# Goal Achievement Guided Exploitation: Rethinking Maximum Entropy Reinforcement Learning

**Anonymous authors**
**Paper under double-blind review**

## Abstract

Reinforcement learning (RL) algorithms often rely on entropy maximization to prevent premature convergence, yet this practice introduces fundamental drawbacks: it alters the optimization objective and cannot guarantee sufficient exploration in some tasks with local optima. We propose Goal Achievement Guided Exploitation (GAGE), a principled alternative that adaptively regulates exploration based on the agent's performance relative to the optimal goal. Instead of maximizing entropy, GAGE enforces hard lower bounds on policy flatness, represented by the standard deviation in continuous actions and the logit range in discrete ones, providing interpretable and controllable exploration without modifying the reward function. This mechanism ensures lower-bounds of action probabilities and naturally reduces stochasticity as learning progresses. Across a suite of challenging robotic control tasks with severe local optima, GAGE consistently improves stability, robustness, and final per formance over entropy-based baselines for both on-policy and off-policy algorithms by a clear margin. Our results suggest that performance-guided exploration offers a scalable and interpretable direction beyond the maximum-entropy paradigm in reinforcement learning.

## 1 Introduction

Local optima in the objective landscape pose a fundamental challenge for gradient-based optimization (Chaudhari et al., 2017), especially with dense rewards. In reinforcement learning (RL), this problem is amplified by the trial-and-error nature, where the agent predominantly learns from suboptimal trajectories. As a result, RL algorithms are often highly sensitive to hyperparameters and prone to premature convergence to arbitrary local optima.

To mitigate premature convergence, the maximum entropy reinforcement learning (MaxEnt RL) framework has become a foundational approach in modern RL. By augmenting the standard reward maximization objective with an entropy term that encourages policy stochasticity, MaxEnt RL promotes exploration and often yields more stable and robust policies (Williams & Peng, 1991; Mnih et al., 2016; Schulman et al., 2017; Haarnoja et al., 2018a). Despite its success, MaxEnt RL exhibits fundamental limitations that undermine learning stability and performance Zhang et al. (2025). First, the entropy term modifies the original objective and can shift the optimal solution, requiring carefully designed annealing schedules to neutralize its influence at convergence. Second, as a soft regularizer and lossy summary statistic, entropy provides only an indirect and non-intuitive means of controlling action probabilities, making it difficult to ensure sufficient exploration throughout training. These issues complicate algorithm design and hyperparameter tuning. For example, the widely used algorithm Soft Actor-Critic (SAC-v2) (Haarnoja et al., 2018b) maintains a fixed target entropy as a practical compromise even though in practice we often desire the agent to dynamically adjust its exploration-exploitation balance across different learning stages.

Despite various recent advances in exploration strategies built upon the MaxEnt RL framework (Ladosz et al., 2022; Wang et al., 2023; Wan et al., 2023; Yan et al., 2024b; Sukhija et al., 2025), we argue that it is time to critically reassess the inherent limitations of this foundational paradigm and explore alternative solutions. In this work, we propose a novel approach that incoporates the agent's *goal achievement*, defined as the ratio between its current performance and the optimal value, into the exploration-exploitation strategy.

By leveraging this signal, our method adaptively reduces exploration as the agent approaches its desired performance, allowing the exploration rate to evolve naturally with learning progress rather than following a manually specified schedule. This is achieved by imposing a hard constraint on the action distribution through an adaptive lower bound on its flatness, which is negatively correlated with the goal achievement.

We implement our approach across both continuous and discrete action spaces, as well as on- and off-policy algorithms. Evaluations on challenging locomotion and dynamic manipulation tasks involving quadruped and humanoid robots, along with benchmarks from HumanoidBench (Sferrazza et al., 2024), show that our approach consistently outperforms state-of-the-art baseline algorithms in both final performance and robustness. Furthermore, we demonstrate that the previously observed advantage of discrete over continuous policies in continuous control tasks (Tang & Agrawal, 2020) primarily arises from premature convergence, a limitation effectively alleviated by our method.

In summary, we make the following key contributions:

1. We propose **G**oal **A**chievement **G**uided **E**xploitation (GAGE), a novel framework that regulates exploration through an adaptive constraint linked to the agent's goal achieving performance, offering a principled and theoretically grounded alternative to entropy-based regularization.

2. We integrate GAGE seamlessly with continuous and discrete action spaces, as well as on- and off-policy algorithms. We demonstrate the strong empirical performance of the GAGE-based algorithms across challenging robot control tasks and the HumanoidBench benchmark.

3. We provide a formal analysis of the limitations of maximum entropy RL and derive a theoretical lower bound on action probabilities guaranteed by our method.

## 2 Background and Related Work

Premature convergence is a long-standing challenge in optimization and machine learning, affecting algorithms including genetic algorithms (Pandey et al., 2014) and reinforcement learning Yan et al. (2024a). To clarify the specific focus of this work, we distinguish our focus on premature convergence from issues such as "reward hacking" and sparse rewards in RL, as elaborated in Appendix A.1. In the following, we review the principle of entropy maximization, analyze its inherent limitations, and discuss related work.

### 2.1 Maximum Entropy Reinforcement Learning

In the standard RL setting, we model the task as a Markov decision process defined by the tuple $(\mathcal{S}, \mathcal{A}, p, r)$, where the agent interacts with the environment following the policy $\pi(\cdot \mid s_t)$ with state $s_t \in \mathcal{S}$ and action $a_t \in \mathcal{A}$. In each time step, the environment transits from state $s_t$ to $s_{t+1}$ according to the state transition probability $p\colon \mathcal{S}\times\mathcal{S}\times\mathcal{A} \to \mathbb{R}_+$ and gives the agent a reward following the reward function $r\colon \mathcal{S}\times\mathcal{A} \to \mathbb{R}$. The objective of standard RL is to maximize the expectation of the cumulative reward $\sum_t \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[r(s_t, a_t)]$, where $\rho_\pi$ stands for the state-action distribution following policy $\pi$. The objective can be extended to infinite-horizon problems with a discount factor $\gamma$.

A common practice in RL is to employ a maximum entropy objective (Williams & Peng, 1991; Mnih et al., 2016; Schulman et al., 2017; Haarnoja et al., 2018a; Espeholt et al., 2018), which augments the standard reward maximization with the expected entropy of the policy:

$$J(\pi) = \sum_{t=0}^{T} \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[r(s_t, a_t) + \beta\mathcal{H}(\pi(\cdot \mid s_t))],$$

where $T$ is the final time step of an episode, $\mathcal{H}(\pi)$ denotes the policy entropy, and $\beta$ controls the strength of the entropy regularization. This formulation encourages exploration by promoting higher entropy during training. However, as discussed below, it has limitations that significantly impair the learning process.

**Changing optimization objective** Since MaxEnt RL augments the reward function with an additional entropy term, it inevitably alters the optimization objective. While entropy maximization improves exploration and robustness (Ahmed et al., 2019), it can be detrimental in tasks that require a precise, low-entropy policy. To recover the conventional objective at convergence, the entropy temperature $\beta$ must be gradually reduced so that $\beta \to 0$. However, since the optimal $\beta$ value depends on the original reward scale, schedule tuning is basically required for each new task, and can become prohibitively costly due to the high sensitivity of RL algorithms to hyperparameters. In contrast, our goal is to develop a more adaptive mechanism that automatically adjusts exploration without the need for such manual scheduling.

**No theoretical guarantee for sufficient exploration** MaxEnt RL maximizes the entropy of the policy but provides no lower bound on the probability of individual actions. As a result, some actions required for optimal trajectories may still receive arbitrarily small probability, preventing adequate exploration. The agent becomes trapped in suboptimal policies once the probability of the optimal action approaches zero. Two factors contribute to this issue. First, entropy regularization is not a hard constraint. The temperature $\beta$ is difficult to tune, lacks intuitive interpretability, and is influenced by multiple factors such as the reward scale, the dimensionality of the action space, and even the specific local optima encountered during training (Haarnoja et al., 2018b). Second, for discrete actions, entropy serves only as a lossy summary statistic: $\mathcal{H}(u) = -\sum_{i=1,\dots,|\mathcal{A}|} u_i \log u_i$, where $u = (p(a_1), \dots, p(a_{|\mathcal{A}|})) \in \mathbb{R}_+^{|\mathcal{A}|}$. Even under a hard constraint on entropy, this formulation cannot ensure lower-bounded action probabilities. As shown in Fig. 1, two distributions can share the same entropy value (0.693) while differing substantially, with one of them collapsing the minimum action probability to zero.

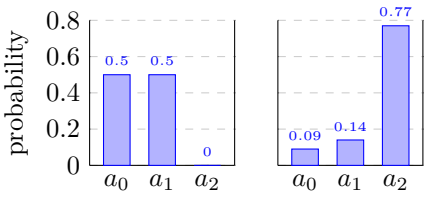



Figure 1: Discrete distributions with the same entropy.

## 2.2 Related Work

In this section, we review contemporary approaches to enhancing exploration in reinforcement learning.

**Adaptive hyperparameter tuning** Soft Actor-Critic (SAC) (Haarnoja et al., 2018a) is one of the most widely adopted off-policy RL algorithms based on the maximum entropy framework. However, its performance is highly sensitive to the temperature hyperparameter $\beta$, whose optimal value is non-trivial to tune. To address this issue, Haarnoja et al. (2018b) proposed learning a gradient-based $\beta$ that matches the expected entropy to a predefined target value. While this approach (SAC-v2) enables dynamic adjustment of $\beta$ during training, it shifts the tuning burden to the choice of the target entropy. The fixed entropy constraint also limits the policy's ability to either start learning with a higher exploration rate or converge to an optimal deterministic solution. To eliminate the need for additional hyperparameters, Wang & Ni (2020) applied a metagradient method (Xu et al., 2018) to tune $\beta$ automatically. However, the reported performance improvement over SAC is marginal, and metagradient updates can still become trapped in local optima due to their gradient-based nature.

