the sampled control energy. The terminal latent is mapped through tanh and optimized by the critic. Appendix C provides a 2D bridge visualization.

where $\alpha > 0$ controls the strength of the soft regularizer. In an off-policy actor-critic algorithm, a critic estimates a soft value landscape and the actor is improved by increasing value while preserving stochasticity. When $\mathcal{A}$ is bounded and u denotes the uniform action law, the standard per-state soft actor improvement step instantiated by SAC [4] can be written as

$$\mathcal{J}_{\mathrm{SAC}}(\pi \mid s) = \mathbb{E}_{a\sim\pi(\cdot|s)}[Q(s,a)] - \alpha D_{\mathrm{KL}}(\pi(\cdot \mid s) \,\|\, u) + \text{const.} \tag{2}$$

This form will be useful below because it separates the value term from a regularizer that keeps the endpoint action distribution close to a high-entropy reference. Explicit-density policies such as standard SAC can evaluate this objective because the policy density is available. The difficulty begins when the policy is a generative sampler whose endpoint density is not directly exposed.

Generative actors as path laws. Many recent generative policies naturally fit a common path-law view. A diffusion actor [9, 7, 19] starts from base noise and samples a reverse denoising chain,

$$y_{k-1} \sim p_\theta(y_{k-1} \mid y_k, s), \qquad y_K \sim \mathcal{N}(0, I), \tag{3}$$

while a flow or flow-matching actor [11, 12, 18] evolves base noise through a learned velocity field,

$$\frac{dy_t}{dt} = v_\theta(y_t, t, s), \qquad y_0 \sim p_0. \tag{4}$$

Beyond multi-step implicit generators, one-shot implicit actors form a degenerate short-path case, where $z_0 \sim p_0$ is mapped by a neural generator to a terminal latent $z_1 = f_\theta(z_0, s)$ [23]. These actors all generate an action by following a latent path from base noise to a terminal latent state. For the rest of the paper, we use the forward ordering $\tau = (z_0, z_1, \ldots, z_K)$, where z_0 denotes the base latent and z_K denotes the terminal action latent. The actor samples this path from a path law $P_\theta(d\tau \mid s)$ and produces an action through a terminal map $a = T(z_K)$. In this paper, T will map the terminal latent state to a bounded action through a tanh transform, but we only need the induced endpoint law $\pi_\theta(\cdot \mid s) = (T \circ z_K)_\# P_\theta(\cdot \mid s)$, namely the distribution of $T(z_K)$ when $\tau \sim P_\theta(\cdot \mid s)$. Such an actor can be easy to sample from while still having an intractable marginal action density. The density of $\pi_\theta(a \mid s)$ may require integrating over all latent paths that terminate at the same action, which is generally unavailable for implicit generators and costly for multi-step diffusion or flow samplers.

3 Maximum-Entropy Reinforcement Learning in Path Space

3.1 Path-Space Objective

When the terminal action density is unavailable, a natural approach is to lift MaxEnt RL from the terminal action distribution to the full generation path. Since a generative actor samples a latent path τ before producing

a terminal action, we define the soft objective over path laws. Let $R(d\tau)$ be a high-entropy reference path law in the same latent space and let u_R be its terminal action law induced by $T(z_K)$. For a fixed state s and critic Q, we consider

$$\mathcal{J}_{\text{path}}(P_\theta \mid s) = \mathbb{E}_{\tau\sim P_\theta(\cdot|s)}\left[Q(s, T(z_K))\right] - \alpha D_{\text{KL}}\left(P_\theta(\cdot \mid s) \,\|\, R\right). \tag{5}$$

The value term still evaluates only the terminal action. The regularizer, however, now compares the full generation path to a high-entropy reference process. This path-space objective is useful only if it remains connected to the endpoint MaxEnt objective. We show this connection by decomposing the path KL into a terminal-action term and a conditional path term.

Proposition 1 (Path-space lift of the endpoint KL regularizer). Let $a = T(z_K)$ be the terminal action. Suppose the actor path law and the reference path law admit disintegrations

$$P_\theta(d\tau \mid s) = \pi_\theta(da \mid s)\, P_\theta(d\tau \mid s, a), \qquad R(d\tau) = u_R(da)\, R(d\tau \mid a). \tag{6}$$

Then

$$D_{\text{KL}}\left(P_\theta(\cdot \mid s) \,\|\, R\right) = D_{\text{KL}}\left(\pi_\theta(\cdot \mid s) \,\|\, u_R\right) + \mathbb{E}_{a\sim\pi_\theta}\left[D_{\text{KL}}\left(P_\theta(\cdot \mid s, a) \,\|\, R(\cdot \mid a)\right)\right]. \tag{7}$$

The proof is a direct chain-rule decomposition of relative entropy and appears in Appendix A.1. This decomposition shows that the path KL contains the endpoint KL regularizer as its marginal component. When u_R is uniform over the bounded action space, the first term in Eq. (7) is exactly the uniform-prior endpoint regularizer in Eq. (2) up to a constant. The second term is the additional structure introduced by the path-space lift. It regularizes how the actor generates an action, not only which terminal action distribution it induces. Thus the path objective is stricter than endpoint entropy while still preserving the endpoint regularizer of MaxEnt RL.

3.2 Unrestricted Soft-Optimal Bridge

We next ask whether this stricter objective changes the ideal soft-optimal endpoint action distribution. If we optimize over all path laws that are absolutely continuous with respect to the reference, the answer is no in the ideal uniform-reference case.

Theorem 1 (Unrestricted soft-optimal bridge). For fixed s, Q and $\alpha > 0$, consider the unrestricted optimization of Eq. (5) over path laws P with $P \ll R$. Assume

$$Z_Q(s) = \mathbb{E}_{\tau\sim R}\left[\exp\left(Q(s, T(z_K))/\alpha\right)\right] < \infty. \tag{8}$$

The unique optimum is the value-tilted reference path law

$$\frac{dP^\star_Q}{dR}(\tau \mid s) = \frac{\exp\left(Q(s, T(z_K))/\alpha\right)}{Z_Q(s)}. \tag{9}$$

Its terminal action law is

$$\pi^\star_Q(a \mid s) = \frac{u_R(a)\exp\left(Q(s,a)/\alpha\right)}{\int_{\mathcal{A}} u_R(b)\exp\left(Q(s,b)/\alpha\right) db}. \tag{10}$$

The proof follows from the Gibbs variational identity and appears in Appendix A.2. Since the value tilt in Eq. (9) depends only on the terminal action, the optimal conditional path law given that action remains the reference conditional bridge $R(d\tau \mid a)$. Value changes the endpoint marginal, while the reference controls how each endpoint is reached. When u_R is uniform, Eq. (10) recovers the usual Boltzmann endpoint policy

proportional to $\exp(Q(s,a)/\alpha)$. The unrestricted path-space objective is therefore *endpoint-equivalent* to the MaxEnt actor update in the ideal reference limit, but it is more selective about the generation path.

3.3 Fixed-Base Bridge Choice

The unrestricted optimum gives a clean conceptual target, but every implementable generative actor also needs to specify how its latent path starts. A diffusion sampler, flow sampler or one-shot generator all begin from some base law. This base law is a practical modeling choice. If the actor base differs from the reference base, the path KL gains an initial-law term before the remaining conditional path is compared to the reference. For a fixed actor base, this initial term is constant with respect to the conditional path law after z_0. Thus the base choice does not change the variational problem solved after conditioning on the base sample. In finite implementations, it mainly acts as an inductive bias by shaping the initial latents from which the rest of the generative path is drawn. To characterize this effect, let the reference path law disintegrate as

$$R(d\tau) = r_0(dz_0)\, R(d\tau_{1:K} \mid z_0), \tag{11}$$

and fix an actor base law p_0 with $p_0 \ll r_0$. We optimize the same path objective as Eq. (5), but restrict the actor path law to satisfy $P_0 = p_0$. The value tilt still points toward the unrestricted soft optimum in Theorem 1, while the actor can only search within the selected fixed-base family. The best attainable solution is therefore the constrained optimum below.

Theorem 2 (Fixed-base bridge optimum). Define

$$Z_Q(s,z_0) = \mathbb{E}_{\tau\sim R(\cdot|z_0)}\left[\exp\left(Q(s,T(z_K))/\alpha\right)\right], \qquad \bar{Z}_Q(s) = \mathbb{E}_{z_0\sim r_0}\left[Z_Q(s,z_0)\right]. \tag{12}$$

Among all path laws with initial marginal p_0, the unique optimum is

$$P^\star_{Q,p_0}(d\tau \mid s) = p_0(dz_0)\,\frac{\exp\left(Q(s,T(z_K))/\alpha\right)}{Z_Q(s,z_0)}\,R(d\tau_{1:K} \mid z_0). \tag{13}$$

Its objective value is

$$\alpha\mathbb{E}_{z_0\sim p_0}\left[\log Z_Q(s,z_0)\right] - \alpha D_{\mathrm{KL}}\left(p_0 \,\|\, r_0\right). \tag{14}$$

The proof applies the Gibbs variational identity conditionally on the fixed base latent and appears in Appendix A.3. The initial KL term is independent of the conditional optimizer after z_0. Thus using a different fixed base is not a heuristic break from the objective. It changes the full path objective by a constant and changes the finite actor's inductive bias through the initial samples. When the fixed actor base matches the value-tilted initial marginal of the unrestricted bridge, this constrained optimum coincides with the unrestricted one. Otherwise, the actor does not reweight the initial latent distribution in a state-dependent way, and the corollary quantifies the resulting gap.

Corollary 1 (Base-constraint gap controls endpoint deviation). Under the assumptions of Theorem 2, the loss relative to the unrestricted optimum is

$$\Delta(s) = \alpha\left(\log\bar{Z}_Q(s) - \mathbb{E}_{z_0\sim p_0}\left[\log Z_Q(s,z_0)\right] + D_{\mathrm{KL}}\left(p_0 \,\|\, r_0\right)\right) \geq 0. \tag{15}$$

Equality holds if and only if p_0 equals the initial marginal of the unrestricted tilted bridge. In the special case $p_0 = r_0$, this reduces to $Z_Q(s,z_0)$ being constant for p_0-almost every z_0. Moreover,

$$D_{\mathrm{KL}}\left(P^\star_{Q,p_0}(\cdot \mid s) \,\|\, P^\star_Q(\cdot \mid s)\right) = \frac{\Delta(s)}{\alpha}, \tag{16}$$

and the induced endpoint deviation obeys

$$D_{\mathrm{KL}}\left(\pi^{\star}_{Q,p_0}(\cdot \mid s) \,\|\, \pi^{\star}_{Q}(\cdot \mid s)\right) \le \frac{\Delta(s)}{\alpha}. \tag{17}$$

The proof is given in Appendix A.4. The unrestricted optimum can tilt the initial latent law by $Z_Q(s, z_0)$, which would require a state-dependent base sampler before any bridge transition. A fixed-base actor avoids this sampler. Its cost is $\Delta(s)$, and we show that this same cost controls the endpoint deviation. The bound is small unless $Z_Q(s, z_0)$ varies sharply across likely base latents, meaning some base regions are much better positioned to reach high-value actions than others. In the regime we target, typical base samples can be transported to useful endpoints. The base choice then acts mainly as a practical inductive bias, while the bridge transitions perform value-guided transport. For a finite-step implementation, the reference terminal law is u_R rather than the ideal uniform action law. The endpoint-MaxEnt connection is exact with respect to u_R and recovers the uniform-prior view when $u_R = u$. What `SoftGAC` optimizes is more concrete. For the chosen actor-reference pair, the finite-step path KL is the exact regularizer up to the fixed initial-base constant. We now need an architecture that exposes this KL from sampled paths.

4 Path-Regularized Generative Actor-Critic

The preceding section turns soft policy improvement into a path-law optimization problem with a fixed base distribution. This leaves an architectural question. How can the actor expose the path likelihood needed by the regularizer while remaining cheap to sample? We answer this question with `SoftGAC`, a path-regularized actor-critic that combines a short bridge actor with an off-policy soft critic. The bridge actor makes the finite-step path KL factor into local Gaussian terms. These terms will become the sampled control energy used by the actor and critic updates.

4.1 Soft Bridge Policies

A soft bridge policy is a finite Markov chain in pre-tanh latent space. It starts from a fixed base law and applies K lightweight Gaussian residual transitions before mapping the terminal latent state to the bounded action. This gives an explicit path law and avoids the long shared-parameter sampler used by high-NFE iterative policies. We write its transition law as

$$z_0 \sim p_0, \qquad q_{\theta,k}(z_{k+1} \mid z_k, s) = \mathcal{N}\left(z_k + h\, u_{\theta,k}(s, z_k), 2h \,\mathrm{diag}(\sigma^2_{\theta,k}(s, z_k))\right), \tag{18}$$

for $k = 0, \dots, K-1$ and $h = 1/K$. The terminal action is $a = T(z_K) = \tanh(z_K)$ after the usual affine rescaling to the environment action bounds. Each transition block has its own parameters and communicates with the next block only through the action-dimensional latent state z_k. Thus the forward pass is a stochastic generative process, but it is still a fixed-depth one-pass network rather than an iterative sampler that reuses the same refinement model many times. During actor updates, gradients pass through this short sequence of transition blocks instead of a long BPTT graph over repeated uses of a shared sampler.

