Adaptive Horizon Actor-Critic for Policy Learning in Contact-Rich Differentiable Simulation

Ignat Georgiev [1] **Krishnan Srinivasan** [2] **Jie Xu** [3] **Eric Heiden** [3] **Animesh Garg** [1 3 4]

Abstract

Model-Free Reinforcement Learning (MFRL), leveraging the policy gradient theorem, has demonstrated considerable success in continuous control tasks. However, these approaches are plagued by high gradient variance due to zeroth-order gradient estimation, resulting in suboptimal policies. Conversely, First-Order Model-Based Reinforcement Learning (FO-MBRL) methods employing differentiable simulation provide gradients with reduced variance but are susceptible to sampling error in scenarios involving stiff dynamics, such as physical contact. This paper investigates the source of this error and introduces Adaptive Horizon Actor-Critic (AHAC), an FO-MBRL algorithm that reduces gradient error by adapting the model-based horizon to avoid stiff dynamics. Empirical findings reveal that AHAC outperforms MFRL baselines, attaining 40% more reward across a set of locomotion tasks and efficiently scaling to high-dimensional control environments with improved wall-clock-time efficiency. adaptive-horizon-actor-critic.github.io

1. Introduction

The Policy Gradients Theorem (Sutton et al., 1999) has enabled the development of Model-Free Reinforcement Learning (MFRL) approaches for solving continuous motor control tasks. Although these methods have achieved impressive results (Hwangbo et al., 2017; Akkaya et al., 2019; Hwangbo et al., 2019), they are hampered by high gradient variance leading to unstable learning and suboptimal policies (Mohamed et al., 2020), as well as subpar sample efficiency (Amos et al., 2021). The latter can be circumvented via the use of efficient vectorized physics simulators. These simulators, when integrated with efficient MFRL methods, facilitate rapid training such as learning quadruped walking in minutes (Rudin et al., 2022). However, the effectiveness of MFRL in addressing motor control challenges, even with extensive data, remains questionable.

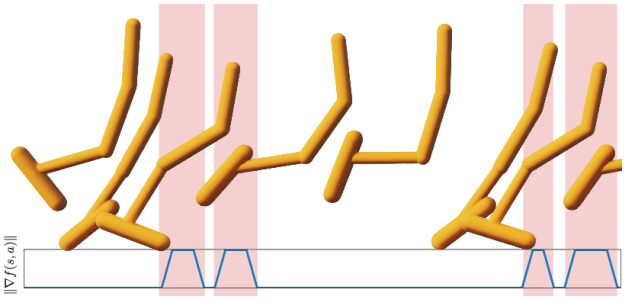

Figure 1. **Overview.** We find that First Order Model-Based RL (FO-MBRL) methods suffer from erroneous gradients arising from stiff dynamics ($\|\nabla f(s,a)\| \gg 0$). Our proposed method, *AHAC*, truncates model-based trajectories at the point of contact, avoiding both the gradient sample error and learning instability exhibited by previous methods using differentiable simulation.

An alternative, Model-Based Reinforcement Learning (MBRL), focuses on learning environmental dynamics to enhance sample efficiency and facilitate novel methods of policy optimization. Recent MBRL research has introduced innovative dynamics models and policy learning techniques, but often without extensive independent evaluation of each component (Hafner et al., 2019b; 2023; Hansen et al., 2023).

When a dynamics model is available, one could employ first-order methods for policy learning, deemed theoretically more efficient (Mohamed et al., 2020; Berahas et al., 2022). This approach has been investigated in model-based control, where dynamics models guide trajectory planning (Kabzan et al., 2019; Kaufmann et al., 2020). However, using first-order methods to learn feedback policies within typical MBRL frameworks is less explored. This paper aims to evaluate First-Order MBRL (FO-MBRL), concentrating on policy learning and utilizing differentiable simulation for dynamics modeling.

Where model-based control literature often designs bespoke models for each problem, differentiable simulation aims to create a physics engine that is fully differentiable (Hu

[1]Georgia Institute of Technology [2]Stanford University [3]Nvidia [4]University of Toronto. Correspondence to: Ignat Georgiev, Animesh Garg <{ignat, animesh.garg}@gatech.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

et al., 2019a; Freeman et al., 2021; Heiden et al., 2021; Xu et al., 2021). Thus, applying it to a different problem is similar to using a different definition of the environment in the simulation setup (e.g., joints and links) and leaving the physics to be calculated by the engine. Short Horizon Actor-Critic (SHAC) (Xu et al., 2022) is an FO-MBRL approach leveraging differentiable simulation and the popular actor-critic paradigm (Konda & Tsitsiklis, 1999). The actor is trained in a first-order fashion, while the critic is trained model-free. This allows SHAC to learn through the highly non-convex landscape by using the critic as a smooth surrogate of the cumulative reward. While SHAC demonstrates impressive sample efficiency, it also faces challenges such as brittleness, learning instability, and dependency on extensive hyper-parameter tuning.

This study addresses these issues, shifting the focus from sample efficiency to the asymptotic performance of FO-MBRL methods in massively parallel differentiable simulation. Our analysis indicates that first-order methods suffer from significant sampling error in gradient estimation, primarily due to high dynamical gradients from stiff contact approximation ($k r f(s;a)k \quad 0$) (Suh et al., 2022; Lee et al., 2023), leading to inefficiency and suboptimal policies. To address this, we propose Adaptive Horizon Actor-Critic (AHAC), a FO-MBRL algorithm that adjusts its trajectory rollout horizon to circumvent stiff dynamics (Figure 1). Experimentally, our method shows superior asymptotic performance over MFRL baselines in complex locomotion tasks, achieving up to 64% higher reward even when baselines are given 10^6 times more training data. Further, AHAC's efficient use of first-order gradients enables scaling to high-dimensional motor control tasks with 152 action dimensions.

2. Preliminaries

This study focuses on discrete-time and finite-horizon reinforcement learning scenarios characterized by system states $s \in \mathrm{R}^n$, actions $a \in \mathrm{R}^m$, and deterministic dynamics described by the function $f : \mathrm{R}^n \quad \mathrm{R}^m \to \mathrm{R}^n$. Actions at each timestep t are sampled from a tanh-transformed stochastic policy $a_t \quad (\cdot|s_t)$, parameterized by $\in \mathrm{R}^d$, and yield rewards from $r : \mathrm{R}^n \quad \mathrm{R}^m \to \mathrm{R}$. The H-step return is defined as:

$$R_H(s_1; \ ) = \sum_{h=1}^{H} r(s_h; a_h)$$
$$s.t: \quad s_{h+1} = f(s_h; a_h) \quad a_h \quad (\cdot|s_h)$$

The policy's objective is to maximize the cumulative reward:

$$\max J(\ ) := \max \mathrm{E}_{\substack{s_1 \\ a_h \quad (\cdot|s_h)}} [R_H(s_1)] \qquad (1)$$

where is the initial state distribution. Without loss of generality, we simplify our derivations:

Assumption 2.1. *is a dirac-delta distribution.*

Similar to prior work Duchi et al. (2012); Berahas et al. (2022); Suh et al. (2022), we are trying to exploit the smoothing properties of stochastic optimization on the landscape of our optimization objective. Following recent successful deep-learning approaches to MFRL (Schulman et al., 2017; Haarnoja et al., 2018), we assume that our policy is stochastic, parameterized by and expressed as $(\cdot|s)$.

To address the main optimization problem in Equation 1, we consider stochastic gradient estimates of $J(\ )$ using zero-order and first-order methods. To guarantee the existence of $r J(\ )$, we need to make certain assumptions:

Definition 2.2. *A function $g : \mathrm{R}^d \to \mathrm{R}^d$ has polynomial growth if there exists constants $a; b$ such that $\forall z \in \mathrm{R}^d$, $\|g(z)\| \quad a(1 + \|z\|^b)$.*

Assumption 2.3. *To ensure gradients are well defined, we assume that the policy $(\cdot|s)$ is continuously differentiable $\forall s \in \mathrm{R}^n; \forall \in \mathrm{R}^d$. Furthermore, the system dynamics f and reward r have polynomial growth.*

2.1. Zeroth-Order Batch Gradient (ZOBG) estimates

These weak assumptions are sufficient to make $J(\ )$ differentiable in expectation by taking samples of the function value in a zeroth-order fashion (Williams, 1992). This gives estimates of $r J(\ )$ via the stochasticity introduced by , as first shown in (Williams, 1992), and commonly referred to as as the *Policy Gradient Theorem* (Sutton et al., 1999).

Definition 2.4. *Given a sample of the H-step return $R_H(s_1) = \sum_{h=1}^{H} r(s_h; a_h)$ following the policy , we can estimate zero-order policy gradients via:*

$$r^{[0]} J(\ ) := \mathrm{E}_{a_h \quad (\cdot|s_h)} \left[R_H(s_1) \sum_{h=1}^{H} r \ \log \ (a_h|s_h) \right] \qquad (2)$$

Lemma 2.5. *Under Assumptions 2.1 and 2.3, the ZOBG is an unbiased estimator of the stochastic objective $\mathrm{E}\ r^{[0]} J(\ ) = r J(\ )$ where $r^{[0]} J(\ )$ is the sample mean of N Monte Carlo estimates of Eq. 2.*

These zero-order policy gradients are known to have high variance (Mohamed et al., 2020), and one way to reduce their variance is by subtracting a baseline from the function estimates. Similar to (Suh et al., 2022), we subtract the return given by the noise-free policy rollout where $R_H(s_1) \quad R_H(s_1)$ is used instead of $R_H(s_1)$ in Eq. 2.