**Intrinsic motivations** While our work addresses premature convergence in dense-reward environments, another line of research focuses on encouraging exploration in sparse-reward tasks, where the optimization landscape is largely flat and external rewards are scarce. Inspired by the curiosity-driven behaviors of animals (Schmidhuber, 1991), intrinsic motivation methods encourage exploration by rewarding the agent for visiting novel or informative states. Count-based approaches (Bellemare et al., 2016; Tang et al., 2017) achieve near-optimal exploration in tabular settings but scales poorly to high-dimensional or continuous domains. Prediction-error-based methods (Pathak et al., 2017; Burda et al., 2019) train forward or inverse dynamics models and use discrepancies between predicted and observed transitions as intrisic rewards. Zhang et al. (2021) proposed rewarding novelty differences between states to encourage breadth-first exploration, while Wan et al. (2023) scaled observation novelty using mutual information between states and trajectories. Although effective in sparse-reward environments such as MiniGrid (Chevalier-Boisvert et al., 2023) or navigation tasks, the applicability of these methods to high-dimentional continuous control task with severe

local optima remains not clear. Moreover, balancing extrinsic and intrinsic rewards is non-trivial. MaxInfoRL (Sukhija et al., 2025) mitigates this issue by introducing information-based rewards and automatically tuning the temperature for the transition information gain. They reported superior results in off-policy continuous control tasks with humanoid robots. However, the method incurs significant computational overhead due to model ensembles and target policies, and it is limited to off-policy settings.

**Goal and trajectory planning** The Markovian assumption in standard RL algorithms may be suboptimal for tasks involving partial observation or long-horizon dependencies. Recent works have approached exploration as a structured, long-term planning problem. Jain et al. (2023) utilized non-Markovian policies that condition on past trajectory to maximize state coverage within limited steps, achieving efficient exploration in gridworld and simple continuous control tasks such as Reacher and Pusher. Hu et al. (2023) leveraged learned world models and planning algorithms to generate exploratory goals with high exploration potential, effectively constructing a goal curriculum. Similarly, Diaz-Bone et al. (2025) quantify the achievability, novelty, and relevance of exploratory goals to guide exploration toward meaningful directions. However, these approaches generally rely on prior task knowledge and are primarily evaluated in low-dimentional action spaces or sparse-reward environments, such as navigation or gridworld tasks.

**Multi-modal policy** In standard MaxEnt RL implementations such as SAC, the policy is typically modeled as a Gaussian distribution, whose unimodal nature limits the representation of multiple behavioral modes. Tang & Agrawal (2020) addressed this issue by discretizing continuous action spaces. With a sufficiently fine granularity (e.g., over 11 bins per action dimension), discrete policies outperformed Gaussian ones on most MuJuCo benchmarks. More recently, Dong et al. (2025) adopted diffusion models for policy representation under the MaxEnt RL framework, enabling flexible, multi-modal action distributions. They demonstrated marginal performance improvement on MuJuCo tasks. Despite their enhanced representation capacity, these approaches remain constrained by the same limitations inherent to MaxEnt RL discussed earlier. In this work, we provide implementations using both Gaussian and discrete policies. We identify combining diffusion-based policies with our framework as a promising direction for future research.

## 3 Goal Achievement Guided Exploitation

To account for the limitations of maximum entropy reinforcement learning, we propose a novel framework named Goal Achievement Guided Exploitation (GAGE). In the optimal case, a policy should not converge before the agent approaches its maximum achievable performance. Therefore, it is natural to relate the policy's convergence level to its goal achievement. In this section, we first define goal achievement $g(\pi)$ formally, then describe how we use it to construct an adaptive constraint for policy convergence, and finally analyze the advantages of GAGE over MaxEnt RL.

### 3.1 Goal Achievement

Since an agent is trained to maximize the cumulative reward, we define its goal achievement as

$$g(\pi) \coloneqq \frac{J_\pi}{J_{\pi^*}}, \qquad J_\pi = \sum_{t=0}^{T} \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[r(s_t,a_t)]. \tag{1}$$

By definition, $g(\pi) \leq 1$ and $g(\pi) \to 1$ as $\pi \to \pi^*$. The expected return of the current policy $J_\pi$ can be estimated using the average return of recent rollouts. However, determining the optimal expected return $J_{\pi^*}$ is sometimes difficult for complex reward functions. In addition, some reward components may not align with the true task goal, potentially leading to suboptimal behaviors, where $g(\pi)$ increases without achieving the actual goal.

To address these issues, we examine the structure of the reward function more closely. A typical reward function consists of multiple terms, which can be categorized into *goal rewards* ($r_\mathrm{g}$) and *auxiliary rewards* ($r_\mathrm{a}$). Goal rewards are mandatory and correspond to the intended task objective, such as winning a game or executing a robotic behavior. Auxiliary rewards are optional heuristics designed to guide or accelerate

learning. We exclude auxiliary rewards when measuring goal achievement for two reasons: (1) they may not align with the true task goal and can lead to suboptimal solutions; (2) their maximum attainable values are often difficult to define, whereas the maximum goal reward values are typically available in the task specification. Accordingly, we define the goal achievement for the goal reward as:

$$g(\pi) \coloneqq \frac{J_{\mathrm{g},\pi}}{J_{\mathrm{g},\pi^*}}, \qquad J_{\mathrm{g},\pi} = \sum_{t=0}^{T} \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[r_{\mathrm{g}}(s_t, a_t)].$$

In practice, we approximate goal achievement using the moving average of the per-episode ratio between the cumulative goal reward $\sum_{t=0}^{T} r_{\mathrm{g},t}$ and its corresponding maximum value $\sum_{t=0}^{T} r_{\mathrm{g},t}^{\max}$.

Tasks can contain multiple goal reward terms (Xu et al., 2020; Hayes et al., 2022). In this work, we focus on single-goal settings and leave the multi-objective extension for future research. We primarily consider non-negative rewards. For negative rewards, transformations such as sigmoid or offset can ensure $g(\pi) \in [0, 1]$. If individual components are unavailable, $g(\pi)$ can be approximated using cumulative return of total rewards as Eq. 1. When the maximum return is unknown, the optimal performance can be estimated empirically from observed trajectories, as further demonstrated in Sec. 4.1.

### 3.2 Mitigating Premature Convergence via Action Smoothing

To prevent policy collapse, where the agent prematurely converges to a few actions in discrete spaces or an excessively narrow Gaussian in continuous spaces, we introduce an *action smoothing* technique. It avoids overconfidence by lower-bounding the flatness of the action distribution according to the current goal achievement. Let $\mathcal{F}(\pi)$ denote a generic flatness measure, and define its adaptive lower bound as a function negatively correlated with goal achievement:

$$\mathcal{F}_{\mathrm{LB}}(\pi) \coloneqq f(g(\pi)).$$

For simplicity, we assume the function $f$ is an affine mapping as in Eqs. 2 and 5, leaving investigation of alternative mappings for future work. We next specify $\mathcal{F}(\pi)$ and its implementation for continuous and discrete action spaces. Full algorithmic details with different backbone algorithms are outlined in Appendix B.

**Continuous action space** In continuous domains, exploration is typically facilitated by modeling actions as Gaussian distributions: $p(a \mid s) \sim \mathcal{N}(\mu(s), \sigma^2)$. This formulation is used in both stochastic policies, such as SAC and Proximal Policy Optimization (PPO) (Schulman et al., 2017; Haarnoja et al., 2018a), and deterministic ones such as Deep Deterministic Policy Gradient (DDPG) (Lillicrap et al., 2016), where Gaussian noise is added for exploration. We employ the standard deviation $\sigma$ of the policy as the flatness measure $\mathcal{F}(\pi)$ and define its adaptive lower bound as:

$$\sigma_{\mathrm{LB}}(\pi) \coloneqq -\sigma_0 g(\pi) + \sigma_0, \tag{2}$$

where the hyperparameter $\sigma_0 > 0$ controls the minimum allowed standard deviation when goal achievement is zero. Action smoothing is applied by clamping $\sigma$ to $\sigma_{\mathrm{LB}}$, ensuring $\sigma \geq \sigma_{\mathrm{LB}}$. When $\sigma_0 = 0$, the formulation reduces to the original baseline algorithms. Although entropy contains similar information to standard deviation for Gaussian distributions, directly constraining $\sigma$ is more intuitive and numerically stable, as $\sigma \to 0$ corresponds to convergence to deterministic policies, whereas entropy is unbounded.

**Discrete action space** For discrete actions $a_k$, where $k \in \{1, \ldots, K\}$, probabilities are typically computed via softmax:

$$p(a_k \mid s) = \mathrm{softmax}(z)_k = \frac{\exp(z_k)}{\sum_{i=1}^{K} \exp(z_i)}, \tag{3}$$

where $z = (z_1, \ldots, z_K) \in \mathbb{R}^K$ are the logits output from the policy network. Analogous to $\sigma$ for Gaussian distribution in continuous action spaces, the range of logits reflects the flatness of the resulting categorical distribution. We thus use the *negative logit range* $\delta_z$ as $\mathcal{F}(\pi)$:

$$\delta_z \coloneqq -\max_{i,j=1,\ldots,K} |z_i - z_j| = \min_k z_k - \max_k z_k, \tag{4}$$

and define the adaptive lower bound of $\delta_z$ as:

$$\delta_{z,\mathrm{LB}} \coloneqq (\delta_{z,1} - \delta_{z,0})g(\pi) + \delta_{z,0}, \tag{5}$$

where $\delta_{z,0}$ and $\delta_{z,1}$ are hyperparameters controlling the minimum allowed $\delta_z$ at $g(\pi) = 0$ and $g(\pi) = 1$. In principle, $\delta_{z,1} \to -\infty$ as the policy approaches deterministic. However, this would make it numerically difficult to calculate Eq. 5. In practice, we find $\delta_{z,1} \approx -15$ is sufficient. We implement this constraint using *softmax with adaptive temperature* (Asadi & Littman, 2017), $\mathrm{softmax}(z/\tau)$, which controls the distribution flatness by adaptively scaling logits before exponentiation. The adjusted logits are computed as:

$$z' = \frac{z}{\tau}, \quad \tau = \frac{\max\{1, |\delta_z|\}}{|\delta_{z,\mathrm{LB}}|},$$

after which action probabilities are calculated via $\mathrm{softmax}(z')$. The temperature $\tau$ rescales the logits so that $\delta_{z'} \geq \delta_{z,\mathrm{LB}}$. In practice, we find that enforcing a flat distribution with a hard constraint on the softmax logits alone can result in prohibitively large logit magnitudes. To address this issue, we regularize the logits by minimizing:

$$\mathcal{L}_z \coloneqq \alpha \sum_{t=0}^{T} \mathbb{E}_{s_t \sim \rho_\pi} \|z(s_t)\|_2^2, \tag{6}$$

where $\alpha$ is a regularization coefficient. It's worth noting that label smoothing (Szegedy et al., 2016), another technique for distribution flattening, is not suitable for action smoothing. We explain our choice for softmax with temperature in Appendix A.2 in detail.