The reference bridge plays the role of a high-entropy action prior. In normalized action space $(-1, 1)^{d_a}$, the pre-tanh density whose tanh image is uniform is

$$q_{\mathrm{ref}}(z) = \frac{1}{2^{d_a}} \prod_{i=1}^{d_a} \mathrm{sech}^2(z_i), \qquad \nabla_z \log q_{\mathrm{ref}}(z) = -2\tanh(z). \tag{19}$$

For the main implementation, we set the actor base law to this reference density, $p_0 = q_{\mathrm{ref}}$. This keeps the full path KL free of an initial constant and gives the cleanest decomposition below. Other fixed bases are also valid.

They add a parameter-independent initial KL to the full objective and mainly act as an implementation-level inductive bias through the base samples. The finite-step reference starts from the logistic reference base and uses an Euler Gaussian kernel based on the score above,

$$r_k(z_{k+1} \mid z_k) = \mathcal{N}\left(z_k - 2h\tanh(z_k), 2hI\right). \tag{20}$$

The continuous reference diffusion is stationary when initialized from q_{ref}, so its tanh image is uniform in action space at every time. The finite-step reference is the object used by the algorithm. The path KL below is exact for this bridge pair, while Appendix D quantifies its endpoint bias.

Lemma 1 (Finite-step control-energy decomposition). Let

$$P_\theta(d\tau \mid s) = p_0(dz_0)\prod_{k=0}^{K-1} q_{\theta,k}(dz_{k+1} \mid z_k, s), \qquad R(d\tau) = r_0(dz_0)\prod_{k=0}^{K-1} r_k(dz_{k+1} \mid z_k). \tag{21}$$

Then

$$D_{\mathrm{KL}}\left(P_\theta(\cdot \mid s) \,\|\, R\right) = D_{\mathrm{KL}}\left(p_0 \,\|\, r_0\right) + \sum_{k=0}^{K-1} \mathbb{E}_{\tau\sim P_\theta}\left[D_{\mathrm{KL}}\left(q_{\theta,k}(\cdot \mid z_k, s) \,\|\, r_k(\cdot \mid z_k)\right)\right]. \tag{22}$$

For Gaussian local kernels, define $C_\theta(s,\tau)$ as the sum of the local Gaussian KL terms. Then

$$D_{\mathrm{KL}}\left(P_\theta(\cdot \mid s) \,\|\, R\right) = D_{\mathrm{KL}}\left(p_0 \,\|\, r_0\right) + \mathbb{E}_{\tau\sim P_\theta(\cdot|s)}\left[C_\theta(s,\tau)\right]. \tag{23}$$

The proof is a factorization of the Markov path likelihood ratio and appears in Appendix A.5. When $p_0 = r_0$, the initial KL vanishes. When p_0 is a different fixed base, this initial term is constant with respect to the actor transitions, so actor training can use the same sampled transition control energy. For the residual actor in Eq. (18) and the reference in Eq. (20), the local control cost is

$$C_k(s,z_k) = \frac{1}{2}\sum_{i=1}^{d_a}\left[\sigma^2_{\theta,k,i}(s,z_k) + \frac{\left(\mu_{\theta,k,i}(s,z_k) - \mu_{R,k,i}(z_k)\right)^2}{2h} - 1 - \log\sigma^2_{\theta,k,i}(s,z_k)\right], \tag{24}$$

where $\mu_{\theta,k}(s,z_k) = z_k + h\,u_{\theta,k}(s,z_k)$ and $\mu_{R,k}(z_k) = z_k - 2h\tanh(z_k)$. The sampled path cost is $C_\theta(s,\tau) = \sum_{k=0}^{K-1} C_k(s,z_k)$. This quantity is the finite-step transition control energy of the bridge path, and its expectation is the actor-reference path KL up to the fixed initial-base constant. The soft regularizer in `SoftGAC` is therefore an analytical path-wise relative entropy rather than an estimate, bound or proxy for endpoint action entropy. The next subsection uses this finite-step path regularizer inside an off-policy actor-critic update.

4.2 Off-Policy Actor-Critic Training

We train this bridge actor with an off-policy critic in `SoftGAC`. For a fixed bridge policy, define the path-regularized soft value

$$V^\pi(s) = \mathbb{E}_{\tau\sim P_\theta(\cdot|s)}\left[Q^\pi(s, T(z_K)) - \alpha\, C_\theta(s,\tau)\right]. \tag{25}$$

The corresponding Bellman equation is

$$Q^\pi(s,a) = r(s,a) + \gamma\mathbb{E}_{s'}\left[V^\pi(s')\right]. \tag{26}$$

Thus the only change from a standard soft actor-critic update is that endpoint entropy is replaced by sampled bridge control energy. Given a replay batch (s,a,r,s',d), we sample a next-state bridge path $\tau' \sim P_{\bar\theta}(\cdot \mid s')$ from the target actor and form the soft bootstrap scalar

$$y = r + \gamma(1-d)\left[Q^{\min}_{\bar\phi}(s', T(z'_K)) - \alpha\, C_{\bar\theta}(s',\tau')\right]. \tag{27}$$

The critic can be any off-policy estimator trained toward this target. Our implementation uses a twin categorical critic with a fixed support $\mathcal{Z} = \{v_1, \ldots, v_M\}$ and a CrossQ-style update [26, 27, 19]. For each critic head, the target distribution is obtained by shifting the support by the sampled soft bootstrap and projecting back to $\mathcal{Z}$ with the usual categorical projection. This critic choice is an implementation detail rather than a requirement of the bridge objective. The actor is updated by sampling current-state paths $\tau \sim P_\theta(\cdot \mid s)$ and minimizing

$$\mathcal{L}_{\text{actor}}(\theta) = \mathbb{E}_{s\sim\mathcal{D}}\mathbb{E}_{\tau\sim P_\theta(\cdot|s)}\left[\alpha\, C_\theta(s,\tau) - Q_\phi^{\min}(s, T(z_K))\right]. \tag{28}$$

At a fixed state, this sampled objective has a precise projection interpretation. It moves the transition actor toward the ideal soft-optimal bridge, while the chosen fixed base determines the practical actor family being searched.

Proposition 2 (Actor update as restricted projection to the ideal bridge). Let $P_Q^\star$ be the unrestricted soft-optimal bridge in Theorem 1. For any finite-step bridge actor with fixed base law $p_0 \ll r_0$ that satisfies Lemma 1,

$$\mathcal{L}_{\text{actor}}(\theta \mid s) = \mathbb{E}_{\tau\sim P_\theta(\cdot|s)}\left[\alpha\, C_\theta(s,\tau) - Q(s, T(z_K))\right] \tag{29}$$

satisfies

$$\mathcal{L}_{\text{actor}}(\theta \mid s) = \alpha D_{\text{KL}}\left(P_\theta(\cdot \mid s) \,\|\, P_Q^\star(\cdot \mid s)\right) - \alpha \log Z_Q(s) - \alpha D_{\text{KL}}\left(p_0 \,\|\, r_0\right). \tag{30}$$

Consequently, minimizing this loss over the unrestricted fixed-base path-law family has global minimizer $P_{Q,p_0}^\star$ from Theorem 2.

The proof appears in Appendix A.6. The global-minimizer statement is a property of the unrestricted fixed-base path-law family. The finite neural Gaussian Markov actor optimized by SGD is a restricted parameterization, so the result should not be read as a global optimization guarantee for the implemented network. Equation (30) is nevertheless useful algorithmically. Once the base law is fixed, the final two terms are constants with respect to the transition actor parameters, and the sampled actor loss follows a reverse-KL projection direction toward the ideal bridge. In the implemented neural family, the loss defines a tractable finite-step objective whose attainable solution is limited by parameterization and optimization.

In addition, we tune the temperature with a SAC-style dual update on the control-energy budget. Let $C_{\text{target}} = \rho_{\text{ctrl}} K d_a$ be a per-step, per-dimension heuristic target cost. Appendix E.4 gives the scale intuition behind this choice. The temperature objective is

$$\mathcal{L}_\alpha = \mathbb{E}\left[\alpha\left(C_{\text{target}} - C_\theta(s,\tau)\right)\right], \qquad \alpha = \exp(\log\alpha). \tag{31}$$

This keeps the average bridge deviation from the high-entropy reference near a chosen budget. At deployment, the policy uses the same bridge forward pass but discards the control-energy bookkeeping. Action generation remains fixed-cost with exactly K local transition blocks. The complete training loop is given in Algorithm 1 in Appendix B.

5 Experiments

Experimental setup. We evaluate `SoftGAC` on challenging high-dimensional continuous-control tasks from the DeepMind Control Suite (DMC) [28, 29] and HumanoidBench [30]. Our goal is to measure not only return, but also the compute-return tradeoff against strong recent generative actor-critic baselines. The baselines include diffusion and flow-matching policies, represented by FLAC [22], DIME [19], FlowRL [17], QSM [14] and QVPO [15]. In addition, we also include CrossQ-SAC [27] which is a strong unimodal Gaussian policy baseline. We report interquartile mean (IQM) [31] over 8 random seeds with 95% bootstrap confidence intervals, and include ablations that remove the soft path-space regularizer to verify its contribution to the performance.

We re-implement all baselines in a single codebase under the framework of StableBaselines3 [32], and we follow the official implementations whenever available. The comparison is unified in infrastructure and comparable parameter budget, and all methods share identical main critic update. `SoftGAC` uses $K = 6$ bridge transitions, where each local Gaussian transition is represented by a lightweight single-layer MLP with mean and variance heads. In particular, we adjust actor sizes so that the methods have comparable actor parameter budgets, while the reported latency measures the complete action-generation computation for each actor. Appendix E and F give the full implementation details, hyperparameters, experimental setup, extended results and ablations.

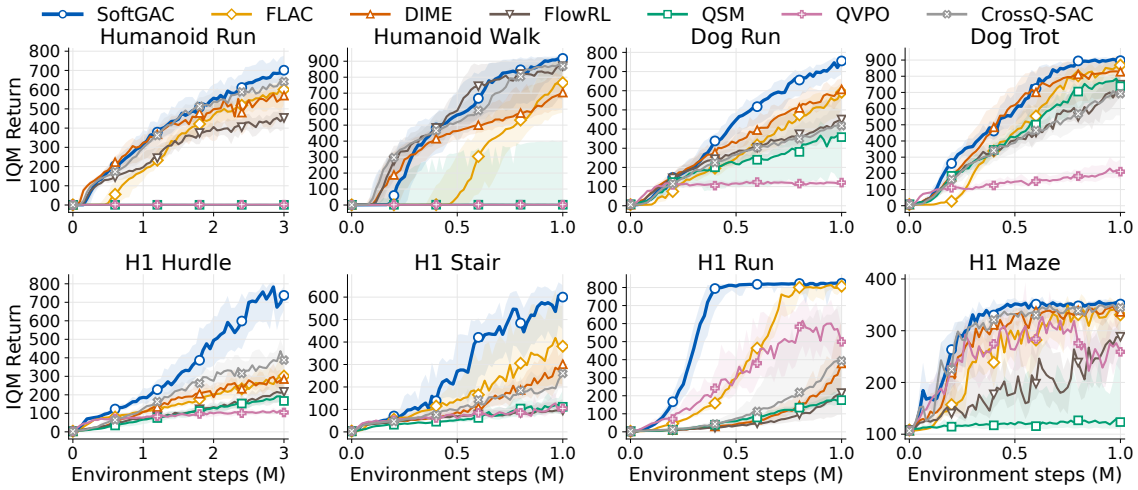

Figure 2. IQM learning curves on the 8 hard control tasks.

Performance. Figure 2 shows that `SoftGAC` gives the strongest overall performance on the hard tasks. The gains are largest on high-dimensional, long-horizon locomotion tasks such as Humanoid Run, Dog Run, H1 Hurdle and H1 Stair, where `SoftGAC` improves over both generative baselines and the CrossQ-SAC Gaussian baseline. On tasks where CrossQ-SAC is already strong, such as Humanoid Walk and H1 Maze, `SoftGAC` remains competitive or better while retaining a richer generative actor. These results suggest that the soft bridge actor improves sample efficiency and final performance, rather than only increasing expressiveness at convergence.