2.2. First-Order Batch Gradient (FOBG) estimates

Given access to a differentiable simulator, first-order gradients induced by the policy can be computed via:

$$r^{[1]} J(\ ) := \mathrm{E}_{a_h \quad (\cdot|s_h)} [r \ R_H(s_1)] \qquad (3)$$

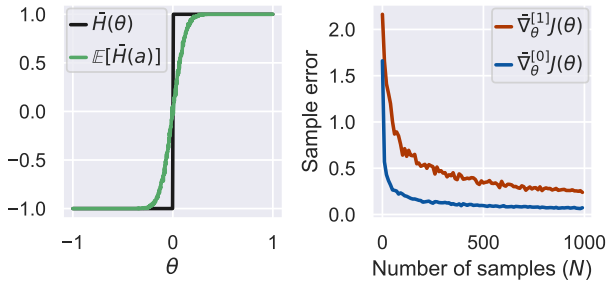

Figure 2. The left figure shows the **Soft Heaviside of Eq 4**. The right figure shows the gradient sample error. We observe that FOBG estimates with finite N exhibit a higher sample error.

However, for these gradients to be well-defined, we need to make further assumptions:

Assumption 2.6. *The dynamics* $f(s; a)$ *and the reward* $r(s; a)$ *are continuously differentiable* $8s \; 2 \; \mathrm{R}^n; 8a \; 2 \; \mathrm{R}^m$.

Although these assumptions are necessary for the analysis of the next section, we relax them in our experiments section and consider contemporary benchmarks.

3. Policy learning through contact

Prior research has established that first-order gradients are statistically unbiased (Schulman et al., 2015). However, the sample error under finite N is heavily dependent on the function they are trying to approximate, referred to as "empirical bias" (Suh et al., 2022; Lee et al., 2023). This paper explores this sampling error using the soft Heaviside function, an approximation of the Coulomb friction model, which is pivotal in discontinuous function analysis within physics simulations:

$$H(x) = \begin{cases} 1 & x > \quad =2 \\ 2x = & jxj \quad =2 \\ 1 & x < \quad =2 \end{cases} \tag{4}$$

where $a \quad ( ) = \quad + w$ and $w \quad N(0; \; ^2)$. As shown in Appendix A, $\mathrm{E} \quad H(a)$ is a sum of error functions whose derivative $r \quad \mathrm{E} \quad H(a) \quad 6= 0$ at $\quad = 0$. However, using FOBG, we obtain $r \quad H(a) = 0$ in samples where $jaj > \quad =2$, which occurs with probability at least $= \quad \frac{p}{2}$. Since in practice we are limited in sample size, this translates to sampling error that is inversely proportional to sample size, as shown in Figure 2. Notably, when $\quad ! \; 0$, we achieve a more accurate approximation of the underlying discontinuous function, but we also increase the likelihood of obtaining erroneous FOBG, thus amplifying error in stochastic scenarios. We use this particular example as the differentiable simulator used in our experiments is based on the Coulomb friction model.

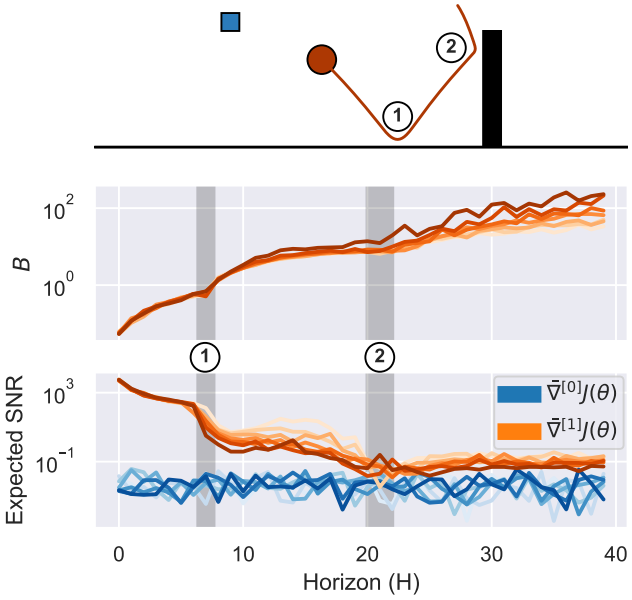

Figure 3. **Toy example where a ball is shot against a wall** trying to reach the target position in blue. The bottom two figures show gradient sample error and Expected SNR estimation with $N = 1024$ samples. Darker shades designate point of contact, which negatively impact FOBG error. Higher ESNR leads to more informative gradients.

Definition 3.1. *FOBGs exhibit sampling error relative to ZOBGs under finite samples, denoted as* B*:*

$$B = \quad r^{[1]} J( ) \quad r^{[0]} J( )$$

We analyze B from the perspective of bias and variance to derive a practical upper bound:

Lemma 3.2. *For an H-step stochastic optimization problem under Assumptions 2.6, which also has Lipshitz-smooth policies* $k r \quad (ajs)k \quad B$ *, Lipshitz-smooth reward function in both arguments* $k r r(s; a)k \quad B_r$ *and Lipshitz-smooth dynamics in both arguments* $k r f(s; a)k \quad B_f$ $8s \; 2 \; \mathrm{R}^n; a \; 2 \; \mathrm{R}^m; \quad 2 \; \mathrm{R}^d$*, then ZOBGs remain consistently unbiased. However, FOBGs exhibit sample error bounded by:*

$$B \quad HB_r B \quad \frac{1}{2} + B_f^H \quad ^1 \tag{5}$$

The proof can be found in Appendix B

As $r(s; a)$ and $\quad$ are often design decisions in problems, we can create them to satisfy the assumptions laid out above. However, bounding the dynamics $jj r f(s_t; a_t)jj$ is impossible due to the natural discontinuities of physics (Lee et al., 2023) leading to $B_f \quad B_r$ and $B_f \quad B$. This combined with the H terms lead the two conclusions from Lemma 3.2: (1) long-horizon rollout lead to increased FOBG sample error and (2) prolonged stiff contact has compound effects on FOBG sample error.

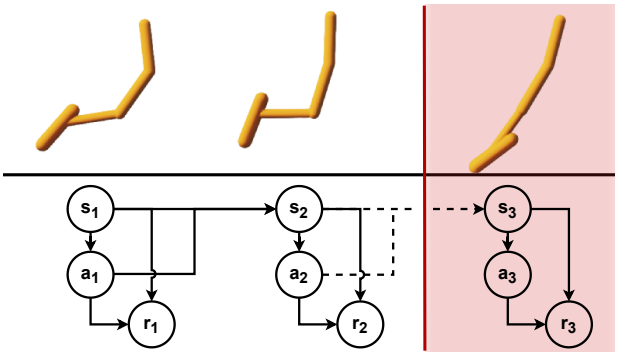

Figure 4. **Example $H = 3$ step trajectory** where s_3 is in contact at which point the trajectory is truncated. When optimizing this trajectory, we completely omit the stiff dynamics gradient $\nabla f(s_2, a_2)$ leading to stabler and less erroneous FOBGs.

Empirical evaluation of Lemma 3.2. We designed a simple experimental setup involving a ball rebounding off a wall to reach a target, as illustrated in Figure 3. The initial position $s_1 = [x_1; y_1]$ and velocity of the ball are fixed. The objective is for the policy to learn the optimal initial orientation in order to reach a target position s_T at the end, defined as $R_H(s_1) = k s_H \quad s_T k_2^{\ 1}$. We use the additive Gaussian policy $a = \quad + w$, where $w \quad N(0; \ ^2)$. With this, zero-order gradients from Eq. 2 can be expressed as:

$$r^{[0]} J(\ ) \quad \frac{1}{N^{\ 2}} \sum_{i=1}^{N} R_H^{(i)}(s_1) \quad R_H^{(i)}(s_1) \ w^{(i)}$$

We collect $N = 1024$ samples of each gradient type for each timestep with $H = 40$. Figure 3 shows that the sample error remains low until the ball encounters contact, after which it starts growing, validating our proposed lemma. Additionally, the error also affects the gradient variance, where ZOBG follow $\text{Var}\ r^{[0]} J(\ ) \quad {}^2 H B_f^2 B^2$ (Suh et al., 2022). However, FOBG variance behaves similarly to Lemma 3.2, growing exponentially after contact. In Figure 3, instead of variance, we show Expected SNR (Eq. 6) as proposed by (Parmas et al., 2023), with higher values translating to more informative gradients. These results suggest that FOBGs exhibit sample error under contact dynamics, which is further worsened with long trajectories (Lee et al., 2023; Zhong et al., 2023).

$$\text{ESNR}(r J(\ )) = \text{E}\left[\frac{\sum \text{E}[r J(\ )]^2}{\sum \text{Var}[r J(\ )]} \right] \tag{6}$$

4. Adaptive Horizon Actor-Critic (AHAC)

4.1. Learning through contact in a single environment

With a clearer understanding of the influence of stiff contact, we aim to develop a First-Order MBRL approach for contact-rich continuous control tasks. Unlike the toy example of the previous section, standard RL multi-step decision processes allow for the avoidance of stiff dynamics gradients using *contact truncation*. Consider the example shown in Figure 4. Truncating the trajectory at the point of contact yields reward gradients without the gradient of stiff dynamics (striked out in red):

$$r \quad r(s_3; a_3) = r_{\ a_3} r(s_3; a_3)\, r \qquad (a_3 | s_3)$$
$$+ \ \cancel{r_{\ s_3} r(s_3; a_3)\, r \quad f(s_2; a_2)}$$

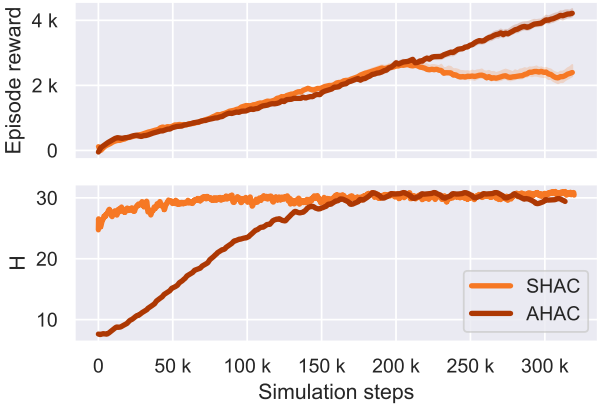

Figure 5. **Comparison between SHAC and AHAC-1 on the Hopper task with only a single environment**. The figure shows rewards and horizons achieved over 5 different random seeds, with the 50% IQM plotted. Note that both algorithms have some horizon oscillation due to the early termination mechanism of the simulator, as noted in Appendix F.

