2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Maximizing Quadruped Velocity by Minimizing Energy

Srinath Mahankali* [1], Chi-Chang Lee* [2], Gabriel B. Margolis[1], Zhang-Wei Hong[1], Pulkit Agrawal[1]

Fig. 1: **High-speed Running Without Reward Shaping.** We train a policy to run at high speeds with just two reward terms: velocity tracking and energy minimization. Furthermore, our approach automatically and adaptively balances these two terms online during training. This is due to Extrinsic-Intrinsic Policy Optimization (EIPO), a constrained policy optimization algorithm that strictly prioritizes the velocity-tracking task performance while treating energy minimization as an intrinsic reward.

***Abstract*— Reinforcement Learning (RL) has been a powerful tool for training robots to acquire agile locomotion skills. To learn locomotion, it is commonly necessary to introduce additional reward-shaping terms, such as an energy minimization term, to guide an algorithm like Proximal Policy Optimization (PPO) to good performance. Prior works rely on hyper-parameter tuning on the weight of the reward shaping terms to obtain satisfactory task performance. To save the efforts of tuning these weights, we adopt the Extrinsic-Intrinsic Policy Optimization (EIPO) framework. The key idea of EIPO is to establish a constrained optimization framework for the primary objective of enhancing task performance and the secondary objective of minimizing energy consumption. It seeks a policy that minimizes the energy consumption objective within the optimal policy space for task performance. This guarantees that the learned policy excels in task performance while conserving energy, all without requiring manual weight adjustments for both objectives. Our experiments evaluate EIPO on various quadruped locomotion tasks, revealing that policies trained with EIPO consistently achieve higher task performance than PPO comparisons while maintaining comparable energy consumption levels. Furthermore, EIPO exhibits superior task performance in real-world evaluations compared to PPO.**

I. Introduction

Reinforcement learning (RL) [1] has been successfully used to train policies for complex and contact-rich motor skills such as quadruped locomotion [2]–[6], legged manipulation [7], and in-hand object re-orientation [8], [9]. The typical paradigm involves training policies in simulated environments and deploying them in the real world [10], [11]. Training policies often requires multiple auxiliary reward-shaping terms to guide the learning process [12]. Such reward-shaping terms commonly include penalties for joint accelerations, velocities, or torques [13]. One important reward-shaping method is penalizing energy consumption, which we focus on in this paper as an auxiliary objective.

Minimizing energy consumption is a natural choice of auxiliary objective, which has been used in many prior works with various different forms such as penalizing mechanical work [10], [14] or penalizing joint torques [2], [13], [15] due to its sim-to-real benefits and the emergence of natural behaviors when minimizing energy consumption [16], [17]. Moreover, prior work has shown that animal and human locomotion are energy-efficient [18], [19], motivating the use of an energy penalty as a heuristic for robot locomotion.

Generally, the policy is trained to jointly maximize task rewards while minimizing energy consumption. The joint optimization objective is typically expressed as $\max_\pi J(\pi) - \lambda F(\pi)$, where π represents the policy, and $J(\pi)$ and $F(\pi)$ stand for task rewards and energy consumption, respectively [16], [17] However, policy training with pre-defined weightings between the competing objectives of maximizing task performance and minimizing energy entails a delicate balance between competing reward criteria [20]. This balance is commonly achieved either through manual adjustment of the ratio between the two reward terms or by employing hyper-parameter optimization techniques (i.e., grid search). Such techniques require training several policies as a result of searching for the optimal weighting λ, leading to expensive computational costs. Moreover, jointly optimizing the task reward and the energy consumption penalty could lead to sub-optimal task rewards J because the policy may maximize the overall policy's objective by consuming very little energy (i.e., minimizing F) and ignoring J. However, the learned policy π should ideally be optimal with respect to the task rewards J while treating the minimization of energy consumption as a secondary objective.

To eliminate the need for tuning λ and obtain the optimal policy for task performance, we apply the Extrinsic-Intrinsic Policy Optimization (EIPO) [21], [22] framework to locomotion. EIPO was originally designed to remove bias from intrinsic rewards, such as exploration bonuses [23], to ensure an RL algorithm can effectively optimize both intrinsic and extrinsic rewards. To address the issue of minimizing the energy consumption while preserving optimal task performance, we treat energy as an intrinsic reward F and task rewards J as the extrinsic rewards and apply the EIPO

* indicates equal contribution. [1] Improbable AI Lab, MIT, Cambridge, USA. [2] Research Center for Information Technology Innovation, Academia Sinica, Taiwan.

Website: https://srinathm1359.github.io/eipo-locomotion

framework. The key idea of EIPO is to separately optimize both objectives, F and J, using constrained optimization. We treat maximizing J as a constraint and minimizing F as the objective within this constraint. Thus, EIPO identifies policy candidates that maximize task rewards J and selects the policy that minimizes energy consumption F from the identified candidates. This approach guarantees that the learned policy π excels in both task performance and minimizing energy consumption without the necessity of tuning λ.

In our experiments, we evaluate our framework with various terrains and rewarding schemes. Our experimental results in simulation showed that EIPO consistently achieves better task performance and comparable energy consumption to the baseline with the extensively tuned weight of energy penalty λ. Furthermore, we extend our evaluation to real-world scenarios, affirming the applicability and effectiveness of EIPO in practical environments.