### 3.3 Advantages over Entropy Regularization

GAGE directly addresses the key limitations of maximum entropy RL described in Sec. 2.1.

**Unchanged optimization ojective** Because GAGE does not modify the reward function, it preserves the original learning objective. Unlike MaxEnt RL methods, GAGE enables a dynamic exploration rate without manuel parameter scheduling, making it easier to tune and generalize. Furthermore, since the constraint depends solely on goal achievement, GAGE promotes more stable learning dynamics that are less sensitive to local optima than MaxEnt objectives.

**Guaranteed sufficient exploration** Unlike the soft regularization of MaxEnt RL, GAGE imposes hard lower bounds on the flatness of the action distribution, ensuring that exploration never vanishes. This formulation analytically lower-bounds the probability of all actions for both continuous and discrete spaces. For Gaussian policy, this property follows naturally from the lower-bounded standard deviation. For discrete actions, the probability bound after action smoothing can be derived explicitly from Eqs. 3 and 4:

$$\begin{aligned}\min_k p(a_k \mid s) =& \frac{\exp\left(\min_k z_k\right)}{\sum_{i=1}^{K} \exp z_i} \\ \geq& \frac{\exp(\max_k(z_k) + \delta_{z,\mathrm{LB}})}{\exp(\max_k(z_k) + \delta_{z,\mathrm{LB}}) + (K-1)\exp(\max_k z_k)} = \frac{\exp \delta_{z,\mathrm{LB}}}{\exp \delta_{z,\mathrm{LB}} + (K-1)}.\end{aligned}$$

Thus, the minimum action probability is adaptively lower-bounded as a function of the goal achievement $g(\pi)$, guaranteeing that every action retains a nonzero probability. This mechanism ensures sufficient exploration of optimal behaviors until the desired performance is reached.

## 4 Experiments

This section validates GAGE on robot control problems characterized by high degrees of freedom and local optima, which often lead to premature convergence in reinforcement learning. We show (i) how GAGE achieves superior on- and off-policy exploration efficacy compared to MaxEnt-based baselines, (ii) how GAGE demonstrates strong robustness to unknown optimal goals and variations in reward shaping, and (iii) how GAGE generalizes to discretized action spaces.

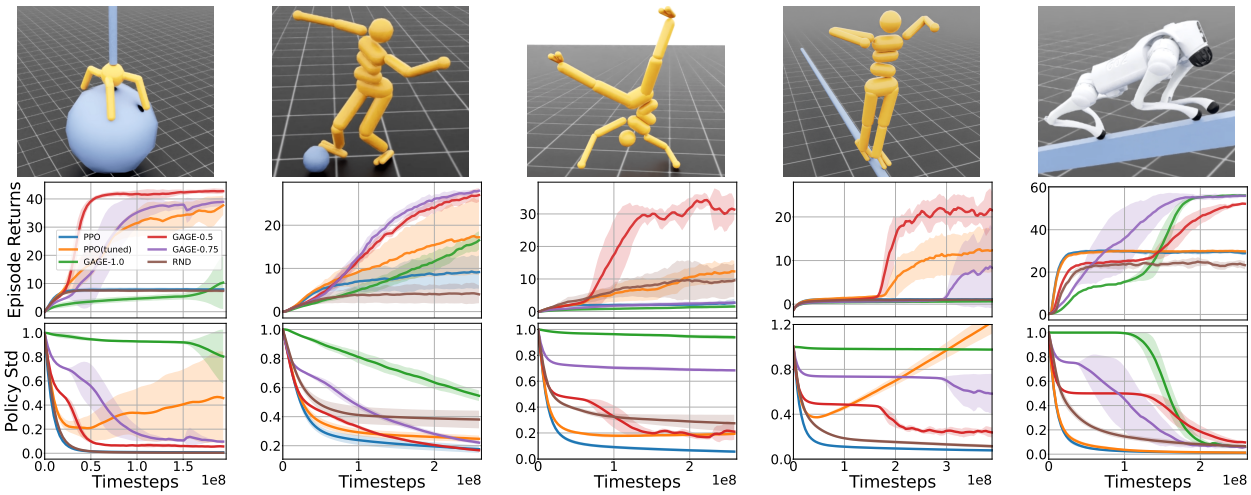



Figure 2: Mean episode returns and action standard deviation $\sigma$ over 5 seeds; shaded regions denote one standard deviation. We denote GAGE-0.5 as using $\sigma_0 = 0.5$. **Top**: tasks (left to right), Ant Acrobatics, Humanoid Dribbling, Humanoid Cartwheel, Humanoid Tightrope, and Dog (Unitree Go2) Balance Beam. **Middle**: training curves of episode returns. **Bottom**: Averaged policy standard deviation $\bar{\sigma}$.

### 4.1 GAGE for Continuous Action Space

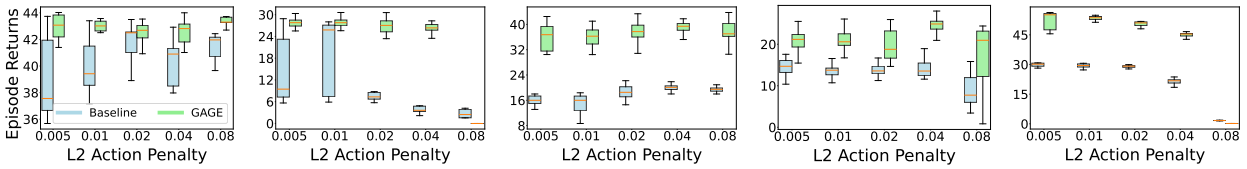
We integrate GAGE into both on-policy and off-policy algorithms by replacing entropy regularization with our adaptive constraint. First, we combine GAGE with PPO, which benefits from large-scale GPU parallelism (Makoviychuk et al., 2021). To test GAGE under severe local optima issues, we design five highly challenging tasks in IsaacLab (Mittal et al., 2023; Yan et al., 2024b) (see Fig. 2). The environment details are provided in Appendix D.1. We then incorporate GAGE into SAC and evaluate SAC-GAGE, on HumanoidBench (Sferrazza et al., 2024), the stand, walk, and run tasks.

#### On-Policy RL with GAGE

We compare PPO-GAGE against: (i) vanilla PPO with default hyperparameters, (ii) PPO with per-task tuned hyperparameters, and (iii) Random Network Distillation (RND) (Burda et al., 2019). We include RND to test whether the curiosity-driven method, designed for sparse rewards, can mitigate premature convergence in dense-reward settings. To obtain strong baselines, we perform a grid search over the PPO exploration hyperparameter, the entropy coefficient, per task and adopt the best-performing values (see Appendix D.2 for details). For RND We follow the hyperparameter settings of the original and subsequent work of Yang et al. (2024). PPO-GAGE sets entropy temperature $\beta = 0$ and uses the same hyperparameters as vanilla PPO for a direct comparison. With action spaces normalized to $[-1, 1]$, intuitively, we set $\sigma_0 \in \{1.0, 0.75, 0.5\}$ for an ablation study.

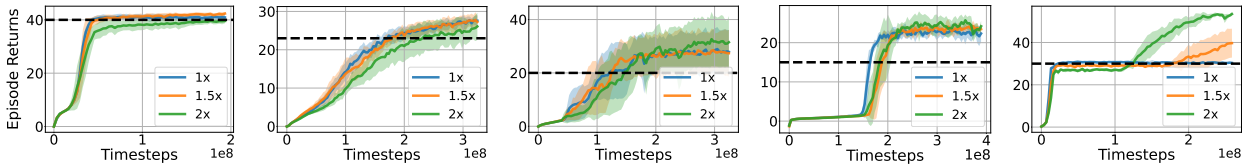
Fig. 2 shows learning curves for episode returns and average $\sigma$ across robot joints. PPO-GAGE with $\sigma_0 = 0.5$ successfully solved all the tasks, whereas vanilla PPO and RND failed. Hyperparameter optimisation (HPO) over the entropy coeffcient substantially improves PPO on Ant Acrobatics, Humanoid Dribbling, Humanoid Cartwheel, and Humanoid Tightrope. However, on the Balance Beam task, HPO yields no noticeable improvement. Among all tasks, PPO-GAGE still wins by clear margins. PPO-GAGE with $\sigma_0 = 0.5$ or 0.75 performs reliably across the majority of tasks. In contrast, agents with $\sigma_0 = 1.0$ generally exhibit poorer performance, as the excessively large action variance produces highly random behaviors. Hence, for practical use, we recommend starting with $\sigma_0$ values between 0.5 and 0.75.

The $\sigma$ plots reveal that vanilla PPO quickly reduces policies' standard deviation at the start of training, achieving higher rewards by over-exploiting reward components such as energy cost or survival. PPO continues to decrease $\sigma$ even after the target rewards plateau. For instance, the dog robot learns to stabilize




(a) Ant Acrobatics (b) Dribbling (c) Cartwheel (d) Tightrope (e) Balance Beam

Figure 3: Box plots of episode returns (excluding action-penalty values) for tuned PPO versus PPO-GAGE. The $x$-axis shows the investigated coefficients of the squared L2 action penalty $\|a\|_2^2$. We aggregate the last 10 episode returns of each seed for 50 data points per box.




(a) Ant Acrobatics (b) Dribbling (c) Cartwheel (d) Tightrope (e) Balance Beam

Figure 4: Using 1×, 1.5×, and 2× return of the best tuned PPO (black dashed line) as the total reward goal $\hat{G}$ for PPO-GAGE. We set $\sigma_0 = 0.75$ for Go2Beam and Humanrope, and 0.5 for the other three tasks. PPO-GAGE matches or exceeds tuned PPO. Larger $\hat{G}$ can further improve performance in some tasks.