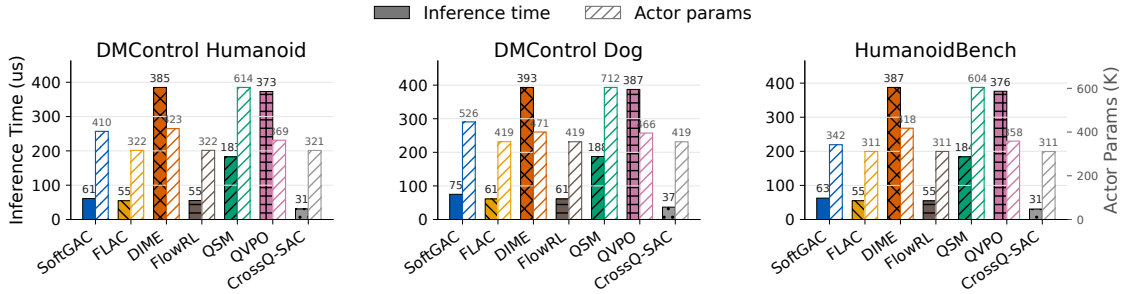

Figure 3. Per-action inference time and actor param count measured on Apple M3 Pro.

Inference cost. Figure 3 shows that this return gain does not come from unrolled sampler evaluations. CrossQ-SAC has the cheapest unimodal Gaussian actor. `SoftGAC` uses about 61–75 μs per action on the CPU benchmark, which places it in the same low-latency range as one-step flow baselines and far below high-NFE diffusion baselines. This matters because actor sampling is used both during environment interaction and inside actor-critic updates. The soft bridge policy generates each action with one sampled pass through its bridge blocks and keeps the actor parameter budget comparable, while still offering superior performance. This explains the improved compute-return tradeoff observed in Figures 2 and 3.

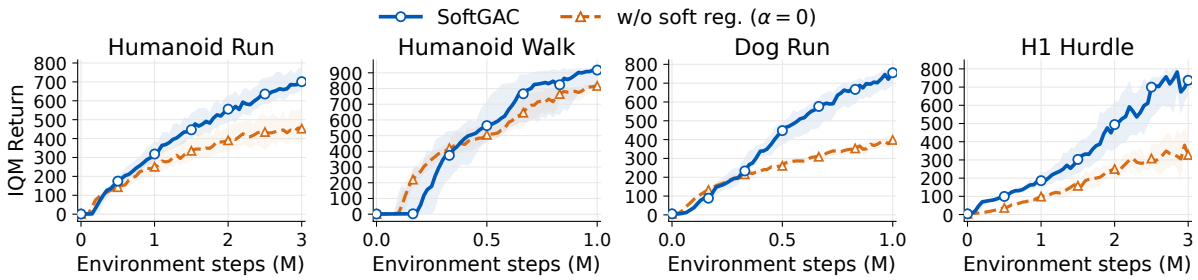

Figure 4. Ablation study of the soft regularizer.

Soft regularization. Figure 4 isolates the path-space soft regularizer by setting $\alpha = 0$ while keeping the same actor and critic. Removing this term consistently hurts Humanoid Run, Humanoid Walk, Dog Run and H1 Hurdle, which suggests that the gains do not come only from the bridge parameterization. The result supports the relative-entropy control energy as a principled soft objective for the one-pass generator, rather than only an exploration heuristic.

6 Related Work and Discussion

Expressive actors beyond diagonal Gaussians, including normalizing-flow, diffusion, score-based and flow-matching policies, have been explored for off-policy continuous-control RL [6, 7, 14, 15, 17, 18]. Since the terminal action density of an implicit generator is usually unavailable, existing soft generative policies often rely on surrogates: DIME uses an entropy bound [19], DACER estimates entropy [20], SAC-Flow uses noise-augmented rollout likelihood [21], and QVPO adds a variational-style approximation [15]. In particular, FLAC is closest in motivation because it casts MaxEnt RL as a generalized Schrödinger bridge problem, but its practical kinetic-energy regularizer is a geometric proxy that does not by itself imply a high-entropy action distribution [22]. Moreover, WPPG introduces entropy through a heat-flow step after Wasserstein proximal transport, where the W_2 term remains an action-space proximal metric [23]. In contrast, `SoftGAC` directly optimizes the finite-step actor-reference path KL, which reduces to sampled transition control energy for Gaussian bridges. On the other hand, efficiency-oriented methods such as FQL and One-Step FQL remove or distill iterative action generation and recursive backpropagation through the sampler [8, 24]. `SoftGAC` shares the goal of low-cost action generation, but derives the single-pass actor from the soft objective itself so that the regularizer remains tractable with a compact actor architecture.

7 Conclusion and Limitation

We presented `SoftGAC`, a generative actor-critic method that lifts maximum-entropy regularization from endpoint actions to the full latent generation path. This path-space view yields an analytical relative-entropy regularizer, implemented as sampled transition control energy for Gaussian bridges. It also gives a single-pass actor that avoids repeatedly applying a high-NFE shared sampler and long BPTT through that sampler. Across challenging locomotion benchmarks, `SoftGAC` achieves higher or competitive returns while remaining in the low-latency regime of one-pass actors, which supports soft bridge policies as a practical design for efficient generative actor-critic learning.

While promising, the method has some limitations. The practical actor uses finite Gaussian bridge transitions, so its endpoint connection to a uniform action prior depends on the reference discretization and base law. Although lightweight, it also introduces architecture choices such as bridge depth, base distribution and target control-energy budget.

Acknowledgments

This work was supported in part by the Google TPU Research Cloud (TRC) program, which provided access to TPU compute resources for the experiments.

References

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

[2] Junhyuk Oh, Gregory Farquhar, Iurii Kemaev, Dan A Calian, Matteo Hessel, Luisa Zintgraf, Satinder Singh, Hado Van Hasselt, and David Silver. Discovering state-of-the-art reinforcement learning algorithms. *Nature*, 648(8093):312–319, 2025.

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

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

[5] Michal Nauman, Mateusz Ostaszewski, Krzysztof Jankowski, Piotr Miłoś, and Marek Cygan. Bigger, regularized, optimistic: scaling for compute and sample efficient continuous control. *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

[6] Chen-Hao Chao, Chien Feng, Wei-Fang Sun, Cheng-Kuang Lee, Simon See, and Chun-Yi Lee. Maximum entropy reinforcement learning via energy-based normalizing flow. *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

[7] Zechu Li, Rickmer Krohn, Tao Chen, Anurag Ajay, Pulkit Agrawal, and Georgia Chalvatzaki. Learning multimodal behaviors from scratch with diffusion policy gradient. *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

[8] Seohong Park, Qiyang Li, and Sergey Levine. Flow Q-learning. *International Conference on Machine Learning (ICML)*, 2025.

[9] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

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

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

[12] David McAllister, Songwei Ge, Brent Yi, Chung Min Kim, Ethan Weber, Hongsuk Choi, Haiwen Feng, and Angjoo Kanazawa. Flow matching policy gradients. *arXiv preprint arXiv:2507.21053*, 2025.

[13] 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. *International Conference on Learning Representations (ICLR)*, 2025.

[14] Michael Psenka, Alejandro Escontrela, Pieter Abbeel, and Yi Ma. Learning a diffusion model policy from rewards via Q-score matching. *International Conference on Machine Learning (ICML)*, 2024.

[15] Shutong Ding, Ke Hu, Zhenhao Zhang, Kan Ren, Weinan Zhang, Jingyi Yu, Jingya Wang, and Ye Shi. Diffusion-based reinforcement learning via Q-weighted variational policy optimization. *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

[16] Shu-Ang Yu, Feng Gao, Yi Wu, Chao Yu, and Yu Wang. D3P: Dynamic denoising diffusion policy via reinforcement learning. *arXiv preprint arXiv:2508.06804*, 2025.

[17] Lei Lv, Yunfei Li, Yu Luo, Fuchun Sun, Tao Kong, Jiafeng Xu, and Xiao Ma. Flow-based policy for online reinforcement learning. *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2025.

[18] Tonghe Zhang, Chao Yu, Sichang Su, and Yu Wang. Reinflow: Fine-tuning flow matching policy with online reinforcement learning. *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2025.

[19] Onur Celik, Zechu Li, Denis Blessing, Ge Li, Daniel Palenicek, Jan Peters, Georgia Chalvatzaki, and Gerhard Neumann. DIME: Diffusion-based maximum entropy reinforcement learning. *International Conference on Machine Learning (ICML)*, 2025.

[20] Yinuo Wang, Likun Wang, Yuxuan Jiang, Wenjun Zou, Tong Liu, Xujie Song, Wenxuan Wang, Liming Xiao, Jiang Wu, Jingliang Duan, et al. Diffusion actor-critic with entropy regulator. *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

[21] 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. *International Conference on Learning Representations (ICLR)*, 2026.

[22] Lei Lv, Yunfei Li, Yu Luo, Fuchun Sun, and Xiao Ma. FLAC: Maximum entropy RL via kinetic energy regularized bridge matching. *arXiv preprint arXiv:2602.12829*, 2026.

[23] Zhaoyu Zhu, Shuhan Zhang, Rui Gao, and Shuang Li. Wasserstein proximal policy gradient. *arXiv preprint arXiv:2603.02576*, 2026.

[24] Thanh Xuan Nguyen and Chang D Yoo. One-step flow Q-learning: Addressing the diffusion policy bottleneck in offline reinforcement learning. *International Conference on Learning Representations (ICLR)*, 2026.

[25] Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. *International Conference on Learning Representations (ICLR)*, 2025.

[26] Marc G Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. *International Conference on Machine Learning (ICML)*, 2017.

[27] Aditya Bhatt, Daniel Palenicek, Boris Belousov, Max Argus, Artemij Amiranashvili, Thomas Brox, and Jan Peters. CrossQ: Batch normalization in deep reinforcement learning for greater sample efficiency and simplicity. *International Conference on Learning Representations (ICLR)*, 2024.

[28] Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

[29] Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020.

[30] Carmelo Sferrazza, Dun-Ming Huang, Xingyu Lin, Youngwoon Lee, and Pieter Abbeel. Humanoidbench: Simulated humanoid benchmark for whole-body locomotion and manipulation. *arXiv preprint arXiv:2403.10506*, 2024.

[31] Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Deep reinforcement learning at the edge of the statistical precipice. *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

[32] 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. URL http://jmlr.org/papers/v22/20-1364.html.

Contents

1 Introduction **1**

2 Preliminaries **2**

3 Maximum-Entropy Reinforcement Learning in Path Space **3**

3.1 Path-Space Objective . . . 3

3.2 Unrestricted Soft-Optimal Bridge . . . 4

3.3 Fixed-Base Bridge Choice . . . 5

4 Path-Regularized Generative Actor-Critic **6**

4.1 Soft Bridge Policies . . . 6

4.2 Off-Policy Actor-Critic Training . . . 7

5 Experiments **8**

6 Related Work and Discussion **10**

7 Conclusion and Limitation **10**

A Theorem Proofs **16**

A.1 Proof of Proposition 1 . . . 16

A.2 Proof of Theorem 1 . . . 16

A.3 Proof of Theorem 2 . . . 17

A.4 Proof of Corollary 1 . . . 18

A.5 Proof of Lemma 1 . . . 19

A.6 Proof of Proposition 2 . . . 20

B Algorithm Pseudocode **21**

C 2D Bridge Visualization **21**

D Finite-Step Reference Endpoint Bias **23**

E Detailed Experimental Setup **25**

E.1 Environment and Task Domains . . . 25

E.2 Training Hardware . . . 26

E.3 Implementation Details . . . 26

E.4 Choosing the Control-Energy Budget . . . 27

E.5 Hyperparameters 27

F Extended Experimental Results and Ablations **29**

A Theorem Proofs

A.1 Proof of Proposition 1

Restatement. Let $a = T(z_K)$ be the terminal action. If

$$P_\theta(d\tau \mid s) = \pi_\theta(da \mid s)\, P_\theta(d\tau \mid s, a), \qquad R(d\tau) = u_R(da)\, R(d\tau \mid a),$$

then

$$D_{\mathrm{KL}}\left(P_\theta(\cdot \mid s) \,\|\, R\right) = D_{\mathrm{KL}}\left(\pi_\theta(\cdot \mid s) \,\|\, u_R\right) + \mathbb{E}_{a\sim\pi_\theta}\left[D_{\mathrm{KL}}\left(P_\theta(\cdot \mid s, a) \,\|\, R(\cdot \mid a)\right)\right].$$

Proof. By the stated disintegrations and the terminal map $a = T(z_K)$, the Radon-Nikodym derivative factorizes into an endpoint term and a conditional path term:

$$\log \frac{dP_\theta(\tau \mid s)}{dR(\tau)} = \log \frac{d\pi_\theta(a \mid s)}{du_R(a)} + \log \frac{dP_\theta(\tau \mid s, a)}{dR(\tau \mid a)}. \tag{32}$$

This is the chain rule for relative entropy written at the level of path laws. Taking expectation under $P_\theta(\cdot \mid s)$ and then disintegrating the expectation by $a \sim \pi_\theta(\cdot \mid s)$ gives

$$\begin{aligned} D_{\mathrm{KL}}\left(P_\theta(\cdot \mid s) \,\|\, R\right) = \mathbb{E}_{a\sim\pi_\theta}&\left[\log \frac{d\pi_\theta(a \mid s)}{du_R(a)}\right] \\ &+ \mathbb{E}_{a\sim\pi_\theta}\mathbb{E}_{\tau\sim P_\theta(\cdot\mid s,a)}\left[\log \frac{dP_\theta(\tau \mid s, a)}{dR(\tau \mid a)}\right]. \end{aligned} \tag{33}$$