We introduce an FO-MBRL algorithm with an actor-critic architecture, akin to SHAC (Xu et al., 2022). The critic, denoted as $V\ (s)$, is model-free and trained using TD( ) (Sutton & Barto, 2018) over an H-step horizon:

$$R_h(s_t) := \sum_{n=t}^{t+h \ 1} \ ^{n \ t} r(s_n; a_n) + \ ^{t+h} V\ (s_{t+h})$$

$$\hat{V}(s_t) := (1 \quad) \sum_{h=1}^{H \ 1} \ ^{h \ 1} R_h(s_t) \ + \ ^{H \ t \ 1} R_H(s_t)$$

The critic loss becomes $L_V(\ )$, while the actor is trained using FOBG as in Equation 3, with the addition of the critic value estimate:

$$L_V(\ ) := \sum_{h=t}^{t+H} V\ (s_h) \quad \hat{V}(s_h) \ ^2_2 \tag{7}$$

$$J(\ ) := \sum_{h=t}^{t+H \ 1} \ ^{h \ t} r(s_h; a_h) + \ ^H V\ (s_{t+T}) \tag{8}$$

Unlike fixed-horizon model-based rollouts in (Xu et al., 2022), our policy is rolled out until stiff contact is detected in simulation, leading to a dynamic horizon adjustment to prevent gradient explosion. However, not all contact results in high error; therefore, we truncate only on stiff

Figure 6. An ablation of short horizons H for the SHAC algorithm applied to Ant. Each run is trained until convergence for 5 seeds. The reward peaked and exhibited the least variance when the horizon length approximated the optimal gait period of 28.

contact $\|\frac{\partial f}{\partial s_t}(s_t, a_t)\| > C$, where C is the contact stiffness threshold. We refer to this algorithm as Adaptive Horizon Actor-Critic 1 (AHAC-1) (see Appendix D).

AHAC-1's performance was tested in a toy locomotion environment. We re-implement the popular Hopper task, where a single-legged agent hops in one axis and is rewarded for high forward velocity (Figure 1). Compared to SHAC, which employs a fixed horizon of $H = 32$, AHAC-1 adjusts its horizon based on a contact stiffness threshold of $C = 500$. Results in Figure 5 indicate that AHAC-1 achieves a higher reward than SHAC. We believe that the more erroneous SHAC gradients steer it towards local minima, while our proposed approach manages to circumvent them and achieve a higher asymptotic reward. However, AHAC-1 is not applicable to parallel vectorized environments due to the challenge of asynchronous trajectory truncation, which leads to in nitely long compute graphs.

4.2. Scaling learning with synchronous parallelization

To address the issue of asynchronous truncation, we explored the short-horizon methodology of SHAC, incorporating graph truncation at stiff contacts. However, this method did not improve performance, likely due to gradient variances across differing trajectory lengths. Consequently, our research pivoted to examine the effect of horizon length H on policy optimality, especially in contact-based tasks like locomotion that demand speci c gait patterns.

We conducted empirical tests using the SHAC algorithm on the Ant locomotion task, where the goal for a quadruped robot is to maximize forward velocity. By altering the horizon length H in SHAC, our ndings in Figure 6 reveal a correlation between gait period and horizon length, with optimal performance at $H = 28$.

Two key insights emerged from this study: (1) each task possesses an inherent optimal model-based horizon length H, closely linked to the gait period; (2) the optimal horizon correlates with the highest reward and lowest variance, aligning with the ndings of Lemma 3.2. These insights informed the development of a generalized, GPU-parallelized version of AHAC-1, termed AHAC. While retaining the same critic training methodology as outlined in Equation 7, AHAC introduces a novel constrained objective for the actor.

$$J(\theta) := \sum_{h=t}^{t+H-1} \gamma^{h-t} r(s_h, a_h) + \gamma^H V_\phi(s_{t+H}) \quad \text{s.t.:} \quad \|\frac{\partial f}{\partial s_t}(s_t, a_t)\| \le C \quad \forall t \in \{0, \dots, H\} \tag{9}$$

The objective seeks to maximize the reward while ensuring that all contact stiffness remains below the threshold. Using the Lagrangian formulation, we derive the dual problem:

$$L(\theta, \lambda) = \sum_{h=t}^{t+H-1} \gamma^{h-t} r(s_h, a_h) + \gamma^H V_\phi(s_{t+H}) + \lambda^T \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} - \begin{bmatrix} C - \|\frac{\partial f}{\partial s_t}(s_t, a_t)\| \\ \vdots \\ C - \|\frac{\partial f}{\partial s_{t+H}}(s_{t+H}, a_{t+H})\| \end{bmatrix} \tag{10}$$

By de nition, $\lambda_i = 0$ if the constraint is met and $\lambda_i > 0$ otherwise. Thus, λ is used to adapt the horizon, resulting in the full AHAC shown in Algorithm 1. Additionally, we introduce a double critic that is trained until convergence, de ned as a small change in the last 5 critic training iterations, $\sum_{i=n-5}^{n} L(\phi) < 0.2$, where we take mini-batch samples from the rollout buffer $(s, \hat{V}(s)) \in \mathcal{D}$.

Algorithm 1 Adaptive Horizon Actor-Critic

Given: α, β, γ: learning rates
Given: C: contact threshold
Initialize learnable parameters θ, ϕ, H, $\lambda = 0$
$t \leftarrow 0$
while episode not done do
  Initialize rollout buffer $\mathcal{D}$
  for $h = 0, 1, \dots, H$ do ▷ rollout policy
    $a_{t+h} \sim \pi_\theta(a_{t+h} | s_{t+h})$
    $r_{t+h} = r(s_{t+h}, a_{t+h})$
    $s_{t+h+1} = f(s_{t+h}, a_{t+h})$
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_{t+h}, a_{t+h}, r_{t+h}, V_\phi(s_{t+h+1}))\}$
  end for
  $\theta \leftarrow \theta + \alpha \nabla_\theta L_\theta(\theta, \lambda)$ ▷ train actor (Eq. 10)
  $\lambda \leftarrow \lambda - \beta \nabla_\lambda L_\theta(\theta, \lambda)$
  $H \leftarrow H + \sum_{h=0}^{H} \lambda_h$
  while not converged do ▷ train critic (Eq. 7)
    sample $(s, \hat{V}(s)) \in \mathcal{D}$
    $\phi \leftarrow \phi - \alpha \nabla_\phi L_V(\phi)$
  end while
  $t \leftarrow t + H$
end while

Figure 7. Experimental Environments. Locomotion tasks in increasing order of action space dimensions (left to right): Hopper ($m = 3$), Ant ($m = 8$), Anymal ($m = 12$), Humanoid ($m = 21$) and SNU Humanoid ($m = 152$).

In practice, truncating based solely on the gradient of dynamics $\nabla f(s_t, a_t)$ proved restrictive due to the variability of contact forces across different tasks and their evolution during the learning process. To address this, we introduced a normalization method for contact forces, utilizing modified acceleration per state dimension $q_t = \max(|\dot{q}_t|, 1)$ applied element-wise, resulting in normalized contact forces $\hat{\nabla} f(s_t, a_t) = \text{diag}(q_t)^{-1} \nabla f(s_t, a_t)$. Notably, this allows using a uniform contact threshold ϵ_C across different tasks.

Furthermore, considering that contact approximation forces are calculated separately in differentiable simulators, there is no need to use the full dynamics Jacobian. Instead, we employ the Jacobian, derived solely from contact forces. The differences between the SHAC and AHAC algorithms are comprehensively delineated in Appendix C.

5. Experiments

The objectives of this section are to (1) assess AHAC's ability to obtain higher asymptotic reward than MFRL baselines; (2) its efficiency in terms of wall-clock time and scalability to high-dimensional environments, and (3) identify the key contributing components of AHAC.

Setup. We evaluate the proposed approach, AHAC, across a set of 5 contemporary locomotion tasks, ranging from the simpler *Hopper* with $n = 11$ and $m = 3$, to the more complex *SNU Humanoid*, which features a muscle-actuated humanoid lower body with $m = 152$ (Figure 7). All tasks aim to maximize forward velocity, chosen for its benchmark relevance (Tassa et al., 2018) and complex optimization landscape as alluded to by previous results (Haarnoja et al., 2018; Hafner et al., 2019a). Experiments are based on *dflex*, a differentiable rigid-body simulator with soft contact approximation, introduced by (Xu et al., 2022), illustrated in Figure 7 and described in more detail in Appendix E. As is customary in prior work in empirical Deep RL (Tassa et al., 2018), we provide experimental results in an infinite-horizon setting and relax Assumption 2.1.

Metrics. We adopt statistical measures for a robust evaluation across 10 random seeds, utilizing the 50% Interquartile Mean (IQM) and 95% Confidence Interval (CI) as recommended for mitigating statistical uncertainties as suggested by (Agarwal et al., 2021). We also report absolute as well as normalized asymptotic rewards in Appendix H.

Figure 8. Episodic rewards of the Ant task against both simulation steps and wall clock time. The episodic reward is normalized by the highest mean reward achieved by PPO (i.e., PPO-normalized). The dashed lines represent the reward achieved by each respective algorithm at the end of their training runs.

Baselines. This study compares first-order methods against zeroth-order methods. As such, we compare with state-of-the-art model-free methods, PPO (on-policy) (Schulman et al., 2017), and SAC (off-policy) (Haarnoja et al., 2018). For a comprehensive study, we also compare to SVG, a FO-MBRL method that does not utilize a differentiable simulator but instead learns the dynamics model (Amos et al., 2021). Additionally, we also compare our results to SHAC (Xu et al., 2022), one of the best-performing methods based on differentiable simulation. We refer the reader to SHAC (Xu et al., 2022) for additional comparisons with other model-based methods, which SHAC already outperforms. We have tuned all baselines individually to perform well per task and trained them until convergence. Due to long training times, we could not tune SVG since it was not vectorized and instead utilized the hyper-parameters from (Amos et al., 2021). All hyper-parameters are included in Appendix G.