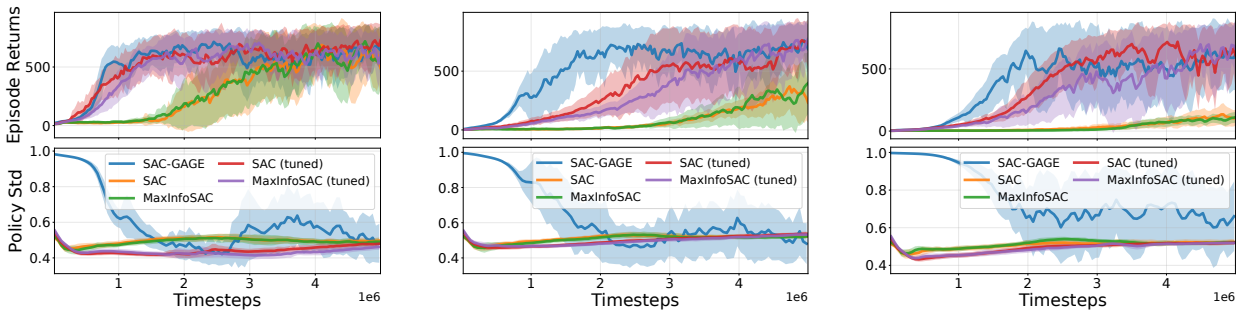
on the beam and ceases exploring despite a forward motion target. In contrast, PPO-GAGE sustains exploration and only reduces $\sigma$ when the desired behaviors are learned (see Appendix D.1). The best tuned PPO agents on Ant Acrobatics and Humanoid Tightrope exhibit an unbounded increase in $\sigma$ after an initial drop, indicating that the agent learns to lock certain robot joints with large actor mean values, while expoiting the maximum entropy reward. This behavior prevents the robot from fully utilizing its action space, resulting in unnatural and suboptimal motion patterns.

RND slows down the reduction of $\sigma$ relative to PPO. However, the additional exploration does not solve the tasks and can even hurt performance. We hypothesize that RND encourages exploration of states even when they are irrelevant to the motion objective. Broad state coverage is useful for navigation in low-dimensional environments but inefficient for robot control with many degrees of freedom. Further experimental results with varying intrinsic reward settings are provided in Fig. 10.

#### Improved Robustness to Reward Shaping

Reward shaping is crucial yet delicate, as small changes to reward weights can result in unsuccessful learning. We suppose the sensitivity to reward shaping largely arises from premature convergence. We conduct experiments to test the robustness of our method to variations in reward shaping.

The L2 norm of actions is used as a penalty term to prevent large actions in robot joints. We vary the action-penalty weight while keeping other weights fixed. The results are illustrated in Fig. 3. To compare across settings, the episode return excludes the action penalty term. PPO with tuned hyperparameters is highly sensitive to the penalty coefficient (e.g., in Dribbling task, the average return at the coefficient of 0.01 is roughly nine times that at 0.04). In contrast, PPO-GAGE outperforms PPO across nearly all settings. It remains robust for action penalty coefficient from 0.005 to 0.04 in every tasks. Both methods struggle when the penalty weight is too large (i.e., 0.08 in Dribbling and Balance Beam), suggesting room for future improvement. The full training curves are included in Fig. 12 and Fig. 13.




(a) H1-Stand-v0 (b) H1-Walk-v0 (c) H1-Run-v0

Figure 5: Comparison of SAC, MaxInfoSAC and SAC-GAGE on HumanoidBench. Mean episode returns and action $\sigma$ over 5 seeds; shaded regions denote one standard deviation.

### Unknown Individual Reward & Optimal Goal

When the agent only receives total rewards and lacks access to individual components, we approximate $g(\pi)$ via the Monte Carlo return as Eq. 1. Since the optimal return $J_{\pi^*}$ is typically unknown, we estimate it by a proxy $\hat{G}$ obtained from the best performance of a reference agent. On our IsaacLab tasks, we consider $\hat{G} \in \{1\times, 1.5\times, 2\times\}$ the best tuned PPO return and train different PPO-GAGE agents. As depicted in Fig. 4, PPO-GAGE performs at least as well as tuned PPO, and improves further as $\hat{G}$ increases for tasks such as Dog Balance Beam and Humanoid Cartwheel. The same approach applies when individual reward terms exist but the optimal goal $J_{\mathrm{g},\pi^*}$ is unknown.

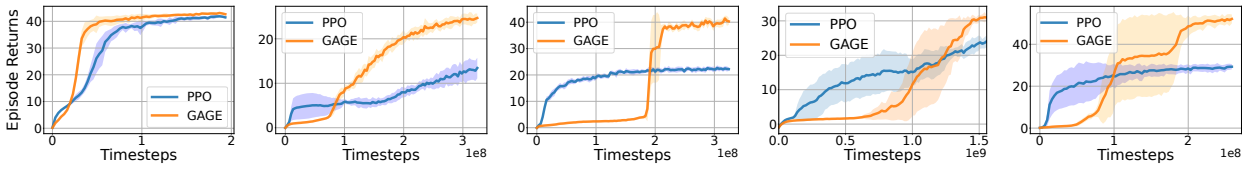
### Off-Policy RL with GAGE

We evaluate SAC-GAGE against SAC and MaxInfoSAC (Sukhija et al., 2025), the state-of-the-art off-policy algorithms on continuous control. Starting from the SAC implementation provided by Sukhija et al. (2025), we replace entropy regularization with GAGE constraint and keep other components unchanged, see Algorithm 1 for details. Because tanh-squashed Gaussians can cause value overestimation and gradient vanishing near action space boundaries, we add a policy regularizer analogous to Eq. 6 to avoid prohibitively large actor mean magnitudes. Without individual reward components, we utilize estimated total return $\hat{G}$ to calculate goal achievement. The hyperparameters are given in Table 5.

We conduct HPO for SAC and MaxInfoSAC over learning rates and gradient clipping values to obtain strong baselines. Fig. 5 shows learning curves for both the original and the best tuned settings: SAC and MaxInfoSAC achieve similar performance and, after hyperparameter tuning, solve the tasks at comparable performance. SAC-GAGE also solves all evaluated tasks with comparable final returns. On H1-Walk and H1-Run, it substantially improves learning speed over the baselines, achieving same final performance using only 40% of the training steps compared to tuned MaxInfoSAC. We attribute this to its initially higher exploration level. Without the adaptive constraint on policy standard deviation GAGE applied, SAC and MaxInfoSAC rapidly reduce the policy standard deviation $\sigma$ below 0.5 early in training, and keeps roughly fixed exploration rate afterwards. In contrast, SAC-GAGE maintains $\sigma \approx 1.0$ until returns improve, then adaptively decreases it in response to goal achievement.

## 4.2 GAGE for Discrete Action Space

We further validate GAGE with discrete policies on our IsaacLab tasks by discretizing each action dimension into 11 evenly spaced atomic actions, following Tang & Agrawal (2020). Each dimension is modeled by its own categorical distribution, yielding a factorized policy. We compare PPO-GAGE against the strongest baseline method, tuned PPO, with discrete policies. For PPO-GAGE, we apply action smoothing (see Algorithm 3) in each action dimension. The PPO entropy coefficient is tuned for each individual task. PPO-




(a) Ant Acrobatics (b) Dribbling (c) Cartwheel (d) Tightrope (e) Balance Beam

Figure 6: PPO vs. PPO-GAGE on discrete action spaces with hyperpameter tuning for both. Mean episode returns and action $\sigma$ over 5 seeds; shaded regions denote one standard deviation.

GAGE uses the other default hyperparameters from vanilla PPO with its own hyperparameters tuned. The tuned hyperparameters can be found in Table 2 and 3.

Fig. 6 presents the learning curves: PPO-GAGE solves all tasks, whereas tuned PPO fails on Humanoid Dribbling, Humanoid Cartwheel, and Dog Balance Beam. Interestingly, although tuned PPO attains relatively high returns on the Humanoid Tightrope task, its goal achievement value remains low. Fig. 7 further compares the episode returns and goal achievement $g$ for PPO with the two best entropy temperatures $\beta = 5.18 \times 10^{-3}$ and $1.39 \times 10^{-2}$ alongside PPO-GAGE for the Humanoid Tightrope task. For $\beta = 5.18 \times 10^{-3}$, three out of five runs with differernt seeds learn to survive by remaining stationary and minimizing energy, yielding deceptively high returns. With $\beta = 1.39 \times 10^{-2}$, all agents learn to move but still converge to suboptimal behaviors, such as unstable motion and excessive energy cost, resulting in lower returns. These observations highlight the vulnerability of MaxEnt RL to local optima, a failure mode that GAGE effectively mitigates. On Humanoid Tightrope, the final performance of GAGE with discrete policies surpasses that with continuous ones (Fig. 2), primarily due to differences in action penalties. In continuous action spaces, the unbounded nature of the PPO policy allows the agent to exploit large action magnitudes beyond the physical limits of the robot's joints. Although such actions are clipped within the simulation environment, they still incur substantial penalties, reducing total reward despite achieving comparable goal achievement.

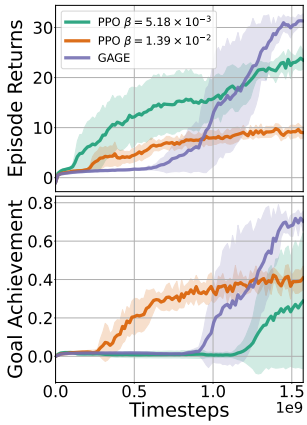



Figure 7: PPO-GAGE vs. best two PPO agents on Humanoid Tightrope. Mean episode returns and goal achievement $g$ over 5 seeds; shaded regions denote one standard deviation.

Under the GAGE framework, discrete policies do not always surpass continuous ones and can even be less efficient such as on Humanoid Dribbling, Cartwheel, and Tightrope, contrary to the results of Tang & Agrawal (2020). While action discretization can mitigate local optima and premature convergence under the MaxEnt framework, it sacrifices the expressiveness of continuous actions and potentially optimal behaviors. The adaptive exploration strategy in GAGE allows the agent to fully leverage the continuous action space, resulting in more efficient learning.

## 5 Discussion

The results presented in this work demonstrate that Goal Achievement Guided Exploitation (GAGE) provides a principled and effective alternative to maximum entropy reinforcement learning (MaxEnt RL) for addressing premature convergence. Across diverse settings, including both discrete and continuous action spaces, on-policy and off-policy algorithms, and challenging robotic control tasks, GAGE consistently enhances learning stability and final performance.

**Adaptive exploitation and learning stability** By linking the degree of exploration directly to the agent's goal achievement, GAGE decouples exploration from manually predefined training schedules or fixed entropy targets. This allows the agent to adaptively reduce stochasticity as it approaches optimal performance. Empirically, this results in smoother convergence curves and reduced variance across random seeds

compared to MaxEnt-based baselines. Notably, in highly non-convex control tasks such as Dog Balance Beam, GAGE avoids early policy collapse and continues to improve after standard methods saturate.