The first term is $D_{\mathrm{KL}}(\pi_\theta(\cdot \mid s) \,\|\, u_R)$. For each terminal action a, the inner expectation in the second term is $D_{\mathrm{KL}}(P_\theta(\cdot \mid s, a) \,\|\, R(\cdot \mid a))$. This proves Eq. (7). □

A.2 Proof of Theorem 1

Restatement. For fixed s, Q and $\alpha > 0$, optimize Eq. (5) over all path laws $P \ll R$. If

$$Z_Q(s) = \mathbb{E}_{\tau\sim R}\left[\exp\left(Q(s, T(z_K))/\alpha\right)\right] < \infty,$$

then the unique optimum is

$$\frac{dP_Q^\star}{dR}(\tau \mid s) = \frac{\exp\left(Q(s, T(z_K))/\alpha\right)}{Z_Q(s)}.$$

Its terminal action law is

$$\pi_Q^\star(a \mid s) = \frac{u_R(a)\exp\left(Q(s, a)/\alpha\right)}{\int_{\mathcal{A}} u_R(b)\exp\left(Q(s, b)/\alpha\right) db}.$$

Proof. Let $f_s(\tau) = Q(s, T(z_K))$. Since $Z_Q(s) < \infty$, the density in Eq. (9) integrates to one under R, so it defines a valid path law $P_Q^\star$ with $P_Q^\star \ll R$. For any admissible path law $P \ll R$,

$$D_{\mathrm{KL}}\left(P \,\|\, P_Q^\star\right) = \mathbb{E}_{\tau\sim P}\left[\log \frac{dP}{dR}(\tau) - \frac{f_s(\tau)}{\alpha} + \log Z_Q(s)\right]. \tag{34}$$

Rearranging yields the Gibbs variational identity

$$\mathbb{E}_{\tau\sim P}[f_s(\tau)] - \alpha D_{\mathrm{KL}}\left(P \,\|\, R\right) = \alpha \log Z_Q(s) - \alpha D_{\mathrm{KL}}\left(P \,\|\, P_Q^\star\right). \tag{35}$$

The left-hand side is exactly the unrestricted path objective evaluated at P. The right-hand side is upper bounded by $\alpha \log Z_Q(s)$ because relative entropy is nonnegative. The upper bound is attained if and only if $D_{\mathrm{KL}}(P \,\|\, P^\star_Q) = 0$, which holds if and only if $P = P^\star_Q$ almost surely. This proves both optimality and uniqueness.

To compute the terminal marginal, disintegrate the reference as $R(d\tau) = u_R(da)R(d\tau \mid a)$. Since $f_s(\tau) = Q(s,a)$ depends only on the terminal action,

$$P^\star_Q(da \mid s) = \frac{\exp(Q(s,a)/\alpha)}{Z_Q(s)} u_R(da). \tag{36}$$

The normalizer can be written as

$$Z_Q(s) = \int_{\mathcal{A}} u_R(b) \exp(Q(s,b)/\alpha) db, \tag{37}$$

because $\int R(d\tau \mid b) = 1$ for every terminal action b. Substituting this expression gives Eq. (10). □

A.3 Proof of Theorem 2

Restatement. Let

$$Z_Q(s, z_0) = \mathbb{E}_{\tau \sim R(\cdot \mid z_0)} \left[\exp\left(Q(s, T(z_K))/\alpha\right) \right], \qquad \bar{Z}_Q(s) = \mathbb{E}_{z_0 \sim r_0} \left[Z_Q(s, z_0) \right].$$

Among all path laws with initial marginal p_0, the unique optimum is

$$P^\star_{Q,p_0}(d\tau \mid s) = p_0(dz_0) \, \frac{\exp\left(Q(s, T(z_K))/\alpha\right)}{Z_Q(s, z_0)} R(d\tau_{1:K} \mid z_0).$$

Its objective value is

$$\alpha \mathbb{E}_{z_0 \sim p_0} \left[\log Z_Q(s, z_0) \right] - \alpha D_{\mathrm{KL}} \left(p_0 \,\|\, r_0 \right).$$

Proof. Any feasible fixed-base law can be written as

$$P(d\tau \mid s) = p_0(dz_0) P(d\tau_{1:K} \mid z_0, s). \tag{38}$$

The reference has the corresponding decomposition

$$R(d\tau) = r_0(dz_0) R(d\tau_{1:K} \mid z_0). \tag{39}$$

The likelihood ratio therefore separates into an initial-base term and a conditional path term:

$$\log \frac{dP}{dR}(\tau \mid s) = \log \frac{dp_0}{dr_0}(z_0) + \log \frac{dP(\cdot \mid z_0, s)}{dR(\cdot \mid z_0)}(\tau_{1:K}). \tag{40}$$

Taking expectation under P gives

$$D_{\mathrm{KL}}\left(P \,\|\, R\right) = D_{\mathrm{KL}}\left(p_0 \,\|\, r_0\right) + \mathbb{E}_{z_0 \sim p_0} \left[D_{\mathrm{KL}}\left(P(\cdot \mid z_0, s) \,\|\, R(\cdot \mid z_0)\right) \right]. \tag{41}$$

The value term also decomposes conditionally:

$$\mathbb{E}_{\tau \sim P} \left[Q(s, T(z_K)) \right] = \mathbb{E}_{z_0 \sim p_0} \mathbb{E}_{\tau_{1:K} \sim P(\cdot \mid z_0, s)} \left[Q(s, T(z_K)) \right]. \tag{42}$$

Hence the fixed-base objective is an average, over $z_0 \sim p_0$, of independent conditional variational problems minus the initial KL constant. For each fixed z_0, define

$$f_{s,z_0}(\tau_{1:K}) = Q(s, T(z_K)). \tag{43}$$

Applying the Gibbs variational identity to the conditional reference $R(\cdot \mid z_0)$ gives

$$\begin{aligned}\Psi_{z_0}(P) &:= \mathbb{E}_{\tau_{1:K}\sim P(\cdot|z_0,s)}\left[f_{s,z_0}(\tau_{1:K})\right] - \alpha D_{\mathrm{KL}}\left(P(\cdot \mid z_0, s) \,\|\, R(\cdot \mid z_0)\right) \\ &= \alpha \log Z_Q(s, z_0) - \alpha D_{\mathrm{KL}}\left(P(\cdot \mid z_0, s) \,\|\, P^\star(\cdot \mid z_0, s)\right),\end{aligned} \tag{44}$$

where the unique conditional optimizer satisfies

$$\frac{dP^\star(\cdot \mid z_0, s)}{dR(\cdot \mid z_0)}(\tau_{1:K}) = \frac{\exp(Q(s, T(z_K))/\alpha)}{Z_Q(s, z_0)}, \tag{45}$$

and where $Z_Q(s, z_0)$ is the conditional partition function in Theorem 2. The conditional KL term is nonnegative and vanishes only at this optimizer, so the optimizer is unique for p_0-almost every z_0. Averaging the conditional value $\alpha \log Z_Q(s, z_0)$ under p_0 and subtracting $\alpha D_{\mathrm{KL}}(p_0 \,\|\, r_0)$ gives Eq. (14). Substituting the conditional optimizer into the fixed-base disintegration gives Eq. (13). □

A.4 Proof of Corollary 1

Restatement. Under the assumptions of Theorem 2, the loss relative to the unrestricted optimum is

$$\Delta(s) = \alpha\left(\log \bar{Z}_Q(s) - \mathbb{E}_{z_0\sim p_0}\left[\log Z_Q(s, z_0)\right] + D_{\mathrm{KL}}\left(p_0 \,\|\, r_0\right)\right) \geq 0.$$

Equality holds if and only if p_0 equals the initial marginal of the unrestricted tilted bridge. If $p_0 = r_0$, this is equivalent to $Z_Q(s, z_0)$ being constant for p_0-almost every z_0. Moreover,

$$D_{\mathrm{KL}}\left(P^\star_{Q,p_0}(\cdot \mid s) \,\|\, P^\star_Q(\cdot \mid s)\right) = \frac{\Delta(s)}{\alpha},$$

and

$$D_{\mathrm{KL}}\left(\pi^\star_{Q,p_0}(\cdot \mid s) \,\|\, \pi^\star_Q(\cdot \mid s)\right) \leq \frac{\Delta(s)}{\alpha}.$$

Proof. The unrestricted partition function can be decomposed by the reference base law:

$$\begin{aligned}Z_Q(s) &= \mathbb{E}_{\tau\sim R}\left[\exp(Q(s, T(z_K))/\alpha)\right] \\ &= \mathbb{E}_{z_0\sim r_0}\left[Z_Q(s, z_0)\right] \\ &= \bar{Z}_Q(s).\end{aligned} \tag{46}$$

Therefore Theorem 1 gives unrestricted value $\alpha \log \bar{Z}_Q(s)$, while Theorem 2 gives fixed-base value

$$\alpha\mathbb{E}_{z_0\sim p_0}\left[\log Z_Q(s, z_0)\right] - \alpha D_{\mathrm{KL}}\left(p_0 \,\|\, r_0\right). \tag{47}$$

Their difference is Eq. (15). This difference is nonnegative because it is the initial-law KL to the unrestricted tilted bridge. Indeed, the unrestricted tilted bridge has initial marginal

$$P^\star_{Q,0}(dz_0 \mid s) = \frac{Z_Q(s, z_0)}{\bar{Z}_Q(s)} r_0(dz_0). \tag{48}$$

Thus

$$D_{\mathrm{KL}}\left(p_0 \,\|\, P^{\star}_{Q,0}(\cdot \mid s)\right) = \mathbb{E}_{z_0\sim p_0}\left[\log \frac{dp_0}{dr_0}(z_0) + \log \bar{Z}_Q(s) - \log Z_Q(s,z_0)\right] = \frac{\Delta(s)}{\alpha}. \tag{49}$$

This proves nonnegativity and the equality condition. If $p_0 = r_0$, the condition reduces to $Z_Q(s,z_0)$ being constant under p_0.

It remains to relate this gap to the path and endpoint laws. The unrestricted tilted bridge satisfies

$$\frac{dP^{\star}_Q}{dR}(\tau \mid s) = \frac{\exp(Q(s,T(z_K))/\alpha)}{\bar{Z}_Q(s)}, \tag{50}$$

while the fixed-base optimum satisfies

$$\frac{dP^{\star}_{Q,p_0}}{dR}(\tau \mid s) = \frac{dp_0}{dr_0}(z_0)\frac{\exp(Q(s,T(z_K))/\alpha)}{Z_Q(s,z_0)}. \tag{51}$$

Therefore

$$D_{\mathrm{KL}}\left(P^{\star}_{Q,p_0} \,\|\, P^{\star}_Q\right) = \mathbb{E}_{z_0\sim p_0}\left[\log \frac{dp_0}{dr_0}(z_0) + \log \frac{\bar{Z}_Q(s)}{Z_Q(s,z_0)}\right] = \frac{\Delta(s)}{\alpha}. \tag{52}$$

The expectation above is under $P^{\star}_{Q,p_0}$. Its initial marginal is still p_0 by construction, which is why the expectation reduces to $z_0 \sim p_0$. Finally, the endpoint action is the measurable map $\tau \mapsto T(z_K)$. Data processing for relative entropy under this map gives

$$D_{\mathrm{KL}}\left(\pi^{\star}_{Q,p_0}(\cdot \mid s) \,\|\, \pi^{\star}_Q(\cdot \mid s)\right) \le D_{\mathrm{KL}}\left(P^{\star}_{Q,p_0}(\cdot \mid s) \,\|\, P^{\star}_Q(\cdot \mid s)\right), \tag{53}$$

which proves Eq. (17). □

A.5 Proof of Lemma 1

Restatement. Let

$$P_\theta(d\tau \mid s) = p_0(dz_0)\prod_{k=0}^{K-1} q_{\theta,k}(dz_{k+1} \mid z_k, s), \qquad R(d\tau) = r_0(dz_0)\prod_{k=0}^{K-1} r_k(dz_{k+1} \mid z_k).$$

Then

$$D_{\mathrm{KL}}\left(P_\theta(\cdot \mid s) \,\|\, R\right) = D_{\mathrm{KL}}\left(p_0 \,\|\, r_0\right) + \sum_{k=0}^{K-1} \mathbb{E}_{\tau\sim P_\theta}\left[D_{\mathrm{KL}}\left(q_{\theta,k}(\cdot \mid z_k, s) \,\|\, r_k(\cdot \mid z_k)\right)\right].$$

For Gaussian local kernels, if $C_\theta(s,\tau)$ is the sum of the local Gaussian KL terms, then

$$D_{\mathrm{KL}}\left(P_\theta(\cdot \mid s) \,\|\, R\right) = D_{\mathrm{KL}}\left(p_0 \,\|\, r_0\right) + \mathbb{E}_{\tau\sim P_\theta(\cdot|s)}\left[C_\theta(s,\tau)\right].$$