II. Preliminaries

Reinforcement Learning for Locomotion: We model locomotion as a discrete-time sequential decision process and aim to learn a policy for a robot. The agent (i.e., robot) starts from an initial observation o_0 representing the robot's initial pose. At each timestep t, the agent perceives the current observation o_t (i.e., robot's proprioception), takes action a_t sampled from the policy (i.e., controller) $\pi(.|o_0 \cdots o_t)$, receives reward r_t and receives the next observation o_{t+1} according to the transition function (i.e., robot's dynamics). The rewards r are referred to as *task rewards*, which will be detailed in Section IV. The agent's goal is to use interactions with the environment to find the optimal policy π such that the expected cumulative reward is maximized:

$$\max_{\pi} J(\pi), \text{where } J(\pi) = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t r_t\Big] \tag{1}$$

where γ denotes a discount factor [1].

Energy Penalty as Auxiliary Objective: There are multiple motivations for minimizing energy consumption as an auxiliary objective: first, this can prevent the policy from consuming excessive energy, and second, energy minimization can occasionally be a useful heuristic for increasing task performance [24]. Thus, energy consumption minimization is commonly considered as an auxiliary objective F to the optimization objective Equation (1). The joint objective to train a policy π is defined as follows:

$$\max_{\pi} J(\pi) - \lambda F(\pi),$$
$$\text{where } J(\pi) - \lambda F(\pi) = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t (r_t - \lambda f_t)\Big] \tag{2}$$

where f_t denotes the auxiliary reward received at timestep t and $\lambda > 0$ controls the weight of auxiliary reward in optimization. The auxiliary reward f is the per-timestep energy consumption, which we define in Section IV.

Algorithm 1 Extrinsic-Intrinsic Policy Optimization

Input: Step sizes η_π, $\eta_{\pi'}$, and η_α
Output: Policy π
Randomly initialize two policies π and π'
Initialize Lagrangian multiplier α
for iteration $= 1 \cdots$ **do**
  Rollout a trajectory by π: $\tau = (o_0, a_0, r_0, o_1, \cdots)$
  Rollout a trajectory by π': $\tau' = (o_0, a_0, r_0, o_1, \cdots)$
  Estimate $\alpha J(\pi) - F(\pi)$ using τ
  Update π: $\pi \leftarrow \pi + \eta_\pi \nabla_\pi (\alpha J(\pi) - F(\pi))$
  Estimate $J(\pi')$ using τ'
  Update π': $\pi \leftarrow \pi + \eta_{\pi'} \nabla_\pi J(\pi')$
  Estimate $J(\pi) - J(\pi')$ using τ and τ'
  Update α with estimates of $J(\pi) - J(\pi')$
end for

III. Method

Although tuning λ in Equation (2) can yield satisfactory task rewards J and energy consumption F, the optimal policy given by Equation (2) may not be optimal for task rewards (i.e., $\arg\max_\pi J(\pi) - \lambda F(\pi) \neq \arg\max_\pi J(\pi)$).

We remove the efforts of tuning λ by reformulating the problem of maximizing task rewards and minimizing energy consumption as a constrained optimization problem, adapting from Extrinsic-Intrinsic Policy Optimization (EIPO) [21]. EIPO optimizes the auxiliary objective $F(\pi)$ under the constraint that the policy achieves the optimal task rewards $J(\pi) = \max_{\pi'} J(\pi')$. The policy optimization objective of EIPO is expressed as follows:

$$\min_{\pi} F(\pi), \text{ where } F(\pi) = \mathbb{E}\Big[\sum_{t=0}^{\infty} \gamma^t f_t\Big]$$
$$\text{subject to } J(\pi) = \max_{\pi'} J(\pi') \tag{3}$$

We adapt the original EIPO algorithm [21] to solve Equation (3) and present the algorithmic details in Algorithm 1. The constraint in Equation (3) is hard to impose since it requires solving the optimal policy with respect to the task reward (i.e., $\max_{\pi'} J(\pi')$). Chen et al. [21] utilize Lagrangian duality to transform Equation (3) into an unconstrained problem in the following max-min optimization objective:

$$\max_{\alpha \geq 0} \min_{\pi} F(\pi) \; - \alpha (J(\pi) - \max_{\pi'} J(\pi')) = \max_{\alpha \geq 0} \min_{\pi} \mathcal{L}(\pi, \alpha),$$
$$\text{where } \mathcal{L}(\pi, \alpha) = F(\pi) - \alpha \left(J(\pi) - \max_{\pi'} J(\pi')\right). \tag{4}$$

α is the Lagrangian multiplier controlling the weight of the task reward in optimization of π. A higher α biases the optimization process towards the task reward $J(\pi)$. The optimal policy π for Equation (3) can be obtained through alternating minimizing $\mathcal{L}(\pi, \alpha)$ for α and maximizing $\mathcal{L}(\pi, \alpha)$ for π. We use stochastic gradient ascent with momentum to learn α, where $\nabla_\alpha \mathcal{L} = J(\pi) - J(\pi')$.