Figure 9. Aggregate asymptotic statistics across all tasks. The left figure shows 50% IQM with 95% CI of asymptotic episode rewards across 10 runs. We observe that AHAC is able to achieve 40% higher reward than our best MFRL baseline, PPO. The right figure shows score distributions as suggested by (Agarwal et al., 2021), which lets us understand the performance variability of each approach. Our proposed approach, AHAC, outperforms baselines even at the worst case, underlining the benefits of first-order methods.

Figure 10. Episodic rewards of the SNU Humanoid task, a muscle-actuated humanoid lower body with $n = 152$. Results are smoothed using EWMA with $= 0.9$. We observe that both SHAC and AHAC scale better to high-dimensional tasks, with the latter achieving 61% more reward than PPO.

Figure 11. Ablations of AHAC on the Ant task. Ablating all additional introduced components reveals that the adaptive horizon objective contributes the most to improving episodic reward, while the double critic helps reduce run-to-run variance.

For comprehension, all rewards presented in this section are re-normalized by the maximum reward achieved by PPO per task. We include the raw numerical results in Appendix H along with further experiment details.

Results. First, we investigate the asymptotic performance of our method on the Ant task, a quadruped with symmetrical legs, $n = 37$ and $m = 8$. The results in Figure 8 show that AHAC achieves a 41% higher reward than the best model-free baseline, PPO, and also outperforms SHAC due to its gradient error avoidance technique. We acknowledge that MFRL methods are computationally simpler and thus also provide results against wall-clock time. Remarkably, PPO and SAC obtain worse episodic rewards over time compared to AHAC, even when they are trained for 3B timesteps, with 100 more samples and 10 more training time. These results suggest that even given practically infinite training data, MFRL methods cannot find truly optimal solutions due to the high variance of zeroth-order gradients.

This trend persists across all evaluated tasks, with AHAC consistently outperforming MFRL baselines. The aggregated statistics in Figure 9 suggest that AHAC obtains a 40% higher reward than our main baseline, PPO. Notably, in high-dimensional tasks such as the SNU Humanoid, AHAC's advantage becomes even more pronounced. Results from Figure 10 suggest that FO-MBRL methods significantly outperform MFRL baselines, with SHAC and AHAC obtaining 44% and 64% more reward than PPO, respectively. However, the larger confidence intervals for SHAC and AHAC hint at ongoing challenges with gradient variance associated with long rollouts. The score distributions in Figure 9 indicate that even with the higher variability, AHAC still outperforms MFRL baselines and exhibits better worst-case performance than SHAC. Additional results and raw metrics are provided in Appendix H.

Ablation study. To elucidate the performance improvements attributed to AHAC, we dissect its pivotal modifications, outlined in Appendix C. Beginning with the SHAC baseline set at $H = 32$, each ablation incrementally introduces a single modification:

1. SHAC H=29: using the H converged by AHAC.
2. Adapt. Obj.: SHAC with Eq. 10 and fixed $H = 32$.
3. Adapt. Horizon: SHAC with Eq. 10 and adapting H.
4. Iterative critic: SHAC with iterative critic training.
5. Double critic: SHAC with a double critic.

Algorithm	Policy Learning	Value Learning	Dynamics Model
PPO (Schulman et al., 2017)	ZOBG	Model-Free	-
SAC (Haarnoja et al., 2018)	0-step FOBG	Model-Free	-
MVE (Feinberg et al., 2018)	0-step FOBG	Model-Based	Deterministic NN
MBPO (Janner et al., 2019)	0-step FOBG	Model-Free	Ensemble NN
PIPPS (Parmas et al., 2018)	ZOBG & FOBG	-	Probabilistic NN
Dreamer (Hafner et al., 2019a)	FOBG	Model-Based	Probabilistic NN
IVG (Byravan et al., 2020)	FOBG	Model-Free	Deterministic NN
SVG (Amos et al., 2021)	FOBG	Model-Free	Deterministic NN
SHAC (Xu et al., 2022)	FOBG	Model-Free	Differentiable sim.
AHAC (ours)	FOBG	Model-Free	Differentiable sim.

Table 1. Comparison between RL algorithms for continuous control. We classify methods by the policy(actor) learning approach. ZOBG stands for methods using Zeroth-Order Batch Gradients following Eq. 2, while FOBG stands for First-Order Batch Gradient methods that differentiate through trajectories following Eq. 3. Model-Based Value Learning refers to methods leveraging Model-Based Value Expansion (MVE) (Feinberg et al., 2018), whereas Model-Free critic learning refers to methods using variants of TD(1) (Sutton & Barto, 2018).

All experiments used the SHAC hyper-parameters, with the exception of the horizon learning rate, specifically adjusted for AHAC. Notably, SHAC with an adaptive horizon (3) is equivalent to AHAC without iterative critic training and single critic implementation. Results, depicted in Figure 11, reveal that incorporating an adaptive horizon significantly enhances the asymptotic reward. Intriguingly, adjusting to $H = 29$ improves rewards over the baseline, yet does not reach the efficacy of the full adaptive horizon approach. This suggests that a static optimal horizon, even if advantageous at policy convergence, may not be optimal during training, leading to local minima. Moreover, the double critic model notably reduces run-to-run variance, surpassing the performance stability of SHAC's single target-critic approach. Additional insights and detailed ablation results are available in Appendix I.

6. Related work

This section reviews recent advancements in continuous control RL, adhering to the actor-critic framework (Konda & Tsitsiklis, 1999), where the critic appraises state-action pairs and the actor identifies optimal actions. We categorize the methods based on their policy training, value estimation, and use of a dynamics model.

In the absence of a known dynamics model, Model-Free Reinforcement Learning (MFRL) methods prevail, which enable learning of action distributions based on state information. Proximal Policy Optimization (PPO) (Schulman et al., 2017) is an on-policy method that utilizes ZOBGs (Eq. 2) and performs gradient updates using recent on-policy samples. Soft Actor-Critic (SAC) (Haarnoja et al., 2018) exemplifies off-policy methods which use a replay buffer to learn from any data and update the actor using 0-step FOBGs defined as $\nabla_\theta J(\theta) := \mathbb{E}_{a_h \sim \pi_\theta(\cdot|s_h)}[\nabla_\theta Q(s,a)]$.

Alternatively, Model-Based Reinforcement Learning (MBRL) methods incorporate a dynamics model to inform learning, either derived from data or assumed a priori. This model can be used to aid the critic's return estimates, which can still be trained model-free (Janner et al., 2019) or through back-propagatable simulated returns via Model-Based Value Expansion (MVE) (Feinberg et al., 2018). Actor training varies; it can be done using 0-step FOBGs augmented by model-generated data (Janner et al., 2019; Feinberg et al., 2018). Alternatively, other work back-propagates through the dynamics model (Hafner et al., 2019a; Byravan et al., 2020) using FOBGs (Eq. 3). (Parmas et al., 2018) also combine ZOBGs and FOBGs, attempting to harness the best of both. Key recent work is summarized in Table 1.

With the emergence of differentiable simulation, many studies (Hu et al., 2019b; Liang et al., 2019; Huang et al., 2021; Du et al., 2021) have explored FOBG optimization by back-propagating through a dynamics model which we refer to as Back-Propagation-Through-Time (BPTT). However, BPTT faces challenges in long episodic RL tasks due to unstable gradients. (Xu et al., 2022) introduces Short Horizon Actor-Critic (SHAC), improving stability through a model-free critic and short rollouts, achieving performance comparable to MFRL with enhanced sample efficiency.

7. Conclusion

Model-free RL (MFRL) approaches are valued for their simplicity, minimal assumptions, and impressive performance. Yet, our study reveals their limitations in complex continuous control tasks, where they achieve good but suboptimal solutions due to high gradient variance. Conversely, First-Order Model-Based RL (FO-MBRL) methods, which leverage efficient gradient propagation through dynamics, have historically lagged behind MFRL in performance.

In this work, we analyze this issue in differentiable simulation through the scope of bias and variance. We derive Lemma 3.2 bounding the observed sample error of first-order gradients relative to zeroth-order gradients, coming to the conclusion that the source of the issue is stiff contact and long horizon rollouts. Based on these insights, we propose the Adaptive Horizon Actor-Critic (AHAC), a new FO-MBRL approach that adapts its rollout horizon during training. Our experiments show that AHAC outperforms MFRL baselines by 40% in complex locomotion tasks, even when the latter are provided with 10^6 times more data. Furthermore, out method maintains competitive time efficiency and shows better scalability to higher-dimensional tasks.

While AHAC outperforms MFRL methods and makes the case for first-order policy learning, it also necessitates the development of differentiable simulators. As such, we admire the simple yet capable MFRL approaches. Our work suggests that future research should not only focus on refining algorithmic approaches for policy learning but also on enhancing simulator technologies to more effectively manage gradient error. Moreover, the practical application of policies trained in differentiable simulators to real-world robotics remains a challenge.

Impact Statement

This paper presents work whose goal is to advance the field of Reinforcement Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

Acknowledgment

The authors (IG, AG) were support by Stephen Fleming Early Career Chair as well as gifts from Nuro and Ford Motor Company, NSERC Discovery Award. This research was supported in part through services from the Partnership for an Advanced Computing Environment (PACE) at the Georgia Institute of Technology, Atlanta, Georgia, USA.

References

Agarwal, R., Schwarzer, M., Castro, P. S., Courville, A. C., and Bellemare, M. Deep reinforcement learning at the edge of the statistical precipice. *Advances in neural information processing systems*, 34:29304–29320, 2021.

