Benchmarking Potential Based Rewards for Learning Humanoid Locomotion

Se Hwan Jeon[1], Steve Heim[1], Charles Khazoom[1], and Sangbae Kim[1]

***Abstract*— The main challenge in developing effective reinforcement learning (RL) pipelines is often the design and tuning the reward functions. Well-designed shaping reward can lead to significantly faster learning. Naively formulated rewards, however, can conflict with the desired behavior and result in overfitting or even erratic performance if not properly tuned. In theory, the broad class of *potential based reward shaping* (PBRS) can help guide the learning process without affecting the optimal policy. Although several studies have explored the use of potential based reward shaping to accelerate learning convergence, most have been limited to grid-worlds and low-dimensional systems, and RL in robotics has predominantly relied on standard forms of reward shaping.**

In this paper, we benchmark standard forms of shaping with PBRS for a humanoid robot. We find that in this high-dimensional system, PBRS has only marginal benefits in convergence speed. However, the PBRS reward terms are significantly more robust to scaling than typical reward shaping approaches, and thus easier to tune.

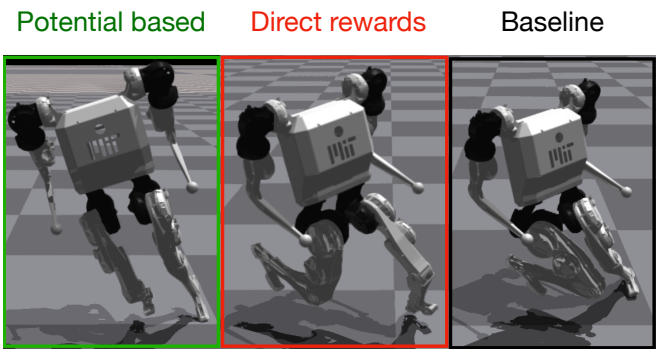


Fig. 1: The potential based (left), direct (middle), and baseline (right) locomotion policies. The controller is capable of forward velocities and yaw rates of approximately 3.5 m/s and 1.5 rad/s, respectively.

I. INTRODUCTION

Designing effective reward functions is an iterative process in optimal control pipelines, both when using model-based methods such as model-predictive control (MPC) or model-free methods such as reinforcement learning (RL) [1], [2]. A simple translation of the engineer's intent into a computable function, such as a quadratic error from the desired state or a boolean on task success, often results in unexpected and undesired behavior [3], [4], especially in RL. Even when the chosen reward function would yield the desired optimal controller, it can often result in slow convergence and local minima. These challenges are typically addressed with *reward shaping*: additional reward terms are added to provide an informative signal of how "close" a trajectory is to an optimal policy. However, defining what "close" means in this context is often not intuitive. In practice, a substantial amount of time is spent tuning these shaping rewards to find an acceptable trade-off between convergence and how closely it represents the desired behavior. Moreover, the entire training process is sensitive to hyperparameters and reward weights, obscuring the effects of a particular reward term on the converged policy and making them challenging to tune precisely [1], [5].

Ng, Harada, and Russell [6] discuss the special class of *potential-based reward shaping* functions (PBRS) that, in theory, do not affect the final policy. This theoretical property is highly attractive since PBRS has the potential to decouple the challenges of reward design, allowing the engineer to use simple, task-based rewards to express intended behavior and PBRS to aid convergence. Several subsequent studies have investigated using PBRS in RL [7]–[10] and reported faster convergence, though these studies are typically limited to grid-worlds or low-dimensional systems. In practice, however, most successful cases of RL in robotics have relied on simpler, direct reward shaping functions that do not enjoy policy invariance properties [1], [11]–[14].

We present an empirical case study of deep RL for running on a humanoid biped robot in 3D, and systematically compare standard reward shaping and PBRS. Unlike many previous studies, we find that the main advantage of the PBRS is not in faster convergence, which we found to be only marginal. The PBRS is, however, substantially more robust than standard shaping, making tuning the reward shaping functions much easier.