IV. Experimental Design

We aim to answer whether EIPO achieves better task performance while consuming comparable energy levels compared to the common practice of optimizing the sum of task

and energy rewards. We also test whether EIPO can be combined with curriculum learning to obtain a fast command-conditioned policy. We train our methods and baselines in the simulator using the Isaac Gym simulator [25], [26].

A. Experiment Settings

We evaluate the ability of EIPO and PPO to train policies on three different tasks, which we detail below.

1) Velocity Tracking on Flat Ground: The task is to control a quadruped robot to move at least as fast as the user-specified target velocities sampled from the range $[0\ \text{m/s}, 1\ \text{m/s}]$. Given the robot's linear velocity $\mathbf{v}_{xy}$ and angular velocity ω_z, we define the task reward function for velocity tracking based on the velocity tracking error:

$$r_{\text{track}} = \exp\{-(|\min(\mathbf{v}_x - \mathbf{v}_x^{\text{cmd}}, 0)|^2 + |\mathbf{v}_y - \mathbf{v}_y^{\text{cmd}}|^2)/\sigma_{vxy}\} + 0.5 \cdot \exp\{-|\omega_z - \omega_z^{\text{cmd}}|^2/\sigma_{\omega z}\} \quad (5)$$

Here, we always set the y and angular velocity commands $\mathbf{v}_y$ and ω_z to 0. We clip the term $\mathbf{v}_x - \mathbf{v}_x^{\text{cmd}}$ to be at most 0 when computing the linear velocity tracking reward. Thus, the task is for the quadruped to move straight with at least the speed given by the specified x velocity command $\mathbf{v}_x$.

2) Velocity Tracking on Rough Terrain: Similar to the previous task, the quadruped is required to move linearly at a velocity $\mathbf{v}_x$ that is at least the specified command $\mathbf{v}_x^{\text{cmd}}$ to maximize its cumulative rewards. We slightly modify the reward function from above, defining the reward function as the sum of x, y, and angular velocity tracking error:

$$r_{\text{track}} = \exp\{-|\min(\mathbf{v}_x - \mathbf{v}_x^{\text{cmd}}, 0)|^2/\sigma_{vxy}\} + 0.05 \cdot \exp\{-|\mathbf{v}_y - \mathbf{v}_y^{\text{cmd}}|^2/\sigma_{vxy}\} + 0.1 \cdot \exp\{-|\omega_z - \omega_z^{\text{cmd}}|^2/\sigma_{\omega z}\} \quad (6)$$

Different from the previous task, in this task, we randomize the terrain roughness to be between 0 cm and 15 cm modeling the rough terrain as Perlin noise.

3) Maximizing Velocity: The final task is to train a policy capable of achieving maximal linear speed in a straight-line trajectory. The reward function for this task is defined as:

$$r_{\text{vel}} = \mathbf{v}_x \cdot \exp\{-|\mathbf{v}_y - \mathbf{v}_y^{\text{cmd}}|^2/\sigma_{vxy} - |\omega_z - \omega_z^{\text{cmd}}|^2/\sigma_{\omega z}\} \quad (7)$$

To enable better sim-to-real transfer, we train policies with a fixed roughness of 5 cm, and also terminate episodes whenever a part of the quadruped other than the foot experiences contact, which incentivizes the robot to avoid potentially damaging behavior when deployed to the real world.

4) Velocity Tracking with Curriculum Learning: In addition to the above tasks, we also incorporate standard techniques used in quadruped locomotion to get a fast command-conditioned policy and transfer it to the real world. Such techniques include using an actuator model [25] and increased domain randomization for better sim-to-real transfer and curriculum learning [5] to enhance exploration. For this task, the task reward function is defined as:

$$r_{\text{track}} = \exp\{-(|\mathbf{v}_x - \mathbf{v}_x^{\text{cmd}}|^2 + |\mathbf{v}_y - \mathbf{v}_y^{\text{cmd}}|^2)/\sigma_{vxy}\} + 0.5 \cdot \exp\{-|\omega_z - \omega_z^{\text{cmd}}|^2/\sigma_{\omega z}\}. \quad (8)$$

For this task, the y-velocity command is set in $[-0.6, 0.6]$, while the x and angular velocity commands are initially set in $[-1, 1]$. They can increase as training progresses based on the curriculum learning setup in [5], with angular velocity command in $[-10, 10]$ and x-velocity command in $[-1, 10]$. For better sim-to-real transfer, we train policies with terrain roughness randomized between 0 cm and 5 cm. We terminate episodes when a part of the quadruped other than the foot experiences contact and when a power consumption of greater than 800 W occurs in a single timestep. We found this was necessary to avoid triggering the emergency stop when giving high x-velocity commands during deployment.