Akkaya, I., Andrychowicz, M., Chociej, M., Litwin, M., McGrew, B., Petron, A., Paino, A., Plappert, M., Powell, G., Ribas, R., et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

Amos, B., Stanton, S., Yarats, D., and Wilson, A. G. On the model-based stochastic value gradient for continuous reinforcement learning. In *Learning for Dynamics and Control*, pp. 6–20. PMLR, 2021.

Berahas, A. S., Cao, L., Choromanski, K., and Scheinberg, K. A theoretical and empirical comparison of gradient approximations in derivative-free optimization. *Foundations of Computational Mathematics*, 22(2):507–560, 2022.

Byravan, A., Springenberg, J. T., Abdolmaleki, A., Hafner, R., Neunert, M., Lampe, T., Siegel, N., Heess, N., and Riedmiller, M. Imagined value gradients: Model-based policy optimization with tranferable latent dynamics models. In *Conference on Robot Learning*, pp. 566–589. PMLR, 2020.

Deng, F., Park, J., and Ahn, S. Facing off world model backbones: Rnns, transformers, and s4. *Advances in Neural Information Processing Systems*, 36, 2024.

Du, T., Wu, K., Ma, P., Wah, S., Spielberg, A., Rus, D., and Matusik, W. Diffpd: Differentiable projective dynamics. *ACM Transactions on Graphics (TOG)*, 41(2):1–21, 2021.

Duchi, J. C., Bartlett, P. L., and Wainwright, M. J. Randomized smoothing for stochastic optimization. *SIAM Journal on Optimization*, 22(2):674–701, 2012.

Feinberg, V., Wan, A., Stoica, I., Jordan, M. I., Gonzalez, J. E., and Levine, S. Model-based value expansion for efficient model-free reinforcement learning. *Proceedings of the 35th International Conference on Machine Learning (ICML 2018)*, 2018.

Freeman, C. D., Frey, E., Raichuk, A., Girgin, S., Mordatch, I., and Bachem, O. Brax–a differentiable physics engine for large scale rigid body simulation. *arXiv preprint arXiv:2106.13281*, 2021.

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019a.

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., and Davidson, J. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pp. 2555–2565. PMLR, 2019b.

Hafner, D., Pasukonis, J., Ba, J., and Lillicrap, T. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Hansen, N., Su, H., and Wang, X. Td-mpc2: Scalable, robust world models for continuous control. arXiv preprint arXiv:2310.16828, 2023.

Heiden, E., Macklin, M., Narang, Y. S., Fox, D., Garg, A., and Ramos, F. DiSECt: A Differentiable Simulation Engine for Autonomous Robotic Cutting. Robotics: Science and Systems, 2021.

Hu, Y., Anderson, L., Li, T.-M., Sun, Q., Carr, N., Ragan-Kelley, J., and Durand, F. Difftaichi: Differentiable programming for physical simulation. arXiv preprint arXiv:1910.00935, 2019a.

Hu, Y., Liu, J., Spielberg, A., Tenenbaum, J. B., Freeman, W. T., Wu, J., Rus, D., and Matusik, W. Chainqueen: A real-time differentiable physical simulator for soft robotics. In 2019 International conference on robotics and automation (ICRA), pp. 6265–6271. IEEE, 2019b.

Huang, Z., Hu, Y., Du, T., Zhou, S., Su, H., Tenenbaum, J. B., and Gan, C. Plasticinelab: A soft-body manipulation benchmark with differentiable physics. arXiv preprint arXiv:2104.03311, 2021.

Hutter, M., Gehring, C., Jud, D., Lauber, A., Bellicoso, C. D., Tsounis, V., Hwangbo, J., Bodie, K., Fankhauser, P., Bloesch, M., et al. Anymal-a highly mobile and dynamic quadrupedal robot. In 2016 IEEE/RSJ international conference on intelligent robots and systems (IROS) pp. 38–44. IEEE, 2016.

Hwangbo, J., Sa, I., Siegwart, R., and Hutter, M. Control of a quadrotor with reinforcement learning. IEEE Robotics and Automation Letters, 2(4):2096–2103, 2017.

Hwangbo, J., Lee, J., Dosovitskiy, A., Bellicoso, D., Tsounis, V., Koltun, V., and Hutter, M. Learning agile and dynamic motor skills for legged robots. Science Robotics, 4(26):eaau5872, 2019.

Janner, M., Fu, J., Zhang, M., and Levine, S. When to trust your model: Model-based policy optimization. Advances in Neural Information Processing Systems, 2019.

Kabzan, J., Hewing, L., Liniger, A., and Zeilinger, M. N. Learning-based model predictive control for autonomous racing. IEEE Robotics and Automation Letters, 4(4): 3363–3370, 2019.

Kaufmann, E., Loquercio, A., Ranftl, R., Müller, M., Koltun, V., and Scaramuzza, D. Deep drone acrobatics. arXiv preprint arXiv:2006.05768, 2020.

Konda, V. and Tsitsiklis, J. Actor-critic algorithms. Advances in neural information processing systems, 12, 1999.

Lee, M., Lee, J., and Lee, D. Differentiable dynamics simulation using invariant contact mapping and damped contact force. In 2023 IEEE International Conference on Robotics and Automation (ICRA), pp. 11683–11689. IEEE, 2023.

Liang, J., Lin, M., and Koltun, V. Differentiable cloth simulation for inverse problems. Advances in Neural Information Processing Systems, 32, 2019.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971, 2015.

Macklin, M. Warp: A high-performance python framework for gpu simulation and graphics. https://github.com/nvidia/warp, March 2022. NVIDIA GPU Technology Conference (GTC).

Mohamed, S., Rosca, M., Figurnov, M., and Mnih, A. Monte carlo gradient estimation in machine learning. The Journal of Machine Learning Research, 21(1):5183–5244, 2020.

Parmas, P., Rasmussen, C. E., Peters, J., and Doya, K. Pipps: Flexible model-based policy search robust to the curse of chaos. In International Conference on Machine Learning, pp. 4065–4074. PMLR, 2018.

Parmas, P., Seno, T., and Aoki, Y. Model-based reinforcement learning with scalable composite policy gradient estimators. In International Conference on Machine Learning, pp. 27346–27377. PMLR, 2023.

Rudin, N., Hoeller, D., Reist, P., and Hutter, M. Learning to walk in minutes using massively parallel deep reinforcement learning. In Conference on Robot Learning, pp. 91–100. PMLR, 2022.

Schulman, J., Heess, N., Weber, T., and Abbeel, P. Gradient estimation using stochastic computation graphs. Advances in neural information processing systems, 28, 2015.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347, 2017.

Suh, H. J., Simchowitz, M., Zhang, K., and Tedrake, R. Do differentiable simulators give better policy gradients? In International Conference on Machine Learning, pp. 20668–20696. PMLR, 2022.

Sutton, R. S. and Barto, A. G. Reinforcement learning: An introduction. MIT press, 2018.

Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

Tassa, Y., Doron, Y., Muldal, A., Erez, T., Li, Y., Casas, D. d. L., Budden, D., Abdolmaleki, A., Merel, J., Lefrancq, A., et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Reinforcement learning*, pp. 5–32, 1992.

Xu, J., Chen, T., Zlokapa, L., Foshey, M., Matusik, W., Sueda, S., and Agrawal, P. An end-to-end differentiable framework for contact-aware robot design. *arXiv preprint arXiv:2107.07501*, 2021.

Xu, J., Makoviychuk, V., Narang, Y., Ramos, F., Matusik, W., Garg, A., and Macklin, M. Accelerated policy learning with parallel differentiable simulation. *arXiv preprint arXiv:2204.07137*, 2022.

Zhong, Y. D., Han, J., Dey, B., and Brikis, G. O. Improving gradient computation for differentiable physics simulation with contacts. In *Learning for Dynamics and Control Conference*, pp. 128–141. PMLR, 2023.

(a) The Soft Heaviside function of Eq. 4. (b) Gradient estimates at $N = 1000$. (c) Gradient sample error for different sample sizes N. Estimated by comparing the difference between the gradient estimate and the true gradient below.

Figure 12. Gradient sample error study for the Soft Heaviside function shown in Eq. 4. Both ZOBG and FOBG exhibit sample errors at low sample sizes; however, FOBGs are especially susceptible to the "empirical bias" phenomena.

A. Heaviside example

This appendix provides additional details on the Heaviside example used to obtain intuition regarding FOBG sample error in Section 3.

$$H(x) = \begin{cases} 1 & x > = 2 \\ 2x= & jxj = 2 \\ 1 & x < = 2 \end{cases}$$

Under stochastic input $x = ( ) = + w$ where $w \sim N(0;)$, we can obtain the expected value:

$$\begin{aligned} E_w\, H(x) &= \int_{-\infty}^{\infty} H(x)\ (x)dx \\ &= \int_{-\infty}^{1} (x)dx + \int_{= 2}^{= 2} \frac{2x}{} (x)dx + \int_{= 2}^{1} (x)dx \\ &= \frac{1}{2}\operatorname{erfc}\left(\frac{+2}{2\sqrt{2}}\right) + \frac{1}{2}\operatorname{erfc}\left(\frac{2}{2\sqrt{2}}\right) + -\operatorname{erf}\left(\frac{2}{2\sqrt{2}}\right) + -\operatorname{erf}\left(\frac{+2}{2\sqrt{2}}\right) \\ &\quad + \frac{\sqrt{2}}{\sqrt{=}}\left[\exp\left(\frac{(+2)^2}{8^2}\right) - \exp\left(\frac{(2)^2}{8^2}\right)\right] \end{aligned}$$

From the expectation, we can obtain the gradient w.r.t. the parameter of interest:

$$\begin{aligned} r\ E_w\, H(x) &= \sqrt{\frac{1}{2}}\exp\left(\frac{(+2)^2}{8^2}\right) + \sqrt{\frac{1}{2}}\exp\left(\frac{(2)^2}{8^2}\right) + \frac{1}{}\operatorname{erf}\left(\frac{2}{2\sqrt{2}}\right) + \frac{1}{}\operatorname{erf}\left(\frac{+2}{2\sqrt{2}}\right) \\ &\quad \frac{\sqrt{2}}{\sqrt{=}}\exp\left(\frac{(2)^2}{8^2}\right) + \frac{\sqrt{2}}{\sqrt{=}}\exp\left(\frac{(+2)^2}{8^2}\right) \\ &\quad \sqrt{\frac{1}{2}}\exp\left(\ (2)^{1=4^2}\ (2)^{1=4^2\ 1}\right) \sqrt{\frac{1}{2}}\exp\left(\ (+2)^{1=4^2}\ (+2)^{1=4^2\ 1}\right) \end{aligned}$$

As seen from the equation above, the true gradient $E_w\, H(x) \neq 0$ at $= 0$. However, using FOBG, we obtain $r\ H(a) = 0$ in samples where $jaj > = 2$, which occurs with probability at least $= \sqrt{2}$. Even though both ZOBG and FOBG are theoretically unbiased as $N \to 1$, both exhibit "empirical bias", as shown in Figure 12.