A. Related Work

A common approach to provide dense rewards is to directly track references that are highly correlated with the desired behavior. For example, Peng, Coumans, Zhang, Lee, Tan, and Levine [15] use motion capture data from dogs to generate reference motion for RL for a quadruped robot, and Rai, Antonova, Song, Martin, Geyer, and Atkeson [12] use clinical data of humans to obtain reference values such as desired body height and orientation for a bipedal walking robot. Green, Godse, Dao, Hatton, Fern, and Hurst [16] precompute a library of reference trajectories for a simpler, lower-dimensional system that is amenable to model-based

[1]All authors are with the Biomimetic Robotics Lab, MIT {sehwan, sheim, ckhaz, sangbae}@mit.edu.

This work was supported by Disney Research Imagineering, NSERC, and the Swiss National Science Foundation (Grant No P2SKP2_194954).

trajectory optimization, and use this library for reward shaping in RL on a bipedal robot. In a similar fashion, Reda, Ling, and Panne [17] solve for an optimal policy on a simpler model using RL, and use these outputs as shaping rewards for learning brachiation in a simulated 2D animation. In all these examples, it is critical to carefully choose the references and the weighting of the shaping rewards, as the policy can learn to overfit to the references instead of the intended task.

In an attempt to side-step this problem, Ng, Harada, and Russell [6] show policy invariance to PBRS (see eq. (4)), and recommend using an approximation of the value function. Using a value function estimate in PBRS form can be seen as performing credit assignment, redistributing the value as instantaneous rewards throughout state-space [18, Sec. 4]. This is particularly helpful if the baseline reward is sparse, such as a boolean indicator of task-completion. Indeed, Wiewiora [19] showed that PBRS is equivalent to initializing Q-values. Since the optimal policy is greedy with respect to the (true optimal) Q-value function, a well chosen PBRS essentially allows the discount factor to be much more myopic. Westenbroek, Castaneda, Agrawal, Sastry, and Sreenath [10] leverage this property for sample-efficient learning directly in hardware, using a discount factor of zero. Though the reward-shaping is justified with control Lyapunov functions, the main case study on a cartpole uses the value function obtained in simulation and is directly equivalent to PBRS with a value function. From ablation studies, they also observed that if the value function used is too inaccurate, it is necessary to increase the discount factor.

Harutyunyan, Devlin, Vrancx, and Nowe [8] and others [7], [9], [20] have explored using PBRS in a more general setting, and consistently find that PBRS greatly accelerates convergence on simple problems such as gridworlds or cartpole balancing. Malysheva, Kudenko, and Shpilman [7] learn locomotion on a higher-dimensional biped constrained to 2D using references similar to those discussed above, but put in PBRS form. They also report faster convergence, though this is strongly influenced by the quality of the references used.

B. Outline

In Section II, we review concepts and terms necessary to describe PBRS and its implications for reinforcement learning. In Section III, we detail the system, observations, and rewards we use for the locomotion task. In Section IV, we compare the effects of PBRS and DRS for training and on the converged policies. Lastly, in Section V, we present our conclusions and outline future directions for using PBRS in RL.

II. Preliminaries and Problem Statement

A. Markov Decision Processes and Reinforcement Learning

A Markov decision process (MDP) is a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, r, \gamma_t)$, with state space $\mathcal{S}$, action space $\mathcal{A}$, transition probabilities $\mathcal{T}(s_{k+1}|s_k, a_k)$ describing the dynamics of the system, reward function r and discount factor γ_t. The reward function r takes the general form $r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ with $r(s_k, a_k, s_{k+1})$ describing the reward received for transitioning to state s_{k+1} from s_k with action a_k.