B. Defining Energy Consumption

As we are interested in learning energy-efficient locomotion policies that do not sacrifice task performance, the auxiliary reward function in this paper is determined by the negative power consumption. Based on the energy metric defined in [27], we define the energy reward function as:

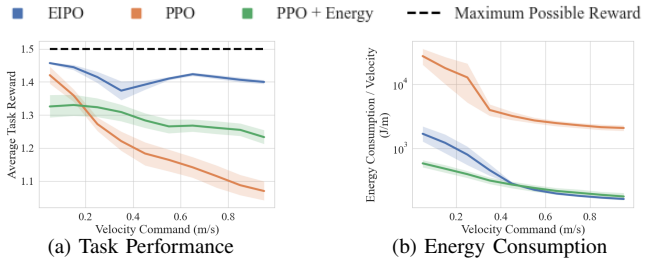
$$r_{\text{energy}} = r_{\text{mechanical}} + r_{\text{heating}} + r_{\text{battery}} - r_{\text{survival}}$$
$$r_{\text{mechanical}} = \text{clip}(\tau, \min = -3, \max = 1\text{e}4)^T \dot{q}$$
$$r_{\text{heating}} = 0.7 \cdot \|\tau * g\|^2, \quad r_{\text{battery}} = 42, \quad r_{\text{survival}} = 200 \quad (9)$$

This reward is multiplied by a negative coefficient to incentivize the robot to use less energy. The parameters of the reward function are not tuned for policy performance and are regressed to explicitly measure its real-world battery consumption. Similar metrics for power consumption have been used for locomotion, with exact formulas depending on the robot [17]. Here, g represents the gear ratios for the Unitree Go1 robot. As the reward would be negative, the robot could be incentivized to terminate the episode early. Thus, we also include a survival bonus of 200 in the reward.

C. Implementation details

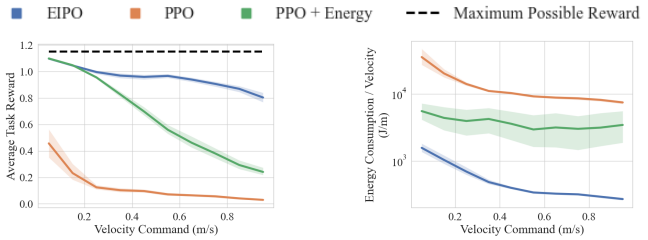
a) Observations and actions: The policy is provided input in the form of an H-step history comprising observations $o_{t-H\ldots t}$, commands $c_{t-H\ldots t}$, previous actions $a_{t-H-1\ldots t-1}$, and timing reference variables $t_{t-H\ldots t}$ [12]. For tasks where the policy does not depend on the command, such as maximizing the velocity of the robot, the commands are fixed to 0. The observation space o_t includes joint positions, velocities $q_t, \dot{q}_t$, and the gravity vector in the body frame g_t, where $q_t, \dot{q}_t$ are measured by joint encoders and g_t is measured by accelerometer. The action a_t consists of position targets for all twelve joints. A zero action corresponds to the nominal joint position, $\hat{q}$. The position targets are tracked using a proportional-derivative controller with $k_p = 20$, $k_d = 0.5$.

b) Model architecture: The model architecture for each PPO policy consists of a Multi-Layer Perceptron (MLP) with hidden layer sizes of $[512, 256, 128]$ and Exponential Linear Unit (ELU) activations. The policy input also incorporates an estimate of the robot body velocity. These parameters are predicted from the observation history using a supervised learning approach [28]. The estimator module consists of an MLP with hidden layer sizes [256, 128] and ELU activations.



(a) Task Performance (b) Energy Consumption

Fig. 2: Comparing EIPO and PPO in Flat Terrain. **(a)** EIPO consistently outperforms both PPO with tuned λ and PPO with no energy reward in task performance. **(b)** The energy consumption of EIPO is slightly greater than PPO with tuned λ, and EIPO is more energy efficient than PPO with no energy reward.



(a) Task Performance (b) Energy Consumption

Fig. 3: Comparing EIPO and PPO in Rough Terrain. **(a)** EIPO, shown in blue, outperforms or matches PPO, both with and without an energy reward, at all velocity commands in terms of task reward. **(b)** EIPO, shown in blue, consumes less energy than PPO.

c) Training hyperparameters: The policy model was trained using Adam [29] with $\beta_1 = 0.9$ and $\beta_2 = 0.999$, learning rate 0.001, gradient clipping 1.0, number of epochs per rollout 5, number of timesteps per rollout 24, and number of minibatches 4. We update the α after each iteration using a learning rate of 0.01, gradient clipping value of 1.0, and momentum coefficient 0.99.

D. Evaluation

We take PPO trained with task and auxiliary rewards weighted by λ as a baseline (Equation (2)). For all experiments, we sweep over the choices of

$$\lambda \in [0, 1\text{e-}4, 2\text{e-}4, 5\text{e-}4, 1\text{e-}3, 2\text{e-}3, 5\text{e-}3, 1\text{e-}2, 2\text{e-}2], \quad (10)$$

which covers a wide range of possible weightings of the different policy objectives. This range covers scenarios where the average energy reward is one order of magnitude less than the task reward to scenarios where it is one order of magnitude more than the task reward.

V. Results in Simulation

To compare EIPO v/s PPO, we train three seeds of policies with both algorithms in all the problem settings described in Section IV-A and report their performance in simulation.

A. Results on Flat Ground

We test whether EIPO can outperform PPO at learning an energy-efficient command-conditioned locomotion policy on flat ground as detailed in Section IV-A.1. We train PPO policies with the task reward defined in Equation (5) and energy consumption reward defined in Equation (9), for each λ defined in Equation (10). Finally, we train EIPO policies where λ is initialized to 0.005. We compute the average performance across random seeds. All policies are trained for 4000 iterations and have a 5-step observation history.