B. Proof of Lemma 3.2

First we reiterate the assumptions to make this section self-suf cient and easier to read.

Assumption B.1. Policy $\pi(\cdot|s) : \mathbb{R}^n \to \mathbb{R}^d \to [0,1]^m$ is continuously differentiable and Lipshitz smooth with $\|\nabla_s \pi(a|s)\| \le B_\pi$.

Assumption B.2. Dynamics function $f(s,a) : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}^n$ is continuously differentiable and Lipshitz smooth in both arguments with $\|\nabla_s f(s,a)\| \le B_f$ and $\|\nabla_a f(s,a)\| \le B_f$.

Assumption B.3. Reward function $r(\cdot,\cdot) : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}$ is continuously differentiable and Lipshitz smooth with $\|\nabla_s r(s,a)\| \le B_r$.

Assumption B.4. ZOBG use baseline subtraction which does not introduce gradient sample error $\hat{\nabla}[0](\theta) = \sum_{h=1}^{H} r(s_h,a_h) \cdot b \cdot \nabla_\theta \log \pi_\theta(a_h|s_h)$

Proof. First, we expand our de nition of sample error and de ne a random variable of a single Monte-Carlo sample

$$
\begin{aligned}
B &= \mathbb{E}\left[\hat{\nabla}^{[1]} J(\theta)\right] - \mathbb{E}\left[\hat{\nabla}^{[0]} J(\theta)\right] = \frac{1}{N}\sum_{i=1}^{N} \hat{\nabla}^{[1]} J_i(\theta) - \frac{1}{N}\sum_{i=1}^{N} \hat{\nabla}^{[0]} J_i(\theta) \\
&= \frac{1}{N}\sum_{i=1}^{N} (\hat{\nabla}^{[1]} J_i(\theta) - \hat{\nabla}^{[0]} J_i(\theta)) \\
&\quad \|\hat{\nabla}^{[1]} J(\theta) - \hat{\nabla}^{[0]} J(\theta)\| \qquad (11)
\end{aligned}
$$

We drop the sample subscript i for simplicity and assume that $\pi(\cdot|s) = \pi(\cdot|\mathrm{sg}(s))$ where $\mathrm{sg}(\cdot)$ is the stop-gradient operator makes the expansion of $\hat{\nabla}^{[1]} J(\theta)$ easier (Deng et al., 2024).

$$
\begin{aligned}
&\hat{\nabla}^{[1]} J(\theta) - \hat{\nabla}^{[0]} J(\theta) \\
&= \nabla_\theta \Big[\sum_{h=1}^{H} r(s_h,a_h)\Big] \sum_{h=1}^{H} r(s_h,a_h) \cdot b \cdot \nabla_\theta \log \pi_\theta(a_h|s_h) \\
&= \sum_{h=1}^{H} \nabla_{a_h} r(s_h,a_h) \nabla_\theta \pi_\theta(a_h|s_h) + \sum_{h'=1}^{h-1} \prod_{t=h'+1}^{h-1} \nabla_{s_t} f(s_t,a_t) \cdot \nabla_{a_{h'}} f(s_{h'},a_{h'})^T \nabla_\theta \pi_\theta(a_{h'}|s_{h'}) \nabla_{s_h} r(s_h,a_h) \\
&\quad \sum_{h=1}^{H} r(s_h,a_h) \cdot b \cdot \nabla_\theta \log \pi_\theta(a_h|s_h) \\
&= \sum_{h=1}^{H} \nabla_\theta \pi_\theta(a_h|s_h)^T \nabla_{a_h} r(s_h,a_h) - r(s_h,a_h) \cdot b \cdot \pi_\theta(a_h|s_h)^{-1} + \\
&\quad \sum_{h'=1}^{h-1} \prod_{t=h'+1}^{h-1} \nabla_{s_t} f(s_t,a_t) \cdot \nabla_{a_{h'}} f(s_{h'},a_{h'}) \nabla_\theta \pi_\theta(a_{h'}|s_{h'})^T \nabla_{s_h} r(s_h,a_h) \qquad (12)
\end{aligned}
$$

where x^{-1} is the Hadamard inverse. Setting $b = r(s_h,0)$, we can exploit the Lipshitz smoothness of r. In general, for any function $f(x)$:

$$
\begin{aligned}
f(y) &\le f(x) + \nabla f(x)^T (y - x) + \frac{L}{2}\|y - x\|^2 \\
f(0) &\le f(x) - \nabla f(x)^T x + \frac{L}{2}\|x\|^2 \\
f(x) &\le f(0) - \nabla f(x)^T x + \frac{L}{2}\|x\|^2 \\
f(x) &\le f(0) + (-\nabla f(x)^T + \frac{L}{2} x^T) x \qquad (13)
\end{aligned}
$$

Applying $b = r(s_h; 0)$ to Eq. 12 yields

$$
\begin{aligned}
&= \sum_{h=1}^{H} r_{} (a_h | s_h)^T \; r_{a_h} r(s_h; a_h) \quad r(s_h; a_h) \quad r(s_h; 0) \quad (a_h | s_h)^{\;1} \; + \\
&\quad \sum_{h'=1}^{h-1} \prod_{t=h'+1}^{h-1} r_{s_t} f(s_t; a_t) \; r_{a_{h'}} f(s_{h'}; a_{h'}) r_{} \quad (a_{h'} | s_{h'}) \quad r_{s_h} r(s_h; a_h) \\
&= \sum_{h=1}^{H} r_{} (a_h | s_h)^T \; r_{a_h} r(s_h; a_h) \quad r_{a_h} r(s_h; a_h) + \frac{B_r}{2} \quad (a_h | s_h) \\
&\quad + \sum_{h'=1}^{h-1} \prod_{t=h'+1}^{h-1} r_{s_t} f(s_t; a_t) \; r_{a_{h'}} f(s_{h'}; a_{h'}) r_{} \quad (a_{h'} | s_{h'}) \quad r_{s_h} r(s_h; a_h) \\
&= \frac{B_r}{2} \sum_{h=1}^{H} r_{} \quad (a_h | s_h)^T \quad (a_h | s_h) + \sum_{h'=1}^{h-1} \prod_{t=h'+1}^{h-1} r_{s_t} f(s_t; a_t) \; r_{a_{h'}} f(s_{h'}; a_{h'}) r_{} \quad (a_{h'} | s_{h'}) \quad r_{s_h} r(s_h; a_h)
\end{aligned}
\tag{14}
$$

Plug Eq. 14 into Eq. 11:

$$
\begin{aligned}
B \quad & \hat{}^{[1]} J(\;) \quad \hat{}^{[0]} J(\;) \\
&= \frac{B_r}{2} \sum_{h=1}^{H} r_{} \quad (a_h | s_h)^T \quad (a_h | s_h) + \sum_{h'=1}^{h-1} \prod_{t=h'+1}^{h-1} r_{s_t} f(s_t; a_t) r_{a_{h'}} f(s_{h'}; a_{h'}) r_{} \quad (a_{h'} | s_{h'}) \quad r_{s_h} r(s_h; a_h) \\
&\qquad \text{Apply Eq. 13} \\
&\quad \frac{B_r}{2} \sum_{h=1}^{H} r_{} \quad (a_h | s_h)^T \quad (a_h | s_h) \\
&\qquad + \sum_{h'=1}^{h-1} \prod_{t=h'+1}^{h-1} r_{s_t} f(s_t; a_t) \; r_{a_{h'}} f(s_{h'}; a_{h'}) r_{} \quad (a_{h'} | s_{h'}) \quad r_{s_h} r(s_h; a_h) \\
&\qquad \text{since} k r_{} \quad (a_h | s_h) k \quad B \quad \text{and} k \quad (a_h | s_h) k \quad 1 \\
&\quad \frac{1}{2} H B_r B \quad + \sum_{h'=1}^{h-1} \prod_{t=h'+1}^{h-1} r_{s_t} f(s_t; a_t) \; r_{a_{h'}} f(s_{h'}; a_{h'}) r_{} \quad (a_{h'} | s_{h'}) \quad r_{s_h} r(s_h; a_h) \\
&\quad \frac{1}{2} H B_r B \quad + (H \quad 1) B_r B \; B_f^{H \quad 1} \\
&\quad \frac{1}{2} H B_r B \quad + H B_r B \; B_f^{H \quad 1} \\
&= H B_r B \; (\frac{1}{2} + B_f^{H \quad 1})
\end{aligned}
$$

□

C. Summary of modi cations

To develop the Adaptive Horizon Actor-Critic (AHAC) algorithm, we used the Short Horizon Actor-Critic (SHAC) algorithm (Xu et al., 2022) as a starting point. This section details all modi cations applied to the SHAC in order to derive AHAC and achieve the reported results in this paper. We also note that some of these are not exclusive to either approach approach.