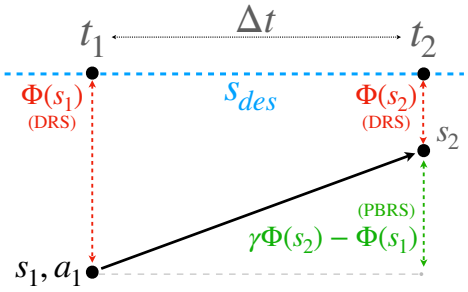


Fig. 2: A visualization of a tracking reward in both direct-reward shaping and potential-based reward shaping form. While DRS rewards return the instantaneous evaluation of Φ, PBRS rewards give returns for *improvement* in Φ at the next state.

For a given MDP, we wish to find a policy $\pi : \mathcal{S} \to \mathcal{A}$ to maximize the discounted sum of all future rewards. For a fixed policy, this quantity is represented as the optimal value function, given as

$$V^*(s_k) = \mathbb{E}\left[\sum_k^{\infty} \gamma_t^k r(s_k, \pi(s_k), s_{k+1})\right]. \tag{1}$$

B. Reward Shaping

Learning the value function is especially challenging when long-term optimality differs strongly from short-term reward signals, for example for sparse, task-based rewards. In these cases, it is typically necessary to specify a discount factor γ close to one, to emphasize the importance of actions on long-term outcomes. It is more challenging because the agent needs to deal with a more ambiguous credit assignment problem. A central theme to reward shaping is to provide a reward signal that is more immediately informative about the current action's effect on the final outcome. This is typically done by formulating *dense* rewards so that informative reward signals are available throughout the trajectory.

We denote these shaping rewards as $R(s_k, a_k, s_{k+1})$, and the corresponding total reward $\hat{r}$ and MDP $\mathcal{M}_{shaped}$ as

$$\hat{r}(s_k, a_k, s_{k+1}) = r(s_k, a_k, s_{k+1}) + R(s_k, a_k, s_{k+1}) \tag{2}$$

$$\mathcal{M}_{shaped} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \hat{r}, \gamma). \tag{3}$$

When well chosen, shaping rewards can greatly help convergence. However, it is important to note that adding shaping terms fundamentally changes the MDP and can have unintended effects on the desired behavior, as discussed by Randløv and Alstrøm [3].

We note that in most RL in robotics studies [11], [16], [21], [22], reward terms are typically restricted to functions that can be computed as $r(a)$, and $r(s)$ or $R(s)$. We will slightly abuse notation and refer to $R(s)$ as *direct reward shaping* (DRS).

TABLE I: Agent Observations

Observation	Dim.	Noise Range
Joint positions $\mathbf{q}$	10	0.005
Joint velocities $\dot{\mathbf{q}}$	10	0.01
Body height z_b	1	0.05
Body velocity $\mathbf{v}_b$	3	0.1
Body angular velocity ω_b	3	0.05
Body frame gravity $\hat{\mathbf{g}}$	3	0.05
Binary foot contact state $\mathbf{b}_c$	2	0.1
Commanded velocities $\begin{cases} c_x \text{ (forward)} \\ c_y \text{ (lateral)} \\ c_\omega \text{ (yaw)} \end{cases}$	3	0
Clock phase $\begin{cases} \sin(\phi) \\ \cos(\phi) \\ \frac{\sin(\phi)}{2\sqrt{\sin(\phi)^2+0.04}} + 0.5 \end{cases}$	3	0