In Figure 2a, we display the task performance of PPO, PPO with auxiliary energy reward (where $\lambda = 0.005$), and EIPO. The task performance is calculated by running each of the policies at each specified x-velocity command, which is between 0 and 1, and computing the sum of rewards obtained by the policy divided by 1000, which is the maximum length of the episode. As shown in Figure 2a, EIPO consistently outperforms both PPO and PPO with an energy reward across velocity commands, showing that including the energy objective helps performance when combined with EIPO.

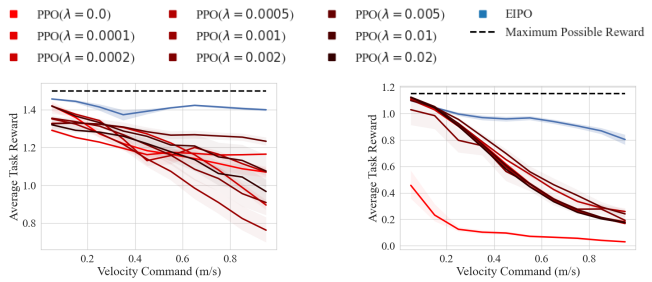
In Figure 2b, we estimate the energy efficiency of the different policies by considering the energy consumption divided by the robot's velocity. We find that PPO with no energy reward consumes a large amount of energy, indicating suboptimal energy consumption. Meanwhile, PPO with the energy reward consumes less energy than EIPO at lower velocity commands but similar amounts of energy at higher commands. This represents EIPO working as intended since EIPO prioritizes task performance over energy minimization, and EIPO achieves a higher task performance in this case.

B. Results on Rough Terrains

Next, we test whether EIPO can outperform PPO at learning an energy-efficient yet high-performing locomotion policy trained on rough terrain reaching up to 15 cm of roughness, as described in Section IV-A.2. Similarly, as in Section V-A, we train policies for each λ given in Equation (10) using PPO as a baseline, and we compare the performance to EIPO. We train policies for 4000 iterations with a 10-step observation history. For this task, we initialize λ to 0.0002 for EIPO. In Figure 3a, we plot the performance of PPO, PPO with a tuned λ for this task ($\lambda = 0.005$), and EIPO across different velocity commands at 15 cm roughness. EIPO decreases in performance as the target velocity increases but is significantly higher than PPO with a tuned λ for velocity commands above 0.4 m/s. PPO fails to perform well without an energy term. EIPO is also more energy efficient than PPO with a tuned λ, as shown in Figure 3b, where we plot the energy divided by velocity. Since the curve for EIPO is far below the curve for PPO with a tuned λ, the EIPO policies are more energy efficient than the policies trained by PPO with a tuned λ.

C. Hyperparameter Study

We analyze the performance of PPO trained with all the different choices for λ, given by Equation (10), in Figure 4 in both the settings described in Section IV-A.1 and Section IV-A.2. As shown in Figure 4a, for the policies trained in flat terrain, EIPO outperforms PPO for all λ, across the entire range of velocity commands in the interval $[0, 1]$ in terms of task reward. For the policies trained in rough terrain, EIPO matches PPO at low-velocity commands in terms of task



(a) Flat Terrain (b) Rough Terrain

Fig. 4: Task Performance of Different Policies Trained in Flat and Rough Terrains. **(a)** EIPO consistently has higher task reward on flat terrain than PPO, regardless of the choice of λ. **(b)** EIPO has much better task reward on rough terrain than PPO.

rewards and outperforms PPO when the velocity command is at least 0.4 m/s, as shown in Figure 4b. Rather than purely acting to regularize the policy's behavior, energy minimization is also helpful for optimizing the task reward, as implied by the fact that PPO improves in performance when using an auxiliary energy penalty in the rough terrain task. This could be due to many optimization-related reasons such as improving exploration or simplifying credit assignment.

D. Results on Velocity Maximization

We also train policies using EIPO and PPO to solve the task described in Section IV-A.3, where the robot must run straight forward as fast as possible to maximize the task reward. We train three seeds per coefficient λ in the set defined by Equation (10) and train three seeds of EIPO, where λ is initialized to 0.0002. Each policy is trained for 10,000 iterations since we observed that policies take longer to converge for this task. Each policy has a 10-step observation history. To respect the torque-speed curve characteristic of all electric motors, we clip the maximum allowed torque for a joint to $\tau_{\max} = 33 \cdot (\dot{q}_{\max} - |\dot{q}|)/\dot{q}_{\max}$, where $\dot{q}$ is the joint velocity and $\dot{q}_{\max}$ is the maximum velocity at that joint specified by the URDF file for the Unitree Go1. We list the peak velocity of the different methods (EIPO, PPO, and PPO with different λ) in Table I, taking the interquartile mean (IQM) over 1000 episodes and the three seeds. We use the IQM to aggregate the performance across episodes and seeds as it is robust to outliers while requiring less data to be statistically significant [30]. We compute the 95% confidence intervals using bootstrapping. As shown in Table I, EIPO achieves a much higher peak velocity in simulation compared to all policies trained by PPO. Penalizing energy consumption hurts task performance when combined with PPO, as it reduces the robot's peak velocity. Therefore, it may be surprising that EIPO can obtain a higher performance when using the energy penalty. We hypothesize that this could be due to EIPO adaptively tuning the weighting of the energy penalty, while PPO holds it constant throughout training.