1. Adaptive horizon objective - instead of optimizing the short horizon rollout return, we introduce the new constrained objective shown in Equation 9. To optimise that and adapt the horizon H, we introduced the dual problem in Equation 10 and optimised it directly for policy parameters θ and the Lagrangian coef cients.

$$J(\theta) := \sum_{h=t}^{t+T-1} \gamma^{h-t} r(s_h, a_h) + \gamma^{T} V(s_{t+T}) \quad \text{s.t.:} \quad \|s_t - f(s_t, a_t)\| \leq \epsilon \quad \forall t \in \{0, \dots, T\}$$

$\underbrace{\qquad\qquad}_{\text{SHAC objective}}$

$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}_{\text{AHAC objective}}$

2. Double critic - the original implementation of SHAC struggled with more complex tasks such as Humanoid due to its highly non-convex value landscape. The authors of (Xu et al., 2022) solved that by introducing a delayed target critic similar to prior work in deep RL (Lillicrap et al., 2015). We found that approach brittle and required more hyper-parameter tuning. Instead, we replaced it with a double critic similar to SAC (Haarnoja et al., 2018). For our work, we found that it reduced the variance of asymptotic rewards achieved by AHAC while removing a hyperparameter. While this technique is usually applied to off-policy algorithms, we nd it helpful in highly parallelized simulations due to the high data throughput.

3. Critic training until convergence - empirically we found that different problems present different value landscapes. The more complex the landscape, the more training the critic required and the critic often failed to t the data with the limited number of critic iterations done in SHAC (16). Instead of training the critic for a xed number of iterations, we trained the (dual) critic of AHAC until convergence de ned by $\frac{1}{5}\sum_{i=n-5}^{n} L_i(\theta) - L_{i-1}(\theta) < 0.5$ where $L_i(\theta)$ is the critic loss for mini-batch iteration i. We allowed the critic to be trained for a maximum of 64 iterations. We found that this resulted in asymptotic performance improvements on more complex tasks such as Humanoid and SNU Humanoid, while removing yet another hyper-parameter.

D. AHAC-1 algorithm

Algorithm 2 shows the single-environment version of AHAC that was described in Section 4.1. While this algorithm applies the contact truncation technique perfectly and avoids all stiff contact, it is also not vectorizable. When attempting to vectorize AHAC-1, it necessitates cutting off compute graphs per-environment based on the individual environment dynamics. This is impossible to accomplish with typical deep learning frameworks such as PyTorch. Another alternative would be to execute different environments in different threads, but unfortunately, that does not bene t from GPU acceleration.

E. Differentiable Simulation Setup: d ex

The experimental simulator, d ex (Xu et al., 2022), employed in Section 5, is a GPU-based differentiable simulator utilizing the Featherstone formulation for forward dynamics and a spring-damper contact model with Coulomb friction.

The dynamics function f is modeled by solving the forward dynamics equations:

$$M(q)\ddot{q} = J^T F(q, \dot{q}) + c(q, \dot{q}) + \tau(q, \dot{q}, a)$$

where $q, \dot{q}, \ddot{q}$ are joint coordinates, velocities, and accelerations, respectively. F represents external forces, c includes Coriolis forces, and τ denotes joint-space actuation. Mass matrix M and Jacobian J are computed concurrently using one thread per-environment. The composite rigid body algorithm (CRBA) is employed for articulation dynamics, enabling caching of the matrix factorization for reuse in the backward pass through parallel Cholesky decomposition.

After determining joint accelerations $\ddot{q}$, a semi-implicit Euler integration step updates the system state $(q, \dot{q})$. Torque-based control is employed for simple environments, where the policy output a sets τ each timestep. For further details, see (Xu et al., 2022). It is noted that d ex is no longer actively developed and has been succeeded by warp (Macklin, 2022).

Algorithm 2 Adaptive Horizon Actor-Critic 1 (Single environment)

Given: : discount rate
Given: ; : learning rates
Given: H: maximum trajectory length
Given: C: contact threshold
Initialize learnable parameters ;
t 0
while episode not done **do**
. Rollout policy
Initialize rollout buffer D
while $k\,r\,f\,k$ C and h H **do**
a_t (js_t)
$r_t = r(s_t; a_t)$
$s_{t+1} = f(s_t; a_t)$
D D [$f(s_{t+h}; a_{t+h}; r_{t+h}; V$ $(s_{t+h+1}))g$
t $t + 1$
end while
. Train actor using Eq. 8
r $J(\ )$
.Train critic using Eq. 7
while not converged **do**
sample $(s; \hat{V}(s))$ D
+ r $L(\ )$
end while
end while

Figure 13. Locomotion environments (left to right): Hopper, Ant, Anymal, Humanoid and SNU Humanoid.

F. Environment details

In this paper, we explore 5 locomotion tasks with increasing complexity. They are described below and shown in Figure 13.

1. **Hopper**, a single-legged robot jumping only in one axis with $n = 11$ and $m = 3$.
2. **Ant**, a four-legged robot with $n = 37$ and $m = 8$.
3. **Anymal**, a more sophisticated quadruped with $n = 49$ and $m = 12$ modeled after (Hutter et al., 2016).
4. **Humanoid**, a classic contact-rich environment with $n = 76$ and $m = 21$, which requires extensive exploration to find a good policy.
5. **SNU Humanoid**, a version of Humanoid lower body where instead of joint torque control, the robot is controlled via $m = 152$ muscles intended to challenge the scaling capabilities of algorithms.

All tasks share the same common main objective - maximize forward velocity v_x:

Environment	Reward
Hopper	$v_x + R_{height} + R_{angle}$ $0{:}1\|a\|_2^2$
Ant	$v_x + R_{height} + 0{:}1R_{angle} + R_{heading}$ $0{:}01\|a\|_2^2$
Anymal	$v_x + R_{height} + 0{:}1R_{angle} + R_{heading}$ $0{:}01\|a\|_2^2$
Humanoid	$v_x + R_{height} + 0{:}1R_{angle} + R_{heading}$ $0{:}002\|a\|_2^2$
Humanoid STU	$v_x + R_{height} + 0{:}1R_{angle} + R_{heading}$ $0{:}002\|a\|_2^2$

Table 2. Rewards used for each task benchmarked in Section 5

We additionally use auxiliary rewards R_{height} to incentivize the agent to, R_{angle} to keep the agent's normal vector point up, $R_{heading}$ to keep the agent's heading pointing towards the direction of running and a norm over the actions to incentivize energy-efficient policies. For most algorithms, none of these rewards, apart from the last one, are crucial to succeeding in the task. However, all of them aid learning policies faster.

$$R_{height} = \begin{cases} h \quad h_{term} & if\, h \quad h_{term} \\ 200(h \quad h_{term})^2 & if\, h < h_{term} \end{cases}$$

$$R_{angle} = 1 \quad \left(\frac{}{term}\right)^2$$

$R_{angle} = \|q_{forward} \quad q_{agent}\|_2^2$ is the difference between the heading of the agent q_{agent} and the forward vector q_{agent}. h is the height of the CoM of the agent and is the angle of its normal vector. h_{term} and $_{term}$ are parameters that we set for each environment depending on the robot morphology. Similar to other high-performance RL applications in simulation, we find it crucial to terminate episode early if the agent exceeds these termination parameters. However, it is worth noting that AHAC is still capable of solving all tasks described in the paper without these termination conditions, albeit slower.

G. Hyper-parameters

This section details all hyper-parameters used in the main experiments in Section 5. PPO and SAC, as our MFRL baselines, have been tuned to perform well across all tasks, including task-specific hyper-parameters. SVG has not been specifically tuned for all benchmarks due to time limitations but instead uses the hyper-parameters presented in (Amos et al., 2021).[1] SHAC is tuned to perform well across all tasks using a fixed $H = 32$ as in the original work (Xu et al., 2022). AHAC shares all of its common hyper-parameters with SHAC and only has its horizon learning rate tuned per-task. The contact threshold C and iterative critic training criteria did not benefit from tuning. Note that the double critic employed by AHAC uses the same hyper-parameters used by the SHAC critic. Therefore, we have left AHAC under-tuned in comparison to SHAC in order to highlight the benefits of the adaptive horizon mechanism presented in this work.

Table 3 shows common hyper-parameters shared between all tasks. While table 4 shows hyper-parameters specific to each problem,. Where possible, we attempted to use the hyper-parameters suggested by the original works; however, we also attempted to share hyper-parameters between algorithms to ease comparison. If a specific hyperparameter is not mentioned, then it is the one used in the original work behind the specific algorithm.

[1]Tuning SVG proved difficult as we were unable to vectorize the algorithm, resulting in up to 2-week training times. This made it difficult to tune for our benchmarks

	AHAC	SHAC	PPO	SAC	SVG
Mini-epochs		16	5		4
Batch size	8	8	8	32	1024
	0.95	0.95	0.95		
	0.99	0.99	0.99	0.99	0.99
H - horizon		32	32		3
C - contact thresh.	500				
Grad norm	1.0	1.0	1.0		
			0.2		
Actor $log(\ )$ bounds				(-5,2)	(-5,2)
- temperature				0.2	0.1
				10^{-4}	10^{-4}
jDj - buffer size				10^6	10^6
Seed steps	0	0	0	10^4	10^4

Table 3. Table of hyper-parameters for all algorithms benchmarked in Section 5. These are shared across all tasks.

	Hopper	Ant	Anymal	Humanoid	SNU Humanoid
Actor layers	(128, 64, 32)	(128, 64, 32)	(256, 128)	(256, 128)	(512, 256)
Actor	$2 \quad 10^{-3}$	$2 \quad 10^{-3}$	$2 \quad 10^{-3}$	$2 \quad 10^{-3}$	$2 \quad 10^{-3}$
Horizon	$2 \quad 10^{-4}$	$1 \quad 10^{-5}$	$1 \quad 10^{-5}$	$1 \quad 10^{-5}$	$1 \quad 10^{-5}$
Critic layers	(64, 64)	(64, 64)	(256, 128)	(256, 128)	(256, 256)
Critic	$4 \quad 10^{-3}$	$2 \quad 10^{-3}$	$2 \quad 10^{-3}$	$5 \quad 10^{e-4}$	$5 \quad 10^{-4}$
Critic	0.2	0.2	0.2	0.995	0.995

Table 4. Task-specific hyper-parameters. All benchmarked algorithms share the same actor and critic network hyper-parameters with ELU activation functions. AHAC and PPO do not have target critic networks and, as such, do not have τ as a hyper-parameter.