Proof. The path likelihood ratio separates into an initial-base term and local transition terms:

$$\log \frac{dP_\theta(\tau \mid s)}{dR(\tau)} = \log \frac{dp_0}{dr_0}(z_0) + \sum_{k=0}^{K-1} \log \frac{q_{\theta,k}(z_{k+1} \mid z_k, s)}{r_k(z_{k+1} \mid z_k)}. \tag{54}$$

Taking expectation under the actor path law gives the initial contribution

$$\mathbb{E}_{z_0 \sim p_0}\left[\log \frac{dp_0}{dr_0}(z_0)\right] = D_{\mathrm{KL}}\left(p_0 \,\|\, r_0\right). \tag{55}$$

$$\begin{aligned}\mathbb{E}_{\tau \sim P_\theta(\cdot|s)}\left[\log \frac{q_{\theta,k}(z_{k+1} \mid z_k, s)}{r_k(z_{k+1} \mid z_k)}\right] &= \mathbb{E}_{z_k \sim P_{\theta,k}(\cdot|s)} \mathbb{E}_{z_{k+1} \sim q_{\theta,k}(\cdot|z_k,s)}\left[\log \frac{q_{\theta,k}(z_{k+1} \mid z_k, s)}{r_k(z_{k+1} \mid z_k)}\right] \\ &= \mathbb{E}_{z_k \sim P_{\theta,k}(\cdot|s)}\left[D_{\mathrm{KL}}\left(q_{\theta,k}(\cdot \mid z_k, s) \,\|\, r_k(\cdot \mid z_k)\right)\right], \end{aligned} \tag{56}$$

where $P_{\theta,k}(\cdot \mid s)$ is the actor marginal law of z_k. Summing this identity over k gives Eq. (22). If the local kernels are Gaussian, each local KL has a closed form. Defining $C_\theta(s,\tau)$ as the sum of those local Gaussian KLs gives Eq. (23). □

A.6 Proof of Proposition 2

Restatement. Let $P_Q^\star$ be the unrestricted soft-optimal bridge in Theorem 1. For any finite-step bridge actor with fixed base law $p_0 \ll r_0$ satisfying Lemma 1,

$$\mathcal{L}_{\mathrm{actor}}(\theta \mid s) = \mathbb{E}_{\tau \sim P_\theta(\cdot|s)}\left[\alpha\, C_\theta(s,\tau) - Q(s, T(z_K))\right]$$

satisfies

$$\mathcal{L}_{\mathrm{actor}}(\theta \mid s) = \alpha D_{\mathrm{KL}}\left(P_\theta(\cdot \mid s) \,\|\, P_Q^\star(\cdot \mid s)\right) - \alpha \log Z_Q(s) - \alpha D_{\mathrm{KL}}\left(p_0 \,\|\, r_0\right).$$

Consequently, minimizing this loss over the unrestricted fixed-base path-law family has global minimizer $P^\star_{Q,p_0}$.

Proof. Fix a state s. By Lemma 1, the sampled control energy satisfies

$$\mathbb{E}_{\tau \sim P_\theta(\cdot|s)}\left[C_\theta(s,\tau)\right] = D_{\mathrm{KL}}\left(P_\theta(\cdot \mid s) \,\|\, R\right) - D_{\mathrm{KL}}\left(p_0 \,\|\, r_0\right). \tag{57}$$

The unrestricted soft-optimal bridge from Theorem 1 has likelihood ratio

$$\log \frac{dP_Q^\star}{dR}(\tau \mid s) = \frac{Q(s, T(z_K))}{\alpha} - \log Z_Q(s). \tag{58}$$

Expanding the reverse KL to this ideal bridge gives

$$\begin{aligned} &\alpha D_{\mathrm{KL}}\left(P_\theta(\cdot \mid s) \,\|\, P_Q^\star(\cdot \mid s)\right) \\ &= \alpha \mathbb{E}_{\tau \sim P_\theta(\cdot|s)}\left[\log \frac{dP_\theta}{dR}(\tau \mid s) - \frac{Q(s, T(z_K))}{\alpha} + \log Z_Q(s)\right] \\ &= \alpha D_{\mathrm{KL}}\left(P_\theta(\cdot \mid s) \,\|\, R\right) - \mathbb{E}_{\tau \sim P_\theta}\left[Q(s, T(z_K))\right] + \alpha \log Z_Q(s) \\ &= \mathcal{L}_{\mathrm{actor}}(\theta \mid s) + \alpha \log Z_Q(s) + \alpha D_{\mathrm{KL}}\left(p_0 \,\|\, r_0\right). \end{aligned} \tag{59}$$

Rearranging gives Eq. (30). Since the projection is optimized over the unrestricted fixed-base family, Theorem 2 identifies the global minimizer as $P^\star_{Q,p_0}$. The finite neural Gaussian Markov actor used by the algorithm is a restricted parameterization of this family, so the proof gives the population projection target rather than a global optimization guarantee for SGD. □

B Algorithm Pseudocode

Algorithm 1 `SoftGAC` off-policy training

Require: replay buffer $\mathcal{D}$, actor P_θ, target actor $P_{\bar{\theta}}$, critic Q_ϕ, target critic $Q_{\bar{\phi}}$, temperature α, target cost C_{target}, discount γ, Polyak coefficient ρ

initialize $\mathcal{D}$, θ, ϕ, $\bar{\theta} \leftarrow \theta$, $\bar{\phi} \leftarrow \phi$
for each environment step **do**
  sample a bridge path $\tau \sim P_\theta(\cdot \mid s)$ and execute $a = T(z_K)$
  store (s, a, r, s', d) in $\mathcal{D}$
  for each gradient update **do**
    sample a replay batch $(s, a, r, s', d) \sim \mathcal{D}$
    sample next paths $\tau' \sim P_{\bar{\theta}}(\cdot \mid s')$
    compute next actions $a' = T(z'_K)$ and bridge costs $C_{\bar{\theta}}(s', \tau')$
    build soft targets $y = r + \gamma(1-d)\left[Q^{\min}_{\bar{\phi}}(s', a') - \alpha C_{\bar{\theta}}(s', \tau')\right]$
    update Q_ϕ toward y with the chosen off-policy critic loss
    if the policy-delay interval is reached **then**
      sample current paths $\tau \sim P_\theta(\cdot \mid s)$
      compute current actions $\tilde{a} = T(z_K)$ and bridge costs $C_\theta(s, \tau)$
      update θ to minimize $\mathbb{E}[\alpha C_\theta(s, \tau) - Q^{\min}_\phi(s, \tilde{a})]$
      update $\log \alpha$ to minimize $\mathbb{E}[\alpha(C_{\text{target}} - C_\theta(s, \tau))]$ with $\alpha = \exp(\log \alpha)$
    end if
    if using target networks and the target-update interval is reached **then**
      soft-update target networks $\bar{\theta} \leftarrow \rho\bar{\theta} + (1-\rho)\theta$ and $\bar{\phi} \leftarrow \rho\bar{\phi} + (1-\rho)\phi$
    end if
  end for
end for

C 2D Bridge Visualization

Figure 5 visualizes the bridge mechanism in a controlled two-dimensional fixed-critic setting. This diagnostic setting isolates the actor update by training a bridge policy against a hand-designed multimodal value function. It makes the finite-step path distribution visible as it moves from the base law to the terminal action law under different control-energy budgets.

The bounded action is $a = (a_1, a_2) \in (-1, 1)^2$. We define an unnormalized fixed critic as a log-sum-exp mixture of three Gaussian-like modes,

$$Q_{\text{raw}}(a) = \log \sum_{m=1}^{3} w_m \exp\left(-\frac{1}{2}\left\|\frac{a - c_m}{\sigma_m}\right\|_2^2\right), \tag{60}$$

with centers

$$c_1 = (-0.68, 0.50), \qquad c_2 = (0.62, 0.50), \qquad c_3 = (-0.06, -0.58), \tag{61}$$

axis-wise widths

$$\sigma_1 = (0.24, 0.24), \qquad \sigma_2 = (0.24, 0.24), \qquad \sigma_3 = (0.34, 0.32), \tag{62}$$

and weights $(w_1, w_2, w_3) = (0.90, 1.00, 0.70)$. We normalize the critic to $[0, 1]$ over a dense action grid and use this normalized value as $Q(s, a)$ for a single dummy state s. The latent landscape shown in the second column is the pullback $Q(s, \tanh z)$.

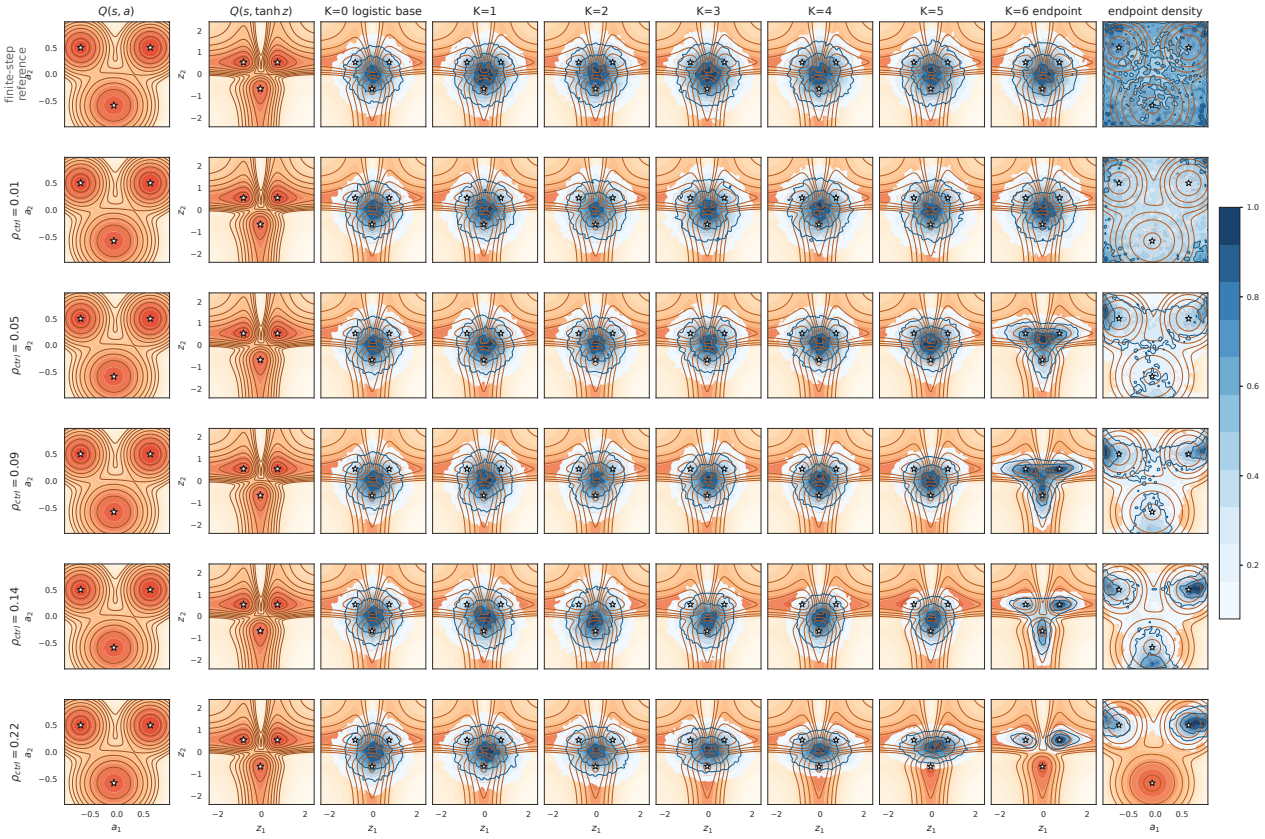

Figure 5. 2D bridge visualization. Each row corresponds to a different control-energy budget. Columns show the action-space critic, the corresponding pre-tanh latent critic, intermediate latent bridge densities and the final action-space policy density.

The reference process is the same finite-step reference used in the method section. We sample the base action from an approximately uniform law on $[-0.995, 0.995]^2$ and set $z_0 = \operatorname{arctanh}(a_0)$. The reference latent process uses $K = 6$ Euler steps with $h = 1/K$,

$$z_{k+1} = z_k - 2h \tanh(z_k) + \sqrt{2h}\,\epsilon_k, \qquad \epsilon_k \sim \mathcal{N}(0, I). \tag{63}$$

This finite chain approximates the continuous reference whose stationary action law after tanh is uniform. We intentionally show the finite-step reference rather than the continuous-limit uniform distribution because the algorithm optimizes the finite-step path KL.