E. Incorporating Curriculum Learning

Finally, we test whether EIPO can be used to train energy-efficient omni-directional velocity-tracking policies that can travel at high speeds with minimal reward shaping, as described in Section IV-A.4. We train policies with EIPO, PPO with a tuned λ for the task ($\lambda = 0.005$), and PPO using a standard shaped reward used in locomotion works [5], on a range of linear and angular velocity commands using curriculum learning. In addition, we test EIPO's applicability to other auxiliary rewards by training policies using a reward function that incentivizes smooth behavior:

$$r_{\text{smooth}} = 2\text{e-}4 \cdot \|\tau\|^2 + 1\text{e-}2 \cdot \|a\|^2 + 2.5\text{e-}7 \cdot \|\ddot{q}\|^2 \qquad (11)$$

This reward function is a small subset of the reward terms found in [5]. For both EIPO policies, we initialize λ to 0.005. As shown in Figure 5, PPO performs poorly using only task and energy rewards. Meanwhile, EIPO trained using only task and energy rewards performs similarly to PPO trained using the same reward function as [5] while using similar or less energy at all commands. It does not perform as well at high yaw-velocity commands, but performs better at high x-velocity commands, which indicates similar performance between policies. Based on Figure 5, EIPO trained using only task and smoothness rewards also performs similarly and consumes less energy than PPO trained with reward shaping. However, it consumes slightly more energy than EIPO trained using only task and energy consumption rewards.

Algorithm	Peak Vel. (m/s)	Algorithm	Peak Vel. (m/s)
EIPO	**2.86 (2.84, 2.89)**	PPO ($\lambda = 0.001$)	1.64 (1.63, 1.65)
PPO ($\lambda = 0$)	2.12 (2.12, 2.12)	PPO ($\lambda = 0.002$)	1.58 (1.55, 1.60)
PPO ($\lambda = 0.0001$)	1.55 (1.53, 1.56)	PPO ($\lambda = 0.005$)	1.62 (1.61, 1.62)
PPO ($\lambda = 0.0002$)	1.49 (1.48, 1.50)	PPO ($\lambda = 0.01$)	1.45 (1.45, 1.46)
PPO ($\lambda = 0.0005$)	1.58 (1.57, 1.60)	PPO ($\lambda = 0.02$)	1.37 (1.37, 1.37)

TABLE I: Maximum velocity (higher is better) achieved by EIPO and PPO trained with various choices of λ in simulation. EIPO achieves a peak velocity of approximately 0.74 m/s faster than the next fastest policy, PPO trained without any energy reward. The peak velocities of the remaining policies are even slower than EIPO's. 95% confidence intervals are estimated using the bootstrapping method.

VI. Real-World Deployment

We test whether the policies from Section V-D and Section V-E behave similarly when deployed in the real world.

a) Deploying the Maximum Velocity Policy: We deploy the EIPO policy, which obtains a peak velocity of approximately 2.9 m/s in simulation, as shown in Table I. We also deploy a PPO policy, trained with $\lambda = 0.01$, which obtains a peak velocity of approximately 1.5 m/s in simulation. We measure the robot's velocity as it runs forward in Figure 6 over time. As shown by the blue curve in Figure 6, EIPO achieves a peak speed of approximately 2.5 m/s and is still accelerating as the blue curve is still increasing at the end. Meanwhile, the PPO policy achieves a peak speed of approximately 1.5 m/s. Thus, the policies learned in simulation with EIPO can transfer to the real world with similar behavior as in simulation, even at speeds of 2.5 m/s. While the compared policies trained with PPO that run at higher velocities in simulation (specifically, those trained without any energy penalty), these policies were unsafe to deploy based on their behavior in simulation, which we show in the attached video.

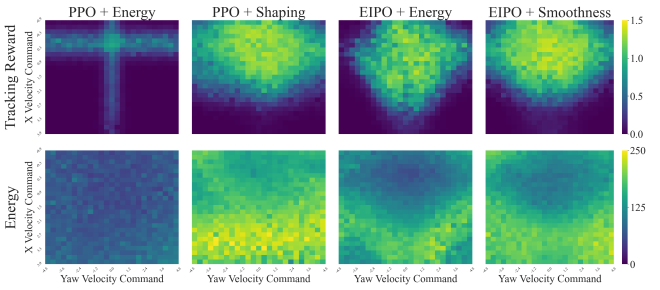


Fig. 5: Task reward and energy consumption across different commands in simulation for PPO trained with only task + energy rewards, PPO trained with reward shaping from [5], EIPO trained with only task + energy rewards, and EIPO trained with the smoothness reward (Equation (11)). As shown in the figure, EIPO with a simple auxiliary reward for this task performs similarly to complex reward shaping functions.

b) Deploying the Curriculum Trained Policy: We deploy the EIPO policy trained with only task and energy consumption rewards in the real world. As shown in Figure 1, the gait that emerges from training is stable and animal-like. It exhibits a variable contact schedule. It has a longer stance phase when walking at low speeds and a gallop when running at top speed: an asymmetrical gait with a four-beat rhythm and a suspension phase. This gait is enabled by the minimal reward shaping used in our method, in contrast to prior works that use reward shaping to incentivize particular gaits or reference trajectories to specify the desired behavior.