H. Experimental results

In addition to the experimental results in Section 5, here we provide the same results in more detail. Figure 14 depicts step-wise and time-wise reward curves for all experiments. Tables 5 and 6 provide asymptotic (converged) results for all tasks with PPO-normalized and raw rewards, respectively.

	Hopper		Ant		Anymal		Humanoid		SNU Humanoid	
PPO	1:00	0:11	1:00	0:12	1:00	0:03	1:00	0:05	1:00	0:09
SAC	0:87	0:16	0:95	0:08	0:98	0:06	1:04	0:04	0:88	0:11
SVG	0:84	0:08	0:83	0:13	0:84	0:19	1:06	0:16	0:75	0:23
SHAC	1:02	0:03	1:16	0:13	1:26	0:04	1:15	0:04	1:44	0:08
AHAC	1:10	0:00	1:41	0:08	1:46	0:06	1:35	0:07	1:64	0:07

Table 5. Tabular results of the asymptotic rewards achieved by each algorithm across all tasks. The results presented are PPO-normalized 50 % IQM and standard deviation across 10 random seeds. All algorithms have been trained until convergence.

	Hopper		Ant		Anymal		Humanoid		SNU Humanoid	
PPO	4742	521	6605	793	12029	360	7293	365	4114	370
SAC	4126	759	6275	528	11788	722	7285	292	3620	453
SVG	3983	379	5482	859	10104	2286	7731	1167	3086	946
SHAC	4837	142	7662	859	15157	481	8387	292	5924	329
AHAC	5216	21	9313	528	17562	722	9846	511	6746	288

Table 6. Tabular results of the asymptotic (end of training) rewards achieved by each algorithm across all tasks. The results presented are 50 % IQM and standard deviation across 10 random seeds. All algorithms have been trained until convergence.

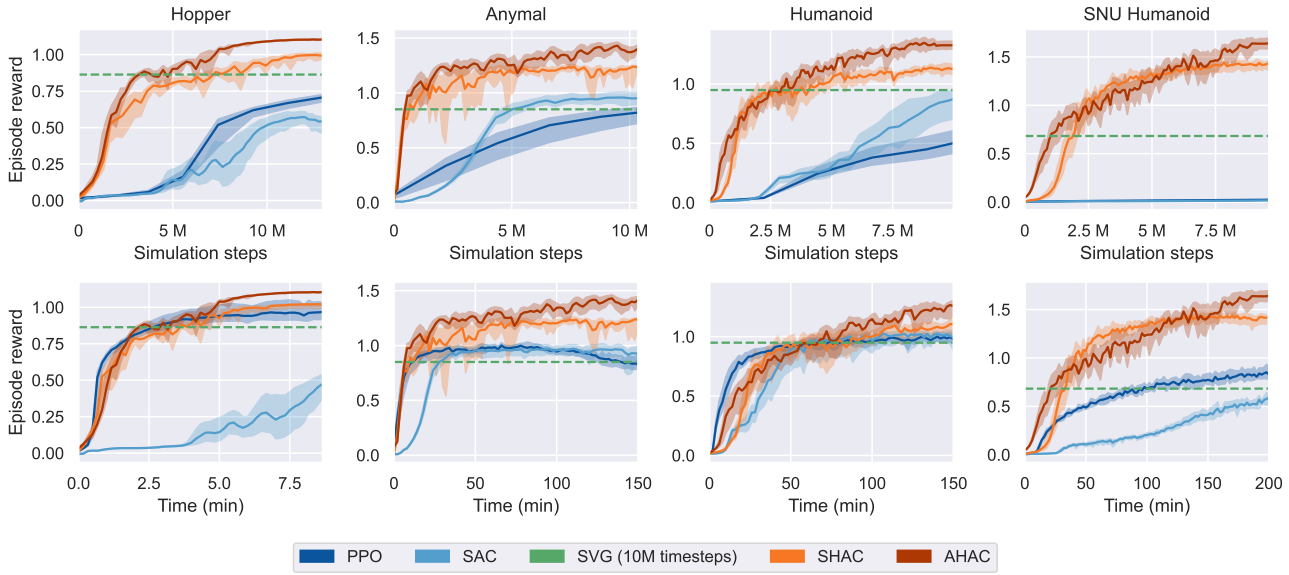

Figure 14. **Reward curves for all tasks against both simulation steps and training time**. We plot 50 % IQM and 95 % CI.

I. Ablation study details

In Section 5, we provided an ablation study of the individual contributions of our proposed approach, AHAC, as summarized in Appendix C. In this section, we provide further details on the conducted experiments. The aim of the study is to understand what changes contribute to the asymptotic performance of AHAC. To best achieve that, we started with SHAC as the baseline, using the tuned version detailed in Appendix G above. Afterwards, we add the individual components that contribute to AHAC using the hyper-parameter from the section above. Note that only hyper-parameters particular to AHAC have been tuned to achieve the results presented in this paper; all other hyper-parameters are the ones tuned to our baseline SHAC with $H = 32$. In particular, we have only tuned the adaptive horizon learning rate and contact threshold C. Table 7 shows the detailed differences between the ablations presented in Section 5. The ablations include:

1. SHAC H=32 - our baseline with most hyper-parameters tuned to it.
2. SHAC H=29 - SHAC using the horizon H which AHAC converges to asymptotically.
3. Adaptive Objective - SHAC using the adaptive horizon objective introduced in Eq. 10 but without using it to adapt to the horizon.
4. Adaptive Horizon - SHAC using the objective in Eq 10 and adapting the horizon. This is equivalent to AHAC without the double critic and with iterative training.
5. Iterative critic - SHAC with a single target critic, utilizing iterative critic training until convergence.
6. Double critic - SHAC with a double critic and no target.

Previously in Section 5, we only provided end of training results for the Ant task. In Table 8 we provide the same results in tabular form. We also provide the learning curves for the same experiments in Figure 15.

	Ablation	H	Actor objective	Critic	Iterative critic training
	SHAC H=32	32	Eq. 8	Single w/ target	
Actor ablations	SHAC H=29	29	Eq. 8	Single w/ target	
	Adapt. Objective	32	Eq. 10	Single w/ target	
	Adapt. Horizon	adaptive	Eq. 10	Single w/ target	
Critic ablations	Iterative critic	32	Eq. 8	Single w/ target	✓
	Double critic	32	Eq. 8	Dual	
	AHAC	adaptive	Eq. 10	Dual	✓

Table 7. Differences between ablations studied, split into actor and critic ablations. All ablations only introduce one component to the baseline, SHAC.

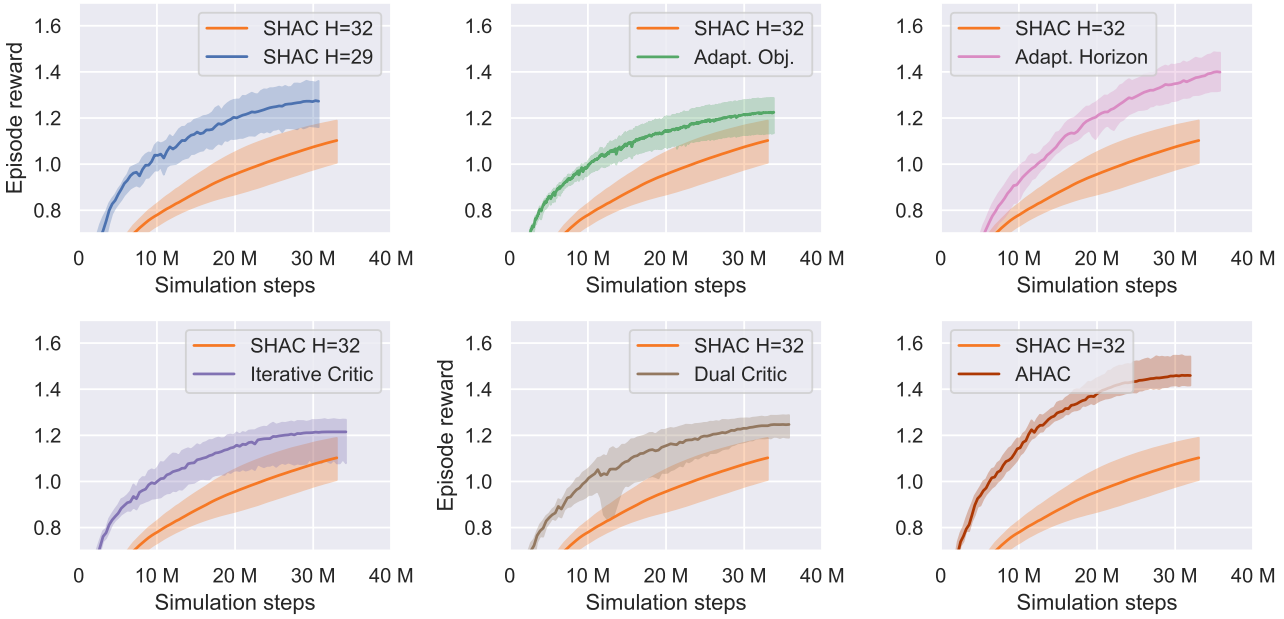

Figure 15. Standalone ablation results for the Ant task. These results are the same as in Figure 11 but presented in a different format for improved legibility.

Ablation	Asymptotic reward	
SHAC H=32	1.16	0.14
1. SHAC H=29	1.23	0.17
2. Adaptive Objective	1.18	0.18
3. Adaptive Horizon	1.35	0.12
4. Iterative Critic	1.17	0.13
5. Double Critic	1.20	0.07
AHAC	1.41	0.08

Table 8. Results of asymptotic performance of our ablation study showing 50% IQM and standard deviation.