**Comparison with entropy-based regularization** A key strength of GAGE lies in its hard constraint formulation. Unlike entropy regularization, which indirectly promotes exploration through a soft additive term in the reward function, GAGE constrains the policy distribution itself through interpretable measures, the policies' standard deviation in continuous spaces and logit range in discrete spaces. This distinction leads to two advantages. First, the original reward objective remains unaltered, eliminating the need for entropy-temperature annealing. Second, the lower-bounded action distribution flatness ensures that all actions maintain a nonzero probability throughout training, thereby preventing the irreversible loss of potentially optimal behaviors.

**Robustness and practical deployment** The experimental results highlight that GAGE is robust to hyperparameter variations and insensitive to reward shaping, two major weaknesses of conventional RL algorithms. Even with intuitive default parameters, GAGE achieves superior results compared to finely tuned baselines. This property simplifies deployment in real-world systems. Furthermore, the framework performs well even when the optimal goal value is approximated, suggesting that GAGE can be readily applied in environments with incomplete reward decomposition.

**Limitations and future work** Despite these strengths, GAGE offers aspects for improvement. GAGE defines the same bound for all action dimensions. It can be interesting to investigate the effect of different bounds for different dimensions. Future research can focus on improving the scalability of GAGE and applying it to more complex, dynamic, and multi-objective environments. Investigating non-linear relationships between the goal achievement and the exploration metrics could further enhance the method's adaptability to diverse RL problems.

In summary, GAGE rethinks the fundamental role of entropy in reinforcement learning. By grounding exploration in measurable progress toward task goals, it preserves the original optimization objective, ensures sufficient exploration, and yields more stable and interpretable learning dynamics, representing a promising direction beyond the maximum entropy paradigm.

## References

Zafarali Ahmed, Nicolas Le Roux, Mohammad Norouzi, and Dale Schuurmans. Understanding the impact of entropy on policy optimization. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proc. of the International Conference on Machine Learning (ICML)*, volume 97, pp. 151–160. PMLR, 2019.

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul F. Christiano, John Schulman, and Dan Mané. Concrete problems in AI safety. *CoRR*, abs/1606.06565, 2016.

Kavosh Asadi and Michael L. Littman. An alternative softmax operator for reinforcement learning. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 243–252. PMLR, 06–11 Aug 2017. URL `https://proceedings.mlr.press/v70/asadi17a.html`.

Marc G. Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Rémi Munos. Unifying count-based exploration and intrinsic motivation. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pp. 1471–1479, 2016.

Yuri Burda, Harrison Edwards, Amos J. Storkey, and Oleg Klimov. Exploration by random network distillation. In *Proc. of the International Conference on Learning Representations (ICLR)*. OpenReview.net, 2019.

Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-SGD: Biasing gradient descent into wide valleys. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2017.

Maxime Chevalier-Boisvert, Bolun Dai, Mark Towers, Rodrigo De Lazcano Perez-Vicente, Lucas Willems, Salem Lahlou, Suman Pal, Pablo Samuel Castro, and J K Terry. Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=PFfmfspm28.

Leander Diaz-Bone, Marco Bagatella, Jonas Hübotter, and Andreas Krause. DISCOVER: Automated curricula for sparse-reward reinforcement learning. In *Proc. of the Conference on Neural Information Processing Systems (NeurIPS)*, 2025.

Xiaoyi Dong, Jian Cheng, and Xi Sheryl Zhang. Maximum entropy reinforcement learning with diffusion policy. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proc. of the International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pp. 13963–13983. PMLR, 13–19 Jul 2025.

Lasse Espeholt, Hubert Soyer, Rémi Munos, Karen Simonyan, Volodymyr Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, and Koray Kavukcuoglu. IMPALA: scalable distributed deep-rl with importance weighted actor-learner architectures. In Jennifer G. Dy and Andreas Krause (eds.), *Proc. of the International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1406–1415. PMLR, 2018.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4RL: datasets for deep data-driven reinforcement learning. *CoRR*, abs/2004.07219, 2020. URL https://arxiv.org/abs/2004.07219.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Jennifer G. Dy and Andreas Krause (eds.), *Proc. of the International Conference on Machine Learning (ICML)*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1856–1865. PMLR, 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. *CoRR*, abs/1812.05905, 2018b.

Conor F. Hayes, Roxana Radulescu, Eugenio Bargiacchi, Johan Källström, Matthew Macfarlane, Mathieu Reymond, Timothy Verstraeten, Luisa M. Zintgraf, Richard Dazeley, Fredrik Heintz, Enda Howley, Athirai A. Irissappane, Patrick Mannion, Ann Nowé, Gabriel de Oliveira Ramos, Marcello Restelli, Peter Vamplew, and Diederik M. Roijers. A practical guide to multi-objective reinforcement learning and planning. *Auton. Agents Multi Agent Syst.*, 36(1):26, 2022. doi: 10.1007/S10458-022-09552-Y. URL https://doi.org/10.1007/s10458-022-09552-y.

Edward S. Hu, Richard Chang, Oleh Rybkin, and Dinesh Jayaraman. Planning goals for exploration. In *Proc. of the International Conference on Learning Representations (ICLR)*. OpenReview.net, 2023.

Arnav Kumar Jain, Lucas Lehnert, Irina Rish, and Glen Berseth. Maximum state entropy exploration using predecessor and successor representations. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Proc. of the Conference on Neural Information Processing Systems (NeurIPS)*, 2023.

Pawel Ladosz, Lilian Weng, Minwoo Kim, and Hyondong Oh. Exploration in deep reinforcement learning: A survey. *Information Fusion*, 85:1–22, 2022. ISSN 1566-2535. doi: https://doi.org/10.1016/j.inffus.2022.03.003. URL https://www.sciencedirect.com/science/article/pii/S1566253522000288.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2016. URL http://arxiv.org/abs/1509.02971.

Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. Isaac gym: High performance GPU based physics simulation for robot learning. In Joaquin Vanschoren and Sai-Kit Yeung (eds.), *Proc. of the Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

Mayank Mittal, Calvin Yu, Qinxi Yu, Jingzhou Liu, Nikita Rudin, David Hoeller, Jia Lin Yuan, Ritvik Singh, Yunrong Guo, Hammad Mazhar, Ajay Mandlekar, Buck Babich, Gavriel State, Marco Hutter, and Animesh Garg. Orbit: A unified simulation framework for interactive robot learning environments. *IEEE Robotics and Automation Letters*, 8(6):3740–3747, 2023. doi: 10.1109/LRA.2023.3270034.

Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In Maria-Florina Balcan and Kilian Q. Weinberger (eds.), *Proc. of the International Conference on Machine Learning (ICML)*, volume 48 of *JMLR Workshop and Conference Proceedings*, pp. 1928–1937. JMLR.org, 2016.

Rafael Müller, Simon Kornblith, and Geoffrey E. Hinton. When does label smoothing help? In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Proc. of the Conference on Neural Information Processing Systems (NeurIPS)*, pp. 4696–4705, 2019.

Hari Mohan Pandey, Ankit Chaudhary, and Deepti Mehrotra. A comparative review of approaches to prevent premature convergence in GA. *Appl. Soft Comput.*, 24:1047–1077, 2014. doi: 10.1016/J.ASOC.2014.08.025. URL https://doi.org/10.1016/j.asoc.2014.08.025.

Deepak Pathak, Pulkit Agrawal, Alexei A. Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2778–2787. PMLR, 2017. URL http://proceedings.mlr.press/v70/pathak17a.html.

Matthias Plappert, Marcin Andrychowicz, Alex Ray, Bob McGrew, Bowen Baker, Glenn Powell, Jonas Schneider, Josh Tobin, Maciek Chociej, Peter Welinder, Vikash Kumar, and Wojciech Zaremba. Multi-goal reinforcement learning: Challenging robotics environments and request for research. *CoRR*, abs/1802.09464, 2018. URL http://arxiv.org/abs/1802.09464.

Juergen Schmidhuber. A possibility for implementing curiosity and boredom in model-building neural controllers. In *Proc. of the International Conference on Simulation of Adaptive Behavior: From Animals to Animats*, pp. 222–227, 1991.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017.

Carmelo Sferrazza, Dun-Ming Huang, Xingyu Lin, Youngwoon Lee, and Pieter Abbeel. Humanoidbench: Simulated humanoid benchmark for whole-body locomotion and manipulation. In Dana Kulic, Gentiane Venture, Kostas E. Bekris, and Enrique Coronado (eds.), *Proc. of Robotics: Science and Systems (RSS)*, 2024.

Bhavya Sukhija, Stelian Coros, Andreas Krause, Pieter Abbeel, and Carmelo Sferrazza. Maxinforl: Boosting exploration in reinforcement learning through information gain maximization. In *Proc. of the International Conference on Learning Representations (ICLR)*. OpenReview.net, 2025.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818–2826. IEEE Computer Society, 2016. doi: 10.1109/CVPR.2016.308. URL https://doi.org/10.1109/CVPR.2016.308.

Haoran Tang, Rein Houthooft, Davis Foote, Adam Stooke, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. #exploration: A study of count-based exploration for deep reinforcement learning. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), *Proc. of the Conference on Neural Information Processing Systems (NeurIPS)*, pp. 2753–2762, 2017.

Yunhao Tang and Shipra Agrawal. Discretizing continuous action space for on-policy optimization. In *Proc. of the AAAI Conference on Artificial Intelligence*, pp. 5981–5988. AAAI Press, 2020.

Shanchuan Wan, Yujin Tang, Yingtao Tian, and Tomoyuki Kaneko. DEIR: efficient and robust exploration through discriminative-model-based episodic intrinsic rewards. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI, Macao, China*, pp. 4289–4298. ijcai.org, 2023. doi: 10.24963/IJCAI.2023/477. URL https://doi.org/10.24963/ijcai.2023/477.

Kaixin Wang, Kuangqi Zhou, Bingyi Kang, Jiashi Feng, and Shuicheng Yan. Revisiting intrinsic reward for exploration in procedurally generated environments. In *Proc. of the International Conference on Learning Representations (ICLR)*, 2023.

Yufei Wang and Tianwei Ni. Meta-sac: Auto-tune the entropy temperature of soft actor-critic via metagradient. *CoRR*, abs/2007.01932, 2020.