VII. Related Work

Prior works in learning legged locomotion have not exclusively optimized the policy for velocity tracking or forward progress but instead used a reward function with a weighted sum of terms to express energy minimization, safety, and other objectives of the robot's motion, with common reward terms developed gradually over time [2], [3], [10], [12], [26], [31]. A common and simple reward formulation consists of minimizing energy consumption and maximizing speed or velocity tracking performance. Yang et al. [32] used a hierarchical architecture where the learned policy predicts a gait style command for a low-level MPC controller and varies the command across speeds to improve energy efficiency. Fu et al. [16] optimized an end-to-end policy with a reward function consisting of a linear function of energy minimization and velocity tracking. Another line of work has incorporated reference motion capture footage to aid learning, also using an additional reward term or additive policy loss [33]–[35]. Some recent promising directions are to simplify or generalize the task specification by using a point reaching formulation [13] or explicitly formulating some terms as constraints [15], [36]. However, in all cases, the policy optimization objective consists of a weighted sum of locomotion and auxiliary reward where the weighting is a hyperparameter that must be tuned. In contrast, our work leverages the EIPO framework to directly optimize the locomotion task performance and automatically tunes the auxiliary rewards to ensure that they aid in the task.

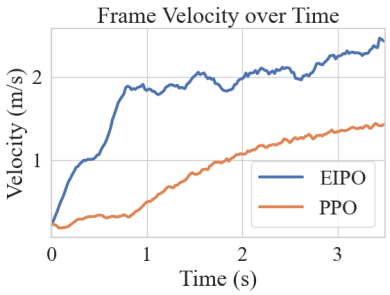


Fig. 6: Velocity of the quadruped with EIPO and PPO policy deployed in the real world. The EIPO policy significantly outperforms the PPO policy, having a peak speed of almost 1m/s higher.

VIII. Discussion

Our experiments demonstrate that EIPO can improve task performance while being comparably energy efficient to PPO in locomotion tasks. Specifically, we observed that in two velocity tracking tasks, policies trained with EIPO outperformed those trained with PPO in adhering to velocity commands, particularly on challenging terrain. Additionally, EIPO policies exhibited energy efficiency on par with or even superior to PPO policies. Furthermore, in a velocity maximization task, policies trained with EIPO achieved significantly higher speed than PPO policies, and these EIPO-trained policies demonstrated equally effective performance in the real world.

Based on our results, a promising direction for further research is to apply EIPO to more domains for motor control, such as manipulation and in-hand reorientation, to see if it improves performance beyond PPO in even more general settings. Another direction for further research is exploring which auxiliary reward terms work better for different types of tasks when combined with EIPO. Previously, EIPO has only been found to improve performance in Atari games [21] when combined with novelty-based intrinsic rewards such as Random Network Distillation [23]. Our work identifies energy consumption as an auxiliary reward that improves performance in locomotion tasks when combined with EIPO.

Acknowledgement

The contributions of all authors are listed at the project website. We thank the members of the Improbable AI lab for the helpful discussions and feedback on the paper. We are grateful to MIT Supercloud and the Lincoln Laboratory Supercomputing Center for providing HPC resources. This research was partly supported by Hyundai Motor Company, DARPA Machine Common Sense Program, the MIT-IBM Watson AI Lab, and the National Science Foundation under Cooperative Agreement PHY-2019786 (The NSF AI Institute for Artificial Intelligence and Fundamental Interactions, http://iaifi.org/). We acknowledge support from ONR MURI under grant number N00014-22-1-2740. Research was sponsored by the Army Research Office and was accomplished under Grant Number W911NF-21-1-0328. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein. This research was also supported by the Paul E. Gray (1954) UROP Fund and the John Reed Fund. This work received partial support from the Biomedical Acoustic Signal Processing (Bio-ASP) Laboratory, led by Dr. Yu Tsao, at the Research Center for Information Technology Innovation, Academia Sinica, Taiwan.

References

[1] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, 2018.

[2] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, p. eaau5872, 2019.

[3] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "RMA: Rapid motor adaptation for legged robots," *Robotics: Science and Systems*, 2021.

[4] G. B. Margolis, T. Chen, K. Paigwar, X. Fu, D. Kim, S. Kim, and P. Agrawal, "Learning to jump from pixels," *Conference on Robot Learning*, 2021.

[5] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," *Robotics: Science and Systems*, 2022.

[6] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning robust perceptive locomotion for quadrupedal robots in the wild," *Science Robotics*, vol. 7, no. 62, p. eabk2822, 2022.

[7] Y. Ji, G. B. Margolis, and P. Agrawal, "Dribblebot: Dynamic legged manipulation in the wild," *arXiv preprint arXiv:2304.01159*, 2023.

[8] T. Chen, J. Xu, and P. Agrawal, "A system for general in-hand object re-orientation," in *Conference on Robot Learning*. PMLR, 2022, pp. 297–307.