For each control-energy budget, we train a separate bridge actor with $K = 6$ action-dimensional latent transitions. Given a sampled logistic reference base latent and Gaussian transition noise, the actor produces a path $\tau = (z_0, \ldots, z_K)$ and terminal action $a = \tanh(z_K)$. We optimize the sampled actor objective

$$\mathbb{E}_{\tau \sim P_\theta}\left[\alpha\, C_\theta(s, \tau) - Q(s, \tanh z_K)\right], \tag{64}$$

where C_θ is the finite-step local Gaussian control energy from Lemma 1. The temperature is tuned with the same budget form as the main algorithm,

$$C_{\text{target}} = \rho_{\text{ctrl}} K d_a, \qquad d_a = 2. \tag{65}$$

The rows in Figure 5 use

$$\rho_{\text{ctrl}} \in \{0.01, 0.05, 0.09, 0.14, 0.22\}. \tag{66}$$

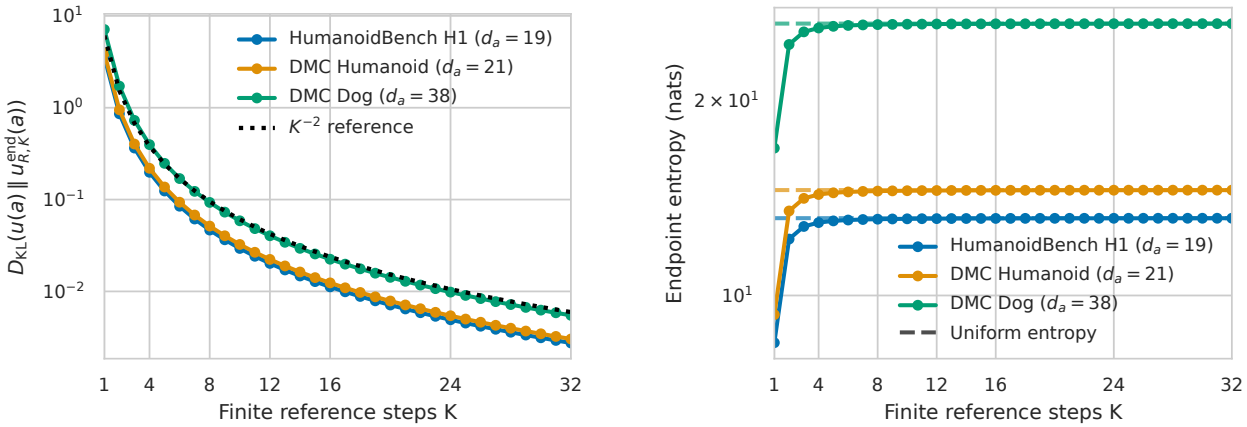

Figure 6. Finite-step reference endpoint bias. Left shows the endpoint action-marginal KL from the ideal uniform prior to the finite-step reference endpoint law. Right shows the finite-step endpoint entropy and the corresponding uniform entropy baselines. The three curves use the action dimensions of HumanoidBench H1, DMC Humanoid and DMC Dog.

Small budgets keep the actor close to the reference bridge and preserve broad coverage. Larger budgets allow stronger value guidance and concentrate the endpoint density near higher-value modes. The first two columns show $Q(s, a)$ in action space and $Q(s, \tanh z)$ in pre-tanh latent space. The next columns show latent densities at bridge steps $K = 0, \ldots, 6$. The final column maps terminal latents back to action space and overlays the resulting policy density on the action-space critic.

D Finite-Step Reference Endpoint Bias

In this paper, we use a finite-step reference bridge to approximate the ideal continuous reference whose endpoint action law is uniform. This approximation affects only the endpoint marginal law used as the reference prior. It is distinct from the path-wise KL used by the actor update, which remains exact for the chosen finite-step actor-reference pair. We quantify this endpoint effect in Figure 6.

Let $q_{\mathrm{ref}}(z) = \frac{1}{2}\mathrm{sech}^2(z)$ be the one-dimensional pre-tanh density whose image under tanh is uniform. For a reference bridge with K Euler steps, let p_K be the one-dimensional terminal latent density produced by

$$z_{k+1} = z_k - 2h\tanh(z_k) + \sqrt{2h}\,\epsilon_k, \qquad h = \frac{1}{K}. \tag{67}$$

The map tanh is bijective between latent space and the normalized action interval, so endpoint KL values can be computed in latent space without estimating an action-space density. For example,

$$D_{\mathrm{KL}}\left(u(a) \,\middle\|\, u^{\mathrm{end}}_{R,K}(a)\right) = D_{\mathrm{KL}}(q_{\mathrm{ref}}(z) \,\|\, p_K(z))\,. \tag{68}$$

The implemented reference factorizes across action dimensions, so the d_a-dimensional laws are product measures and relative entropy is additive across coordinates. It is therefore enough to compute the one-dimensional terminal density and multiply the resulting KL by d_a. Let $\mathcal{T}_h$ be the one-dimensional Euler transition operator,

$$(\mathcal{T}_h f)(z') = \int \mathcal{N}(z' \mid z - 2h\tanh z, 2h)\, f(z)\, dz, \qquad h = \frac{1}{K}. \tag{69}$$

Then $p_K = \mathcal{T}_h^K q_{\mathrm{ref}}$, and the d_a-dimensional endpoint-prior gap has the exact finite-K form

$$G_K(d_a) = D_{\mathrm{KL}}\left(u(a) \,\middle\|\, u^{\mathrm{end}}_{R,K}(a)\right) = d_a \int q_{\mathrm{ref}}(z) \log \frac{q_{\mathrm{ref}}(z)}{p_K(z)} dz. \tag{70}$$

Thus the dependence on d_a is exactly linear. The dependence on K comes only from the Euler discretization error in p_K. Under the standard first-order density expansion for Euler discretization over a fixed time horizon, $p_K(z) = q_{\mathrm{ref}}(z)(1 + K^{-1}r(z) + O(K^{-2}))$ with $\int q_{\mathrm{ref}}(z)r(z)\,dz = 0$. The first-order term cancels in the KL because of normalization, so the leading contribution is quadratic in the density error. Substituting this expansion into Eq. (70) gives

$$G_K(d_a) = \frac{d_a}{2K^2}\int q_{\mathrm{ref}}(z)r(z)^2\,dz + O\left(\frac{d_a}{K^3}\right). \tag{71}$$

Thus the leading endpoint-prior gap scales as $O(d_a/K^2)$ whenever this expansion holds. This rate explains the observed finite-step bias but is not an assumption used by the algorithm. The entropy deficit has the same leading rate because it equals $D_{\mathrm{KL}}(u^{\mathrm{end}}_{R,K}\|u)$. Likewise, the endpoint entropy satisfies

$$H\left(u^{\mathrm{end}}_{R,K}\right) = d_a \log 2 - D_{\mathrm{KL}}\left(u^{\mathrm{end}}_{R,K}\,\middle\|\,u\right), \tag{72}$$

where $d_a \log 2$ is the d_a-dimensional uniform endpoint entropy. We compute the one-dimensional density deterministically on a dense grid by applying the Gaussian transition kernel, then report the corresponding d_a-dimensional quantities for the action dimensions used in our experiments.

The bias decreases quickly with K. At the default $K = 6$, the endpoint KL $D_{\mathrm{KL}}(u \,\|\, u^{\mathrm{end}}_{R,K})$ is approximately 0.085, 0.094 and 0.169 nats for $d_a = 19$, 21 and 38. The endpoint entropy is already close to the uniform entropy in all three domains, while the network is still shallow to optimize. This supports the interpretation that the finite reference mainly provides an implementable high-entropy prior, while the algorithm optimizes the exact finite-step path regularizer defined by that prior.

E Detailed Experimental Setup

E.1 Environment and Task Domains

Our benchmark contains 12 continuous-control tasks from three domains. Figure 7 shows representative rendered states. All experiments use vector observations and continuous bounded actions. The reported dimensions are the observation and action dimensions after the environment wrappers used by our training code.

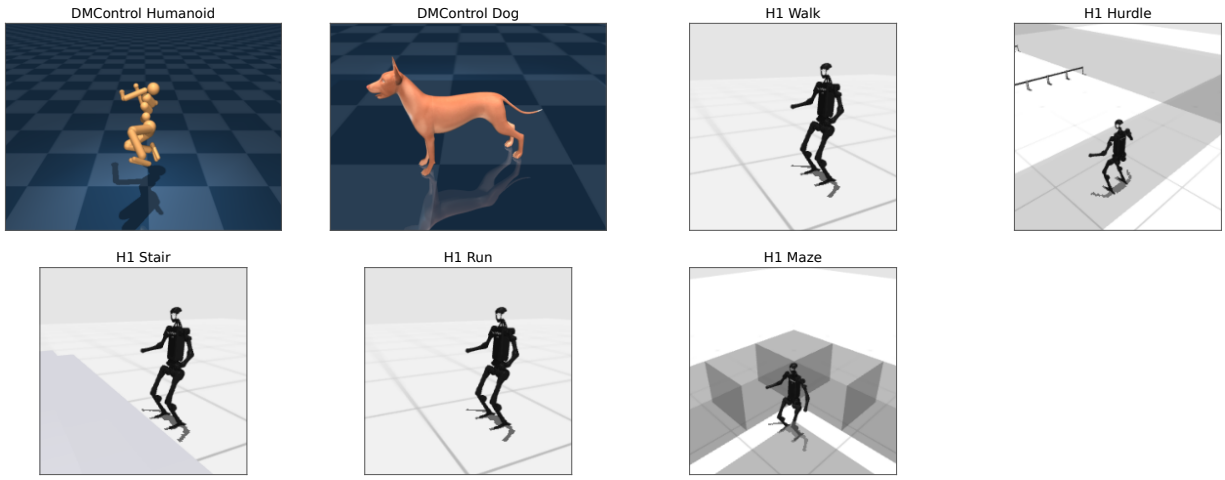

Figure 7. Representative screenshots of the benchmark domains and H1 tasks.

Table 1. Benchmark task dimensions.

Domain	Task	Observation dim.	Action dim.
DMC Humanoid	`dm_control/humanoid-run`	67	21
DMC Humanoid	`dm_control/humanoid-walk`	67	21
DMC Humanoid	`dm_control/humanoid-stand`	67	21
DMC Dog	`dm_control/dog-run`	223	38
DMC Dog	`dm_control/dog-trot`	223	38
DMC Dog	`dm_control/dog-walk`	223	38
DMC Dog	`dm_control/dog-stand`	223	38
HumanoidBench H1	`h1-walk-v0`	51	19
HumanoidBench H1	`h1-hurdle-v0`	51	19
HumanoidBench H1	`h1-stair-v0`	51	19
HumanoidBench H1	`h1-run-v0`	51	19
HumanoidBench H1	`h1-maze-v0`	51	19

DMC Humanoid evaluates high-dimensional biped locomotion with running, walking and standing objectives. DMC Dog evaluates quadruped control with a larger observation and action space, including fast running, trotting, walking and standing. HumanoidBench H1 uses a humanoid robot with 19 actuated degrees of freedom. The walk and run tasks measure whole-body locomotion, while hurdle, stair and maze add obstacle negotiation, contact-rich terrain interaction and navigation constraints.

Benchmark assets and licenses. We use the DeepMind Control Suite through `dm_control` version 1.0.39 from https://github.com/google-deepmind/dm_control, which is released under the Apache License 2.0. We use HumanoidBench through the official repository's main branch from https://github.com/carlosferrazza/humanoid-bench, which is released under the MIT License and includes third-party notices in its repository license file.

Baseline implementation references. All baseline results are produced by our unified JAX codebase. We use the official repositories in Table 2 as implementation references, then re-implement the algorithm-specific actor, target construction and auxiliary losses in JAX inside our training stack. This gives each method the same logging, replay, evaluation and experiment infrastructure while preserving the algorithm-specific components required by the original implementations.

Table 2. Official implementation repositories used as baseline references.

Method	Official repository
FLAC	https://github.com/bytedance/FLAC
DIME	https://github.com/ALRhub/DIME
FlowRL	https://github.com/bytedance/FlowRL
QSM	https://github.com/escontra/score_matching_rl
QVPO	https://github.com/wadx2019/qvpo
CrossQ-SAC	https://github.com/adityab/CrossQ

E.2 Training Hardware

Main training runs were executed on Google Cloud TPU v4-16/v6e-8 VMs. On the v6e hosts used for our experiments, Linux reports two AMD EPYC 9B14 sockets with 90 cores per socket, 180 logical CPUs, one thread per core and approximately 1.4 TiB of system memory. The machines run Ubuntu Linux with kernel 6.8 on x86_64 CPUs. We use the same hardware class for the reported TPU experiments unless otherwise noted, and Appendix F reports representative wall-clock curves for difficult tasks. This hardware was used to run many tasks and seeds in parallel, not because the model itself requires unusually large compute. The actor and critic networks are lightweight by modern deep RL standards. Under the reported hyperparameters, a consumer GPU such as an RTX 2080 Ti-class device with 32 GB of host memory is sufficient to train these agents on the benchmark tasks, although wall-clock time will be longer than on our TPU cluster.