Ronald J Williams and Jing Peng. Function optimization using connectionist reinforcement learning algorithms. *Connection Science*, 3(3):241–268, 1991.

Jie Xu, Yunsheng Tian, Pingchuan Ma, Daniela Rus, Shinjiro Sueda, and Wojciech Matusik. Prediction-guided multi-objective reinforcement learning for continuous robot control. In *Proc. of the International Conference on Machine Learning (ICML)*, volume 119 of *Proceedings of Machine Learning Research*, pp. 10607–10616. PMLR, 2020. URL http://proceedings.mlr.press/v119/xu20h.html.

Zhongwen Xu, Hado van Hasselt, and David Silver. Meta-gradient reinforcement learning. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett (eds.), *Proc. of the Conference on Neural Information Processing Systems (NeurIPS)*, pp. 2402–2413, 2018.

Shengchao Yan, Baohe Zhang, Joschka Boedecker, and Wolfram Burgard. Goal achievement guided exploration: Mitigating premature convergence in learning robot control. In *CoRL 2024 Workshop on Whole-body Control and Bimanual Manipulation: Applications in Humanoids and Beyond*, 2024a.

Shengchao Yan, Baohe Zhang, Yuan Zhang, Joschka Boedecker, and Wolfram Burgard. Learning continuous control with geometric regularity from robot intrinsic symmetry. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 49–55. IEEE, 2024b.

Kai Yang, Jian Tao, Jiafei Lyu, and Xiu Li. Exploration and anti-exploration with distributional random network distillation. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=rIrpzmqRBk.

Ruipeng Zhang, Ya-Chien Chang, and Sicun Gao. When maximum entropy misleads policy optimization. In *Forty-second International Conference on Machine Learning*, 2025.

Tianjun Zhang, Huazhe Xu, Xiaolong Wang, Yi Wu, Kurt Keutzer, Joseph E. Gonzalez, and Yuandong Tian. Noveld: A simple yet effective exploration criterion. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Proc. of the Conference on Neural Information Processing Systems (NeurIPS)*, pp. 25217–25230, 2021.

# A More Background and Related Work

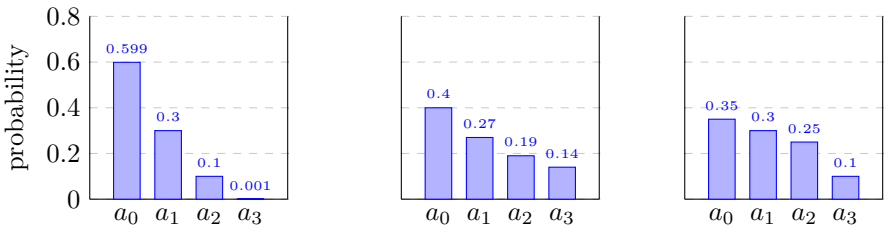
## A.1 Premature Convergence

Premature convergence is ubiquitous in reinforcement learning due to the non-convexity of system dynamics, reward shaping, multi-objective, and function approximation error using neural networks. A policy may converge to suboptimal solutions before reaching the desired performance. We briefly clarify the differences between premature convergence and two other related problems in RL, i.e., reward hacking and the sparse reward problem.

In reward hacking, the objective function that the designer writes down admits of some "clever" solutions that formally maximize it but pervert the spirit of the designer's intent (Amodei et al., 2016). For example, a cleaning robot can close its sensors to maximize the reward for observing fewer messes. This differs from premature convergence, as agents stuck at suboptimal solutions typically do not maximize the objective function.

Sparse rewards have been regarded as one of the most challenging problems in reinforcement learning (Ladosz et al., 2022). With limited rewards scattered in the vast search space, the result is a primarily flat objective landscape. In such settings, the agent can hardly find the rewards and maximize the objective, no matter whether the policy is converged or not. This differs from premature convergence, as agents increase the objective by converging to suboptima. Of course, the challenge is exacerbated by the combination of these two issues, such as the noisy-TV problem (Burda et al., 2019). However, in this work, we focus on tasks with dense settings and numerous local optima. Note that sparse reward tasks, such as navigation and long-term robot control, can be transferred to dense reward tasks based on domain knowledge. However, auxiliary rewards often introduce new local optima and exacerbate the problem of premature convergence.

## A.2 Label Smoothing




(a) Original distribution (b) Label smoothing (c) Action smoothing

Figure 8: A categorical distribution and its flattened results using different techniques. The resulting distributions have the same entropy 1.31. (a) Original distribution. (b) Label smoothing has a deterministic result calculated with smoothing parameter $\epsilon = 0.58$. (c) Action smoothing also has a deterministic result using temperature $\tau = 5.34$.

Label smoothing (Szegedy et al., 2016) is a popular technique for flattening discrete distributions. It is widely used in classification problems to reduce overconfidence through soft learning targets (Müller et al., 2019). It typically mixes the original distribution with a uniform distribution: $p'(a_k \mid s) = (1 - \epsilon)p(a_k \mid s) + \frac{\epsilon}{K}$, where the smoothing parameter $0 \leq \epsilon \leq 1$. However, it introduces a prior of uniform distribution that is inappropriate for policy learning. As shown in Fig. 8, actions with the lowest learned probabilities often lead to obvious penalties or termination states, which the agent should avoid. These actions experience the most significant increase through label smoothing, which can potentially lead to failed trials. In contrast, with action smoothing, the most significant probability increases occur for actions with middle-valued probabilities.

## B Algorithm Implementation

We provide the pseudo code for SAC/PPO+GAGE with a Gaussian policy in Alg. [1,2] and action smoothing of a categorical policy in Alg. 3. We strike through the MaxEnt related components in the original algorithms to highlight the differences.

**Algorithm 1** Soft Actor-Critic (SAC-v2) with Gaussian Policy + ~~MaxEnt~~ GAGE

**Require:** environment $\mathcal{E}$, discount $\gamma$, target smoothing $\tau$, replay size $N$, batch size $B$, ~~target entropy $\bar{\mathcal{H}}$ (e.g., $-\dim(\mathcal{A})$)~~, standard deviation lower bound $\sigma_{\text{LB}}$, goal achievement update factor $\lambda$

Initialize actor $\pi_\phi(a|s) = \mathcal{N}\big(\mu_\phi(s), \Sigma_\phi(s)\big)$ (Tanh-squashed Gaussian), critics $Q_{\theta_1}, Q_{\theta_2}$, target critics $\bar{Q}_{\theta'_1} \leftarrow Q_{\theta_1}$, $\bar{Q}_{\theta'_2} \leftarrow Q_{\theta_2}$, ~~temperature $\alpha > 0$~~, goal achievement $g = 0$
Initialize replay buffer $\mathcal{D} \leftarrow \emptyset$
**for** episode $= 1, 2, \ldots$ **do**
  Reset env, get $s_0$
  **for** t = 0,1,2,... **do**
    $\sigma = \sigma_\phi(s_t)$ ~~$\sigma_\phi(s_t)$~~ $\max(\sigma_{\text{LB}}, \sigma_\phi(s_t))$
    Sample action via reparameterization: $\epsilon \sim \mathcal{N}(0, I)$;  $u = \mu_\phi(s_t) + \sigma \odot \epsilon$;  $a_t = \tanh(u)$
    Execute $a_t$ in $\mathcal{E}$, observe $(r_t, s_{t+1}, d_t)$;  store $(s_t, a_t, r_t, s_{t+1}, d_t)$ in $\mathcal{D}$
    **for** gradient step $= 1, 2, \ldots$ **do**
      Sample minibatch $\{(s_i, a_i, r_i, s'_i, d_i)\}_{i=1}^{B} \sim \mathcal{D}$
      $\sigma = \sigma_\phi(s'_i)$ ~~$\sigma_\phi(s'_i)$~~ $\max(\sigma_{\text{LB}}, \sigma_\phi(s'_i))$
      **Target action and log-prob:** $\epsilon' \sim \mathcal{N}(0, I)$; $u' = \mu_\phi(s'_i) + \sigma \odot \epsilon'$; $a'_i = \tanh(u')$; compute $\log \pi_\phi(a'_i|s'_i)$ with tanh correction
      **Target Q:** $\hat{Q}_i = \min\big(\bar{Q}_{\theta'_1}(s'_i, a'_i),\, \bar{Q}_{\theta'_2}(s'_i, a'_i)\big)$
      **Bellman target:** $y_i = r_i + \gamma(1 - d_i)\big(\hat{Q}_i - $ ~~$\alpha \log \pi_\phi(a'_i|s'_i)$~~$\big)$
      **Critic loss:** $\mathcal{L}_Q(\theta_j) = \frac{1}{|B|}\sum_i \big(Q_{\theta_j}(s_i, a_i) - y_i\big)^2, \quad j \in \{1, 2\}$
      $\sigma = \sigma_\phi(s_i)$ ~~$\sigma_\phi(s_i)$~~ $\max(\sigma_{\text{LB}}, \sigma_\phi(s_i))$
      Update $\theta_1, \theta_2$ by minimizing $\mathcal{L}_Q$
      **Actor sampling:** $\epsilon \sim \mathcal{N}(0, I)$; $u = \mu_\phi(s_i) + \sigma \odot \epsilon$; $a_i = \tanh(u)$; compute $\log \pi_\phi(a_i|s_i)$
      **Actor loss (reparameterized):**

$$\mathcal{L}_\pi(\phi) = \frac{1}{|B|}\sum_i \Big( \text{~~}\alpha \log \pi_\phi(a_i|s_i)\text{~~} - \min_{k=1,2} Q_{\theta_k}(s_i, a_\phi(s_i)) \Big)$$

      Update $\phi$ by minimizing $\mathcal{L}_\pi$
      ~~**Temperature loss (optional tuning):**~~

$$\text{~~}\mathcal{L}_\alpha(\alpha) = \frac{1}{|B|}\sum_i \alpha\big(\text{~~}-\log \pi_\phi(a_i|s_i) - \bar{\mathcal{H}}\text{~~}\big)$$

      ~~Update $\alpha$ by minimizing $\mathcal{L}_\alpha$~~
      **Target update:** $\theta'_j \leftarrow \tau\, \theta_j + (1 - \tau)\, \theta'_j, \quad j \in \{1, 2\}$
    **end for**
    **if** $d_t$ **then**
      Update goal achievement: $g \leftarrow \lambda g + (1 - \lambda) g_{\text{e}}$, where $g_{\text{e}}$ is the episode goal achievement
      Update the $\sigma_{\text{LB}}$ based on the agent's performance: $\sigma_{\text{LB}} \leftarrow \sigma_{\text{LB}}(1 - g)$
      **break**
    **end if**
    $s_t \leftarrow s_{t+1}$
  **end for**
**end for**