[9] T. Chen, M. Tippur, S. Wu, V. Kumar, E. Adelson, and P. Agrawal, "Visual dexterity: In-hand dexterous manipulation from depth," *arXiv preprint arXiv:2211.11744*, 2022.

[10] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke, "Sim-to-real: Learning agile locomotion for quadruped robots," *arXiv preprint arXiv:1804.10332*, 2018.

[11] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 3803–3810.

[12] G. B. Margolis and P. Agrawal, "Walk these ways: Tuning robot control for generalization with multiplicity of behavior," in *Conference on Robot Learning*, 2023.

[13] N. Rudin, D. Hoeller, M. Bjelonic, and M. Hutter, "Advanced skills by learning locomotion and local navigation end-to-end," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 2497–2503.

[14] Z. Fu, X. Cheng, and D. Pathak, "Deep whole-body control: learning a unified policy for manipulation and locomotion," in *Conference on Robot Learning*. PMLR, 2023, pp. 138–149.

[15] Y. Kim, H. Oh, J. Lee, J. Choi, G. Ji, M. Jung, D. Youm, and J. Hwangbo, "Not only rewards but also constraints: Applications on legged robot locomotion," *arXiv preprint arXiv:2308.12517*, 2023.

[16] Z. Fu, A. Kumar, J. Malik, and D. Pathak, "Minimizing energy consumption leads to the emergence of gaits in legged robots," in *Conference on Robot Learning*, 2021.

[17] Y. Yang, T. Zhang, E. Coumans, J. Tan, and B. Boots, "Fast and efficient locomotion via learned gait transitions," in *Conference on Robot Learning*. PMLR, 2022, pp. 773–783.

[18] D. F. Hoyt and C. R. Taylor, "Gait and the energetics of locomotion in horses," *Nature*, vol. 292, no. 5820, pp. 239–240, 1981.

[19] J. E. Bertram, "Constrained optimization in human walking: cost minimization and gait plasticity," *Journal of experimental biology*, vol. 208, no. 6, pp. 979–991, 2005.

[20] R. Yang, X. Sun, and K. Narasimhan, "A generalized algorithm for multi-objective reinforcement learning and policy adaptation," *Advances in neural information processing systems*, vol. 32, 2019.

[21] E. Chen*, Z.-W. Hong*, J. Pajarinen, and P. Agrawal, "Redeeming intrinsic rewards via constrained optimization," *Advances in Neural Information Processing Systems*, vol. 35, pp. 4996–5008, 2022.

[22] I. Shenfeld, Z.-W. Hong, A. Tamar, and P. Agrawal, "TGRL: An algorithm for teacher guided reinforcement learning," in *Proceedings of the 40th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, Eds., vol. 202. PMLR, 23–29 Jul 2023, pp. 31 077–31 093. [Online]. Available: https://proceedings.mlr.press/v202/shenfeld23a.html

[23] Y. Burda, H. Edwards, A. Storkey, and O. Klimov, "Exploration by random network distillation," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=H1lJJnR5Ym

[24] A. Y. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *Icml*, vol. 99. Citeseer, 1999, pp. 278–287.

[25] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, *et al.*, "Isaac gym: High performance gpu-based physics simulation for robot learning," *arXiv preprint arXiv:2108.10470*, 2021.

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

[27] S. Seok, A. Wang, M. Y. Chuah, D. J. Hyun, J. Lee, D. M. Otten, J. H. Lang, and S. Kim, "Design principles for energy-efficient legged locomotion and implementation on the mit cheetah robot," *Ieee/asme transactions on mechatronics*, vol. 20, no. 3, pp. 1117–1129, 2014.

[28] G. Ji, J. Mun, H. Kim, and J. Hwangbo, "Concurrent training of a control policy and a state estimator for dynamic and robust legged locomotion," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4630–4637, 2022.

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[30] R. Agarwal, M. Schwarzer, P. S. Castro, A. C. Courville, and M. Bellemare, "Deep reinforcement learning at the edge of the statistical precipice," *Advances in neural information processing systems*, vol. 34, pp. 29 304–29 320, 2021.

[31] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science robotics*, vol. 5, no. 47, p. eabc5986, 2020.

[32] Y. Yang, T. Zhang, E. Coumans, J. Tan, and B. Boots, "Fast and efficient locomotion via learned gait transitions," in *Conference on Robot Learning*, 2021.

[33] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa, "Amp: Adversarial motion priors for stylized physics-based character control," *ACM Transactions on Graphics (ToG)*, vol. 40, no. 4, pp. 1–20, 2021.

[34] A. Escontrela, X. B. Peng, W. Yu, T. Zhang, A. Iscen, K. Goldberg, and P. Abbeel, "Adversarial motion priors make good substitutes for complex reward functions," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 25–32.

[35] Y. Fuchioka, Z. Xie, and M. Van de Panne, "Opt-mimic: Imitation of optimized trajectories for dynamic quadruped behaviors," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 5092–5098.

[36] J. Lee, L. Schroth, V. Klemm, M. Bjelonic, A. Reske, and M. Hutter, "Evaluation of constrained reinforcement learning algorithms for legged locomotion," *arXiv preprint arXiv:2309.15430*, 2023.