E.3 Implementation Details

This subsection describes the concrete network and update implementation used for `SoftGAC`. The actor operates in pre-tanh latent space and samples a tensor of noises with shape $B \times (K+1) \times d_a$. The first slice gives the base latent z_0. In the main runs, we use the logistic base by sampling $a_0 \sim \mathrm{Unif}((-1,1)^{d_a})$ and setting $z_0 = \mathrm{arctanh}(a_0)$. The actor forward pass is:

```
def bridge_actor(obs, noise):
    z = noise[:, 0]                          # base latent z_0
    latents, drifts, sigmas = [z], [], []
    h = 1.0 / K
    for k in range(K):
        x = concat(obs, z)
        x = LayerNorm(x)
        x = Dense(hidden_size)(x)
        x = elu(x)
        x = LayerNorm(x)
        drift = Dense(action_dim)(x)
        sigma = softplus(Dense(action_dim)(x))
        z = z + h * drift + sqrt(2 * h) * sigma * noise[:, k + 1]
        latents.append(z)
        drifts.append(drift)
```

```
        sigmas.append(sigma)
    action = action_scale * tanh(z) + action_bias
    return action, latents, drifts, sigmas
```

Each block uses one hidden layer with layer normalization before and after the ELU activation, followed by drift and positive diagonal-scale heads. Main runs use $K = 6$ blocks and width 512, with width 256 for DMC Dog to keep the actor parameter budget close to the baselines. The actor update minimizes $\alpha C_\theta(s, \tau) - Q_\phi^{\min}(s, a)$. The control energy C_θ is the accumulated closed-form Gaussian transition KL to the reference bridge, so no endpoint entropy estimator is used. The temperature $\alpha = \exp(\log \alpha)$ is tuned toward $\rho_{\text{ctrl}} K d_a$, with default $\rho_{\text{ctrl}} = 0.2$. The main critic is the same twin C51 critic with CrossQ-style update adopted in DIME [19].

E.4 Choosing the Control-Energy Budget

The target control-energy budget is meant to set how strongly the value term can move the bridge away from the high-entropy reference, not to introduce a task-specific objective. A useful way to choose its scale is to ask how far one action dimension should be allowed to move under value-guided control. The path KL factorizes over K transitions and d_a action dimensions, so we set

$$C_{\text{target}} = \rho_{\text{ctrl}} K d_a. \tag{73}$$

This makes ρ_{ctrl} the average local KL budget, measured in nats, per transition and per action dimension. Equivalently, each action dimension receives a total path budget of $\rho_{\text{ctrl}} K$ along the full bridge.

This scale has a simple endpoint interpretation. Ignore the learned variance term for a moment and assume that the actor and reference transitions share covariance $2hI$, where $h = 1/K$. If the actor adds a one-dimensional mean control δz_k at step k, the local KL contribution is

$$D_{\text{KL}}\left(\mathcal{N}(\mu_R + \delta z_k, 2h) \,\|\, \mathcal{N}(\mu_R, 2h)\right) = \frac{(\delta z_k)^2}{4h}. \tag{74}$$

For a desired terminal displacement Δz in one pre-tanh action dimension, the cheapest constant-speed control has $\delta z_k \approx \Delta z / K$. The accumulated control energy is then

$$\sum_{k=0}^{K-1} \frac{(\Delta z/K)^2}{4h} = \frac{(\Delta z)^2}{4}. \tag{75}$$

Thus a budget $\rho_{\text{ctrl}} K$ per action dimension roughly permits a terminal pre-tanh displacement

$$|\Delta z| \approx 2\sqrt{\rho_{\text{ctrl}} K}. \tag{76}$$

If we want the actor to be able to push a dimension close to the saturated action region, say $|a| \approx 0.95$ to 0.98, the corresponding pre-tanh scale is $z_{\text{sat}} = \operatorname{arctanh}(|a|) \approx 1.8$ to 2.3. Matching $\rho_{\text{ctrl}} K \approx z_{\text{sat}}^2/4$ gives $\rho_{\text{ctrl}} \approx z_{\text{sat}}^2/(4K)$. For the main bridge depth $K = 6$, this range is about 0.14 to 0.22. We therefore use $\rho_{\text{ctrl}} = 0.2$ as a moderate default. It allows value-guided transport toward near-saturated high-value actions, but still assigns a visible cost to collapse away from the broad reference. Smaller budgets keep broader action coverage, while larger budgets permit sharper value-guided concentration. This is the same qualitative trend visualized in Appendix C.

E.5 Hyperparameters

Table 3 reports the actor and main critic parameter counts by domain, and Table 4 summarizes the main hyperparameters. All methods use 8 seeds and Adam optimizers. To reduce critic-side confounding, we use the

same twin C51 main critic and CrossQ-style main critic update adopted in DIME [19] across the compared methods. When an algorithm includes additional auxiliary networks, such as FlowRL's buffer critic, we keep the corresponding architecture and update rule from the official implementation.

Table 3. Actor and main critic parameter counts in the experiment domains, in millions of parameters.

	DMC Humanoid		DMC Dog		HumanoidBench	
Algorithm	Actor	Critic	Actor	Critic	Actor	Critic
`SoftGAC`	0.410	9.19	0.526	9.90	0.342	9.11
FLAC	0.322	9.19	0.419	9.90	0.311	9.11
DIME	0.423	9.19	0.472	9.90	0.418	9.11
FlowRL	0.322	9.19	0.419	9.90	0.311	9.11
QSM	0.614	9.19	0.712	9.90	0.604	9.11
QVPO	0.369	9.19	0.466	9.90	0.359	9.11
CrossQ-SAC	0.321	9.19	0.419	9.90	0.311	9.11

Table 4. Main hyperparameters used in the experiments.

Hyperparameter	`SoftGAC`	FLAC	DIME	FlowRL	QSM	QVPO	CrossQ-SAC
UTD ratio	2	2	2	2	2	2	2
Discount	0.99	0.99	0.99	0.99	0.99	0.99	0.99
Batch size	256	256	256	256	256	256	256
Buffer size	10^6	10^6	10^6	10^6	10^6	10^6	10^6
Learn starts	5000	5000	5000	5000	5000	5000	5000
Actor lr	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}
Critic lr	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}	3×10^{-4}
Policy delay	2	2	2	2	2	2	2
Regularizer	path KL	kinetic	entropy bd.	CFM	score-Q	weighted-Q	entropy
Target entropy/energy	$0.2Kd_a$	$0.5d_a$	$4d_a$	N/A	N/A	N/A	$-d_a$
Temp. lr	10^{-3}	$3\cdot10^{-5}$	10^{-3}	N/A	N/A	N/A	10^{-3}
Critic depth	2	2	2	2	2	2	2
Critic hidden size	2048	2048	2048	2048	2048	2048	2048
Num. bins	101	101	101	101	101	101	101
Actor depth	6	2	3	2	2	2	2
Actor width	256/512[†]	512	256	512	512	512	512
Multi-modal	yes	yes	yes	yes	yes	yes	no
Prior dist.	Logistic	clip $\mathcal{N}(0,I)$	$\mathcal{N}(0,2.5^2I)$	clip $\mathcal{N}(0,I)$	$\mathcal{N}(0,I)$	$\mathcal{N}(0,I)$	$\mathcal{N}(0,I)$
Iteration	1	2	16	2	5	16	1

[†] `SoftGAC` uses width 256 only for DMC Dog and width 512 otherwise for controlling the parameter count.

F Extended Experimental Results and Ablations

We provide more detailed results in this section. Figures 8 and 9 report the full 12-task learning curves and final IQM returns, which evaluate the sample-efficiency of each algorithm. Figure 10 shows the per-task compute-return tradeoff using actor inference time, which gives a more direct efficiency comparison than training wall-clock time. Figure 11 gives supplementary wall-clock views on Humanoid Run and Dog Run using median seed time at each evaluation step. Figures 12, 14 and 13 report actor-width, bridge-depth and control-budget sensitivity.

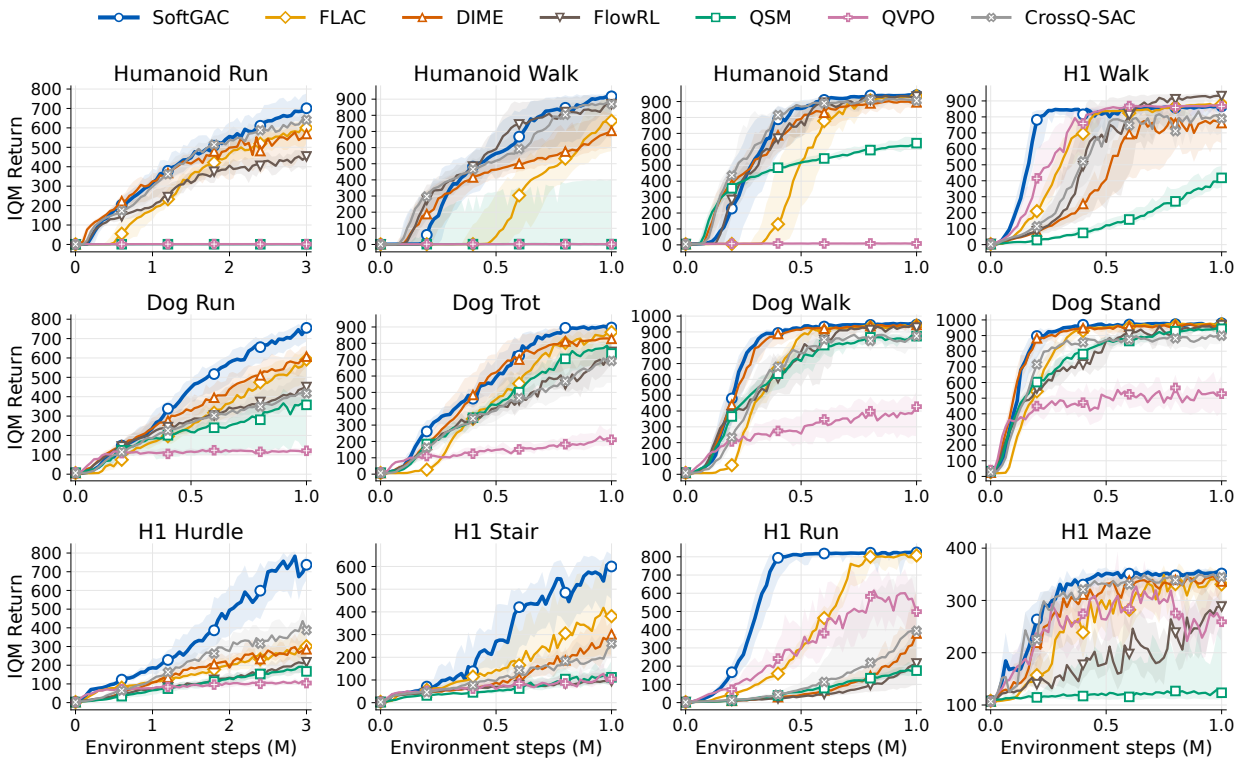

Figure 8. Full IQM learning curves on all benchmark tasks.

The full learning curves and final IQM summaries support the same conclusion. The main hard-task trends are not caused by a small task subset. `SoftGAC` gives considerable gains on Humanoid Run, Dog Run, H1 Hurdle, H1 Stair and H1 Run, where high-dimensional locomotion and long-horizon credit assignment make the actor design important. On easier stand or walk tasks, several algorithms eventually reach strong returns, so the main difference is often sample efficiency rather than final solvability. The final bars report the IQM of the last plotted evaluation points. The curves show when the advantage appears during training.

The compute-return plots make the Pareto structure explicit. CrossQ-SAC has the cheapest Gaussian actor but lacks the multimodal generative policy class. DIME, QSM and QVPO sit in a higher-latency region because they evaluate iterative diffusion-style actors. `SoftGAC` occupies a favorable region across the hard tasks because its actor remains close to one-pass inference cost while achieving higher or competitive IQM return.

The wall-clock curves provide a complementary but noisier view of the same result on three difficult tasks. Training wall-clock time depends on system scheduling, host load, TPU availability and run placement, and online RL also spends substantial time in CPU-side environment interaction. We therefore treat wall-clock as supporting evidence rather than the primary efficiency metric. We use median seed wall-clock at each environment step to reduce sensitivity to stragglers and system noise. Under this measurement, `SoftGAC` still reaches high return within a wall-clock budget comparable to low-NFE flow baselines, while DIME and QVPO pay the cost of a many-step diffusion actor.