**Algorithm 2** Proximal Policy Optimization (PPO) Algorithm with Gaussian Policy + ~~MaxEnt~~ GAGE

1: Initialize policy mean parameters $\theta_0$, policy standard deviation $\sigma_0$, value function parameters $\phi_0$, goal achievement $g_0 = 0$, goal achievement update factor $\lambda$
2: **for** iteration $k = 0, 1, 2, \ldots$ **do**
3: Collect set of trajectories $\{(s_t, a_t, r_t, s_{t+1})\}$ by running policy $\pi_{\theta_k}(a_t|s_t) = \mathcal{N}(\mu_{\theta_k}(s_t), \sigma_k^2)$ in the environment
4: **for** each time step $t$ **do**
5: Compute advantage estimates $\hat{A}_t$ based on value function $V_{\phi_k}(s_t)$
6: **end for**
7: Update the policy by maximizing the PPO-CLIP objective with an added entropy term:

$$\theta_{k+1}, \sigma_{k+1} = \arg\max_{\theta,\sigma} \mathbb{E}_t \Big[ \min \left( \frac{\mathcal{N}(\mu_\theta(s_t), \sigma^2)}{\mathcal{N}(\mu_{\theta_k}(s_t), \sigma_k^2)} \hat{A}_t,\ \text{clip} \left( \frac{\mathcal{N}(\mu_\theta(s_t), \sigma^2)}{\mathcal{N}(\mu_{\theta_k}(s_t), \sigma_k^2)}, 1-\epsilon, 1+\epsilon \right) \hat{A}_t \right)$$

$$\text{~~$+ \beta H(\pi_\theta(a_t|s_t))$~~} \Big]$$

where $\mu_{\theta_k}(s_t)$ is the mean of the Gaussian action distribution, $\sigma_k$ is the standard deviation (separately learned), and $H(\pi_\theta(a_t|s_t))$ is the entropy of the policy, encouraging exploration. The term $\beta$ controls the weight of the entropy regularization.
8: Update the value function by minimizing the following loss:

$$\phi_{k+1} = \arg\min_{\phi} \mathbb{E}_t \left[ (V_\phi(s_t) - R_t)^2 \right]$$

9: Update the goal achievement: $g_{k+1} = \lambda g_k + (1-\lambda) R$
10: Lower bound the standard deviation parameter $\sigma$ based on the agent's performance:

$$\sigma_{k+1} = \max(\sigma_{k+1}, -\sigma_0 g_k + \sigma_0)$$

11: **end for**

**Algorithm 3** Action Smoothing Algorithm

**Require:** Network outputs $\{z_1, z_2, \ldots, z_K\}$, goal achievement $g(\pi)$, hyperparameters $\delta_{z,0}$, $\delta_{z,1}$
**Ensure:** Action probabilities $p(a_k \mid s)$

1: **Calculate the original logit difference**:

$$\delta_z = \max_k(z_k) - \min_k(z_k).$$

2: **Compute the upper-bound** for the logit difference:

$$\delta_{z,\text{UB}} = (\delta_{z,1} - \delta_{z,0}) g(\pi) + \delta_{z,0}.$$

3: **Calculate the temperature**:

$$\tau(g, z) = \frac{\max(1, \delta_z)}{\delta_{z,\text{UB}}}.$$

4: **Compute action probabilities** using softmax with temperature:

$$p(a_k \mid s) = \frac{\exp(z_k / \tau(g,z))}{\sum_{i=1}^{K} \exp(z_i / \tau(g,z))}, \quad \text{for } k = 1, 2, \ldots, K.$$

## C Additional Results

In this section, we provide more detailed experiment results with continuous action spaces.

**$\sigma$ Schedule** To compare our method with exploration approaches using constant or linearly decreasing standard deviations, we conducted experiments on the Dog Balance Beam task. The agent was trained with constant $\sigma$ values of 0.25 and 0.75, as well as with linearly decreasing schedules ranging from 0.8 to 0.01 over $2.5 \times 10^7$ and $2.5 \times 10^8$ timesteps.

As shown in Fig. 9, only the agent with a linearly decreasing standard deviation similar to the curve discovered by our method achieved performance comparable to GAGE. This result further validates the effectiveness of our approach. Additionally, since tuning a predefined entropy schedule—considering both entropy values and training duration—is highly resource-intensive, our method significantly reduces the workload by introducing an adaptive schedule.

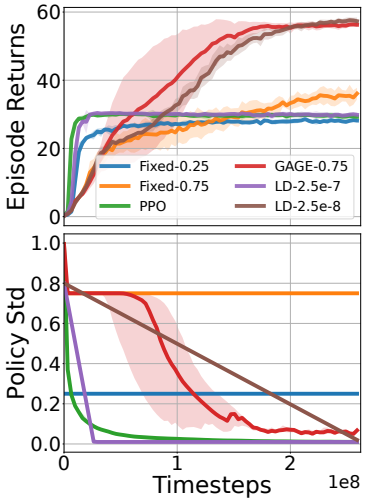
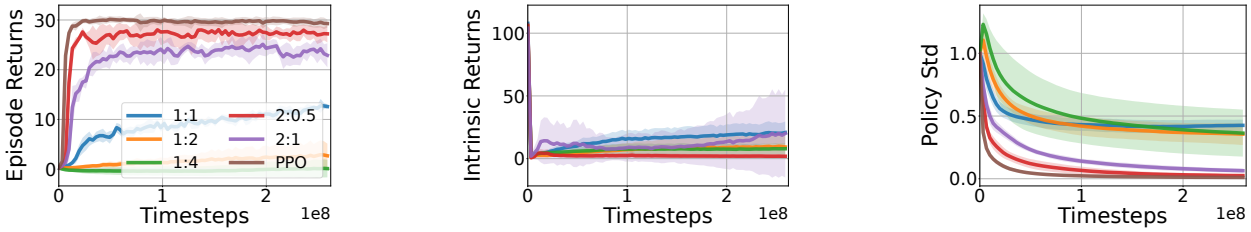



Figure 9: Additional training results of experiments with continuous action space in Dog Blanace Beam task with differerent $\sigma$-schedules. We plot the mean over 5 seeds. The faint area represents one standard deviation

**Intrinsic Reward Weight** To evaluate the effect of intrinsic rewards in the proposed challenging control tasks, we trained several RND agents using different weight combinations for extrinsic and intrinsic rewards: (2.0, 1.0), (2.0, 0.5), (1.0, 1.0), (1.0, 2.0), and (1.0, 4.0). The weight values (2.0, 1.0) are consistent with those used in the original RND work (Burda et al., 2019) and subsequent research (Yang et al., 2024). Therefore, we also used this ratio for the experiments presented in Fig. 2. As shown in Fig. 10, none of the RND agents succeeded in solving the task. Agents with larger ratios of extrinsic-to-intrinsic weights exhibited learning patterns similar to standard PPO, which does not use intrinsic rewards. As the ratio decreased, the agents focused more on exploring novel states, as indicated by larger standard deviations during training. However, this increased exploration did not contribute to solving the task. Instead, the novelty-based exploration resulted in decreased extrinsic rewards. This phenomenon highlights the distinct focus of our work compared to novelty-based exploration methods. Our work focuses on addressing premature convergence, an issue that is equally important but has been largely overlooked until now. In contrast, curiosity-based methods primarily tackle sparse rewards. The difference in focus is also reflected in the existing benchmarks for exploration algorithms. Most environments are designed with sparse rewards and moderate local optima, which can be effectively addressed using novelty-based exploration. For example, environments like Fetch (Plappert et al., 2018), MiniGrid (Chevalier-Boisvert et al., 2023), AntMaze, and Adroit manipulation tasks (Fu et al., 2020) are "safe," with sparse termination states or penalties distributed across the state space. Agents can easily avoid termination and penalty states while exploring for rewards. In such environments, exploring unseen states is a highly effective strategy. However, novelty-based methods struggle in scenarios with more severe local optima. For instance, Noisy-TV has been recognized as a major issue for novelty-based methods, even though it only involves local optima introduced by environment stochasticity. The challenges posed by more severe local optima have not yet been fully explored. In this work, we aim to push the boundaries of RL exploration research into environments with more challenging local optima issues. The proposed IsaacLab tasks reflect real-world robot control scenarios where optimal behaviors occupy only a small portion of the state space, while most of the state space leads to penalties such as falling down or wasting energy. This dominant penalizing space creates challenging local optima. In such environments, novelty-based exploration often results in sampling mostly failed trajectories and becoming trapped in local optima.




(a) Extrinsic reward (b) Intrinsic reward (c) Standard deviation

Figure 10: Investigating the effect of novelty-based intrinsic reward to the learning of Dog Balance Beam task. The curves with legend 1:2 represent the agent trained using extrinsic and intrinsic coefficients of (1.0, 2.0).

Table 1: Reward weights of continuous control tasks. The rewards and penalties from left to right are for desired locomotion velocity, environment not terminating, robot orientation, robot distance to the manipulated object, large action commands, energy consumption, joint position too close to limitations, robot velocity perpendicular to the desired direction, object velocity perpendicular to the desired direction, joint torque, joint acceleration, and action changing rate. The selected goal reward for goal achievement calculation is marked in green background.

| | **reward** | | | | **penalty** | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $v_x$ | alive | orient | $d_{\text{obj}}$ | $\|a\|_2^2$ | $E$ | $\theta_{\text{limit}}$ | $v_y$ | $v_{y,\text{obj}}$ | $T$ | $\ddot{\theta}$ | $\dot{a}$ |
| HT | 0.5 | 1.0 | 1.0 | 0 | 0.01 | 0.05 | 0.25 | 1.0 | 0 | 0 | 0 | 0 |
| HD | 0.3 | 0.4 | 1.0 | 0.2 | 0.01 | 0.01 | 0.25 | 0 | 0.5 | 0 | 0 | 0 |
| HC | 2.0 | 1.0 | 0.5 | 0 | 0.01 | 0.05 | 0.25 | 0 | 0 | 0 | 0 | 0 |
| DB | 1.0 | 1.0 | 1.0 | 0 | 0.005 | 0 | 0 | 1.0 | 0 | 1e-6 | 2.5e-8 | 0.001 |
| AA | 1.0 | 1.0 | 1.0 | 0 | 0.005 | 0.05 | 0.1 | 0 | 1.0 | 0 | 0 | 0 |

# D Experimental Details

## D.1 Tasks Setup

We design five challenging continuous control tasks in IsaacLab. Three robots with many degrees of freedom learn challenging locomotion or dynamic manipulation behaviors. The robots include a humanoid robot with 21 joints, a dog robot (Unitree Go2) with 12 joints, and an ant robot with 8 joints. In Table 1, we provide the reward composition of different tasks.