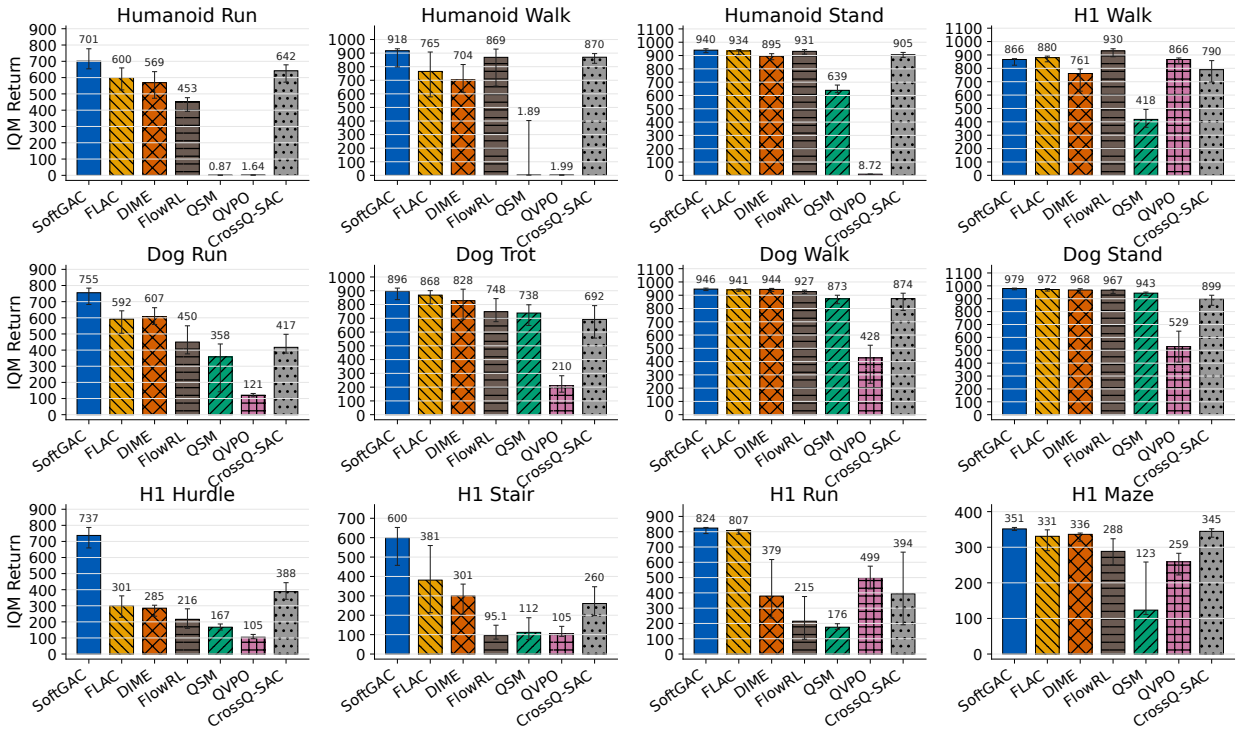

Figure 9. Final IQM return by task.

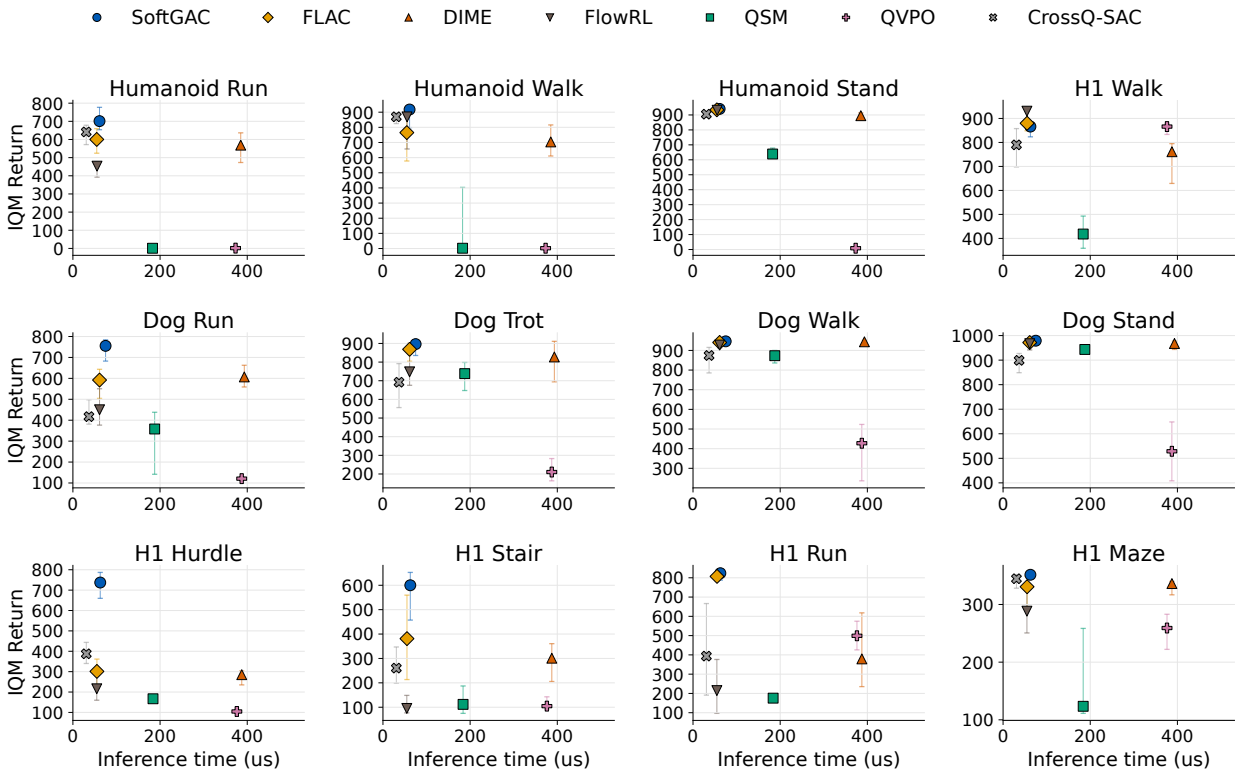

Figure 10. Compute-return tradeoff by task. The horizontal axis is per-action actor inference time, and the vertical axis is IQM return.

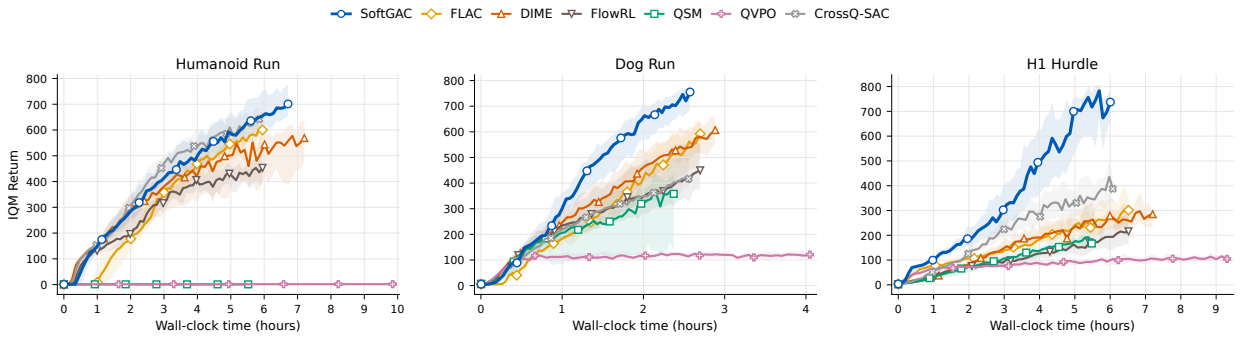

Figure 11. Wall-clock IQM learning curves on Humanoid Run, Dog Run and H1 Hurdle. The horizontal axis uses median seed wall-clock time at each evaluation step. All runs use the v6e-8 TPU VM, but the actual time depends on system scheduling and load.

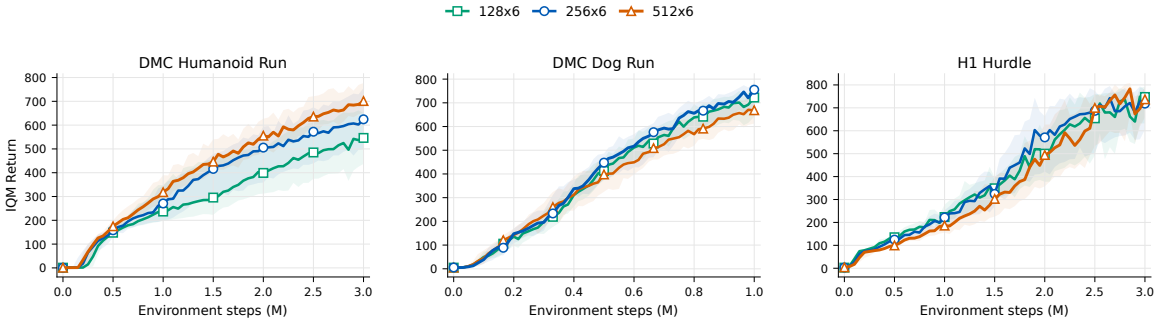

Figure 12. Actor-width sensitivity on Humanoid Run, Dog Run and H1 Hurdle.

The width-sensitivity curves show that the selected actor sizes are not a fragile single setting. Increasing width generally improves or stabilizes learning on Humanoid Run and H1 Hurdle. On Dog Run, the smallest 128-wide bridge already gives strong performance under a lower parameter budget, while wider actors do not provide a consistent gain. The main runs use width 512 for all tasks except Dog, where we use width 256 to keep the parameter count close to the baselines. The sensitivity curves suggest that fine-tuning actor could give a further boost on some tasks, but our goal is to verify the effectiveness of `SoftGAC` under a comparable parameter budget.

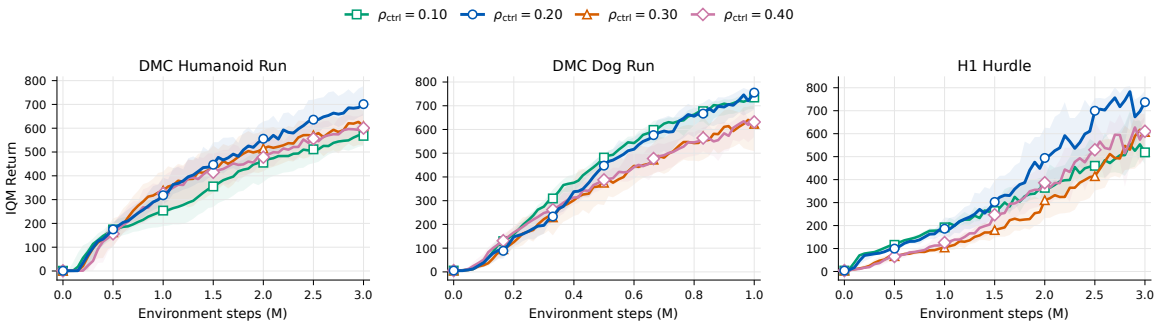

Figure 13. Target control-budget sensitivity on Humanoid Run, Dog Run and H1 Hurdle.

The control-budget sensitivity shows the expected tradeoff. Small ρ_{ctrl} keeps the bridge close to the high-entropy reference and can limit value-guided concentration. Larger budgets allow stronger control, but overly large values do not always improve returns. The default $\rho_{\text{ctrl}} = 0.20$ is therefore a stable middle setting across these three domains rather than a value tuned to a single task, as discussed in Appendix E.4. Overall, the sensitivity curves suggest that the algorithm is not fragile to this hyperparameter, and that tuning it could give a further boost on some tasks.

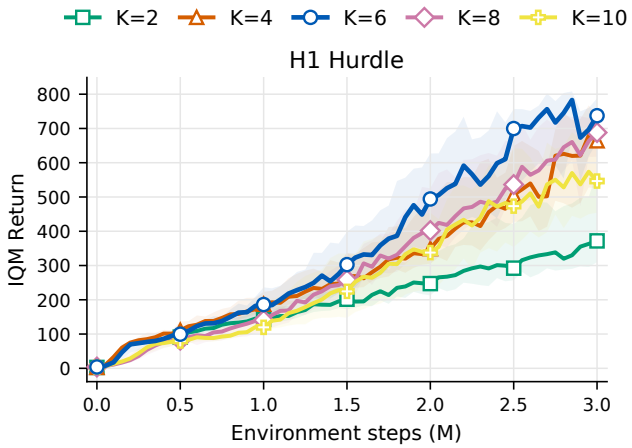

Figure 14. Bridge-depth sensitivity on H1 Hurdle with $K \in \{2, 4, 6, 8, 10\}$.

The bridge-depth ablation complements the width study by varying the number of local transition blocks while keeping the width fixed. On H1 Hurdle, moving from $K = 2$ to $K = 4$ gives a clear improvement, and $K = 6$ gives the best final performance in this sweep. This supports the role of a short but nontrivial path structure. With too few bridge steps, the actor has less room for value-guided transport and the finite-step reference has a larger endpoint bias. The default $K = 6$ provides enough intermediate structure without turning the actor into a long iterative sampler. Increasing the depth beyond $K = 6$ gives marginal or inconsistent gains on this task, possibly because deeper bridges are harder to optimize. We therefore use $K = 6$ in the main runs. Deeper bridges may benefit from more stable training tricks and more trainable transition architectures, which we view as future work rather than the focus of this paper. Depth sensitivity can vary across tasks, but this ablation supports the broader principle that a reasonably short bridge can improve substantially over one-step transport while keeping inference efficient.