**Humanoid Tightrope (HT)** The humanoid robot learns side walking for 2m/s on a tightrope, i.e., a cylindrical bar with a diameter of only 0.1m. This is more challenging than walking forward because balancing with two arms stretched to both sides would be more difficult.

**Humanoid Dribbling (HD)** The humanoid robot learns to dribble a football at a high speed (3.5m/s). Additionally, the robot gets random commands for turning the target direction for up to $\frac{\pi}{4}$rad.

**Humanoid Cartwheel (HC)** The humanoid robot learns to perform cartwheel at a speed of 6rad/s. It requires precisely coordinated movement of the whole body, which is more difficult than normal locomotion tasks, which primarily rely on the lower body of the robot.

**Dog Balance Beam (DB)** The dog robot learns to walk on a balance beam at a speed of 2m/s. The beam has a square cross-section with 0.1m side length. Moreover, the balance beam is tilted for $\frac{\pi}{9}$rad so that the robot has to climb a slope while balancing.

**Ant Acrobatics (AA)** The ant robot with four legs learns to balance a pole vertically on its torso while standing on a ball. The pole has a length of 2m. The ball has a diameter of the same value. Moreover, the robot has to learn to roll the ball forward at a target speed of 1m/s.

### D.2 Hyper-Parameters Optimization and Implementation

**Default Hyperparameter for PPO** We follow the implementation in RSL-RL[1] PPO and reuse their default hyperparameters in the vanilla PPO for continuous control tasks. GAGE shares hyperparameters with vanilla PPO except for its own hyperparameters. We include the default parameters in Table 4.

Table 2: Tuned hyperparameters for PPO on different tasks

| | Action Type | AA | HD | HC | HT | DB |
|---|---|---|---|---|---|---|
| entropy coef | continuous | 1.39e-2 | 7.2e-4 | 1.93e-3 | 5.18e-3 | 1e-4 |
| | discrete | 5.18e-3 | 5.18e-3 | 1.93e-3 | 5.18e-3 | 5.18e-3 |

**Hyperparameter Optimization for PPO** GAGE is addressing the premature convergence by adaptively changing the lower bound of the policies' standard deviation. To obtain a strong baseline to compare in our tasks, we perform a grid search for the entropy coefficient of each continuous or discrete task. We evaluate 8 different entropy coefficients uniformly in log-scale from $1 \times 10^{-4}$ to 0.1 with 5 seeds. The top-performing hyperparameter setting will then be used for the tuned PPO baseline. We select the best-performing hyperparameter by averaging the last episode return over 5 runs with different seeds. Detailed training curves can be found in Fig. 11. The resulting entropy coeffcients are summarized in Table 2.

**Hyperparameter for continuous PPO-GAGE** We use Proximal Policy Optimization (PPO) as the backbone algorithm for the IsaacLab experiments. We adjust the implementation of rsl_rl v2.0.0 according to Algorithm 3. Except for deactivating the MaxEnt term for PPO-GAGE by setting the entropy temperature as zero, we have not changed any hyperparameters. They are kept the same for all agents for a fair comparison (see Table 4).

Table 3: Tuned hyperparameters for PPO-GAGE on different tasks with discrete actions

| | AA | HD | HC | HT-F | DB |
|---|---|---|---|---|---|
| $\delta_{z,0}$ | -4 | -4 | -3 | -4 | -5 |
| $\delta_{z,1}$ | -20 | -20 | -20 | -15 | -15 |
| $\alpha_{\text{logit}}$ | 0.1 | 0.01 | 0.1 | 0.01 | 0.01 |

**Hyperparameter for discrete PPO-GAGE** We discretize each action dimension into 11 bins. Other hyperparameters are the same as the continuous PPO implementation. We have three hyperparameters, $(\delta_{z,0}, \delta_{z,1}, \alpha_{\text{logit}})$, in the discrete PPO-GAGE implementation. We implement grid-search optimization for each environment in ranges $[3, 4, 5], [10, 15, 20], [0.001, 0.01, 0.1]$. We list the best performing hyperparameters in Table 3.

**Hyperparameter for SAC-GAGE** For the experiments with Humanoid-Bench, we implement the algorithm based on the code provided by Sukhija et al. (2025)[2]. We deactivate the MaxEnt term by setting the entropy temperature as zero. Other hyperparameters are the same as those provided in the code to ensure a fair comparison with the baseline algorithms (see Table 5). For SAC-GAGE we have three hyperparameters, $(\sigma_0, \alpha_{\text{mean}}, \hat{G})$. We employ grid-search optimization in ranges $[0.5, 0.75, 1.0], [0.001, 0.01, 0.1], [800, 1200]$. We demonstrate the learning curves with $(\sigma_0 = 1.0, \alpha_{\text{mean}} = 0.01, \hat{G} = 1200)$ in Fig. 6.

[1] https://github.com/leggedrobotics/rsl_rl

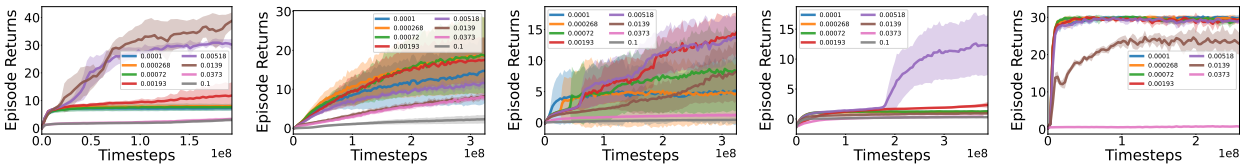
[2] https://github.com/swan-utokyo/deir

Table 4: Common hyperparameters for PPO(-GAGE)

| Hyperparameter | Value |
|---|---|
| **Algorithm** | |
| Value loss coefficient | 1.0 |
| Clip parameter ($\epsilon$) | 0.2 |
| Use clipped value loss | True |
| Desired KL divergence | 0.01 |
| Discount factor ($\gamma$) | 0.99 |
| GAE parameter ($\lambda$) | 0.95 |
| Max gradient norm | 1.0 |
| Learning rate | 0.0005 |
| Number of learning epochs | 5 |
| Number of mini-batches | 4 |
| Learning rate schedule | Adaptive |
| **Policy** | |
| Activation function | ELU |
| Actor hidden dimensions | [400, 200, 400] |
| Critic hidden dimensions | [400, 200, 100] |
| Initial noise standard deviation | 1.0 |
| **Runner** | |
| Number of steps per environment | 24 |
| Number of environments | 1024 |
| Empirical normalization | False |
| **RND** | |
| Intrinsic Reward coefficient | 1 |
| Extrinsic Reward coefficient | 2 |
| Intrinsic Reward Normalization | yes |

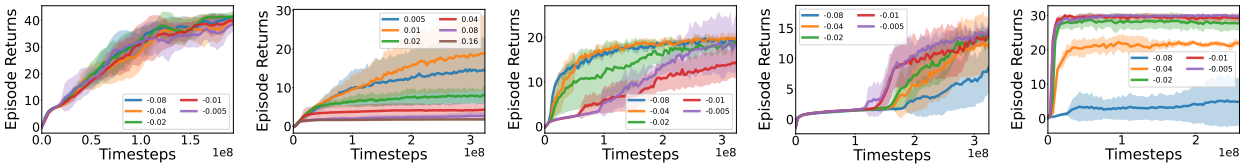
Table 5: Hyperparameters for SAC & MaxInfoSAC & SAC-GAGE

| **Hyperparameter** | **SAC/tuned** | **MaxInfoSAC/tuned** | **SAC-GAGE** |
|---|---|---|---|
| hidden dimensions | [512, 512] | [512, 512] | [512, 512] |
| discount ($\gamma$) | 0.99 | 0.99 | 0.99 |
| tau | 0.005 | 0.005 | 0.005 |
| target update period | 1 | 1 | 1 |
| target entropy | $-\dim(\mathcal{A})$ | $-\dim(\mathcal{A})$ | |
| backup entropy | true | true | |
| actor lr | 0.0005 | 0.0005 | 0.0005 |
| critic lr | 0.0005/0.0001 | 0.0005/0.0001 | 0.0001 |
| temperature lr | 0.0005 | 0.0005 | |
| initial temperature | 1.0 | | |
| ensemble lr | | 0.0005 | |
| information gain temperature lr | | 0.0005 | |
| gradient norm clip | None/1.0 | None/1.0 | 1.0 |



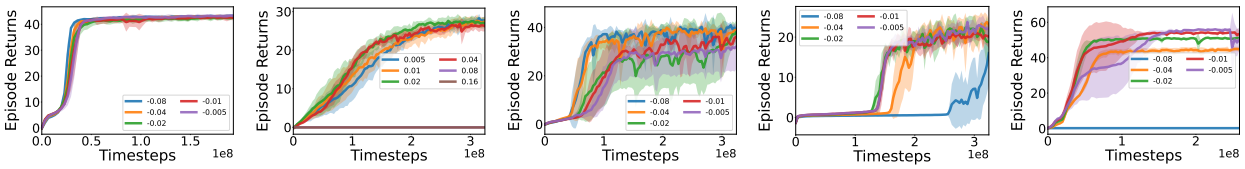

(a) Ant Acrobatics (b) Dribbling (c) Cartwheel (d) Tightrope (e) Balance Beam

Figure 11: Different Entropy Coefficient




(a) Ant Acrobatics (b) Dribbling (c) Cartwheel (d) Tightrope (e) Balance Beam

Figure 12: Training curves of different L2 action penalty coefficients for PPO in continuous control tasks




(a) Ant Acrobatics (b) Dribbling (c) Cartwheel (d) Tightrope (e) Balance Beam

Figure 13: Training curves of different L2 action penalty coefficients for GAGE in Continuous Control Tasks