TABLE II: Training rewards

Baseline	Weight	Function
Linear velocity	10	$\exp(-\lvert v_{x,y} - c_{x,y}\rvert^2/\sigma_{xy})$
Angular velocity	5	$\exp(-(\omega_z - c_\omega)^2/\sigma_\omega)$
1st order action rate	-1e-3	$\lvert(q_\pi^k - q_\pi^{k-1})/\Delta t\rvert^2$
2nd order action rate	-1e-4	$\lvert(q_\pi^k - 2q_\pi^{k-1} + q_\pi^{k-2})/\Delta t\rvert^2$
Torques	-1e-4	$\lvert\boldsymbol{\tau}\rvert^2$
Torque limits	-0.01	$\max(\lvert\boldsymbol{\tau}\rvert - \beta_\tau \boldsymbol{\tau}_{max}, 0)$
Joint limits	-10	$\max(\lvert\boldsymbol{\tau}\rvert - \beta_q \boldsymbol{\tau}_{max}, 0)$
Termination	-100	$\begin{cases} 1, \lvert\mathbf{v}_b\rvert \geq 10 \text{ [m/s]}, \\ 1, \lvert\boldsymbol{\omega}_b\rvert \geq 5 \text{ [rad/s]}, \\ 1, \hat{g}_x, \hat{g}_y \geq 0.7, \\ 1, \text{self-collision}, \\ 0, \text{otherwise}. \end{cases}$
Direct Shaping	**Weight**	**Function**
Orientation R_{ori}	5.0	$\exp(-(\hat{g}_x^2 + \hat{g}_y^2)/\sigma_\theta)$
Height R_{h}	2.0	$\exp(-(z_b - z_{des})^2/\sigma_h)$
Joint regularization R_{j}	1.0	$\exp(-(q_a^L - q_a^R)^2/\sigma_q) + \exp(-(q_p^L - q_p^R)^2/\sigma_q) + \exp(-(q_y^L)^2/\sigma_q) + \exp(-(q_y^R)^2/\sigma_q)$
Potential Shaping	**Weight**	**Function**
Orientation	1.0	$\gamma R_{\text{ori}}(\mathbf{s}_{k+1}) - R_{\text{ori}}(\mathbf{s}_k)$
Height	1.0	$\gamma R_{\text{h}}(\mathbf{s}_{k+1}) - R_{\text{h}}(\mathbf{s}_k)$
Joint regularization	1.0	$\gamma R_{\text{j}}(\mathbf{s}_{k+1}) - R_{\text{j}}(\mathbf{s}_k)$

C. Potential-based Shaping

The focus of this paper is *potential-based shaping* of the reward function [6]. Consider a modified MDP, $\mathcal{M}_{potential} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \tilde{r}, \gamma)$, defined with

$$P(s_k, s_{k+1}) = \gamma\Phi(s_{k+1}) - \Phi(s_k) \tag{4}$$

$$\tilde{r}(s_k, a_k, s_{k+1}) = r(s_k, a_k, s_{k+1}) + P(s_k, s_{k+1}), \tag{5}$$

where $\Phi(\cdot)$ is some scalar, real-valued function and $P(\cdot)$ is the potential-based reward. As shown in Fig. 2, potential based rewards are concerned with the *change* of the rewards pushed through the dynamics, as opposed to their instantaneous values.

Note that any DRS term $R(s)$ can trivially be used as a potential function for PBRS, and for the rest of the paper we will focus on comparisons where we use $\Phi(s) = R(s)$. Ng, Harada, and Russell [6] presented theoretical results demonstrating that even for arbitrary potential functions, the optimal policy for the original MDP $\mathcal{M}$ is invariant to this class of potential-based shaping rewards. Furthermore, the advantage function and policy gradients are also unaffected by the addition of $P(s_k, s_{k+1})$ to the original set of rewards [18]. This means that, in theory, an agent trained on the shaped MDP $\mathcal{M}_{shaped}$ should converge to a policy that is also optimal for the original MDP $\mathcal{M}$.

In practice, however, RL algorithms are affected by a myriad of hyperparameters, such as function approximation choices and exploration heuristics, that prevent the agent from converging to the true optimal policy in reasonable time. We focus on comparing the performance of shaping with DRS and PBRS in a practical setting for training a humanoid robot to run.

III. Humanoid Locomotion Case Study

To empirically test the effectiveness of PBRS for continuous, high-dimensional robot control, we benchmark a learning pipeline for running with the MIT Humanoid robot. We start with a minimal set of *baseline rewards* and then benchmark a set of commonly used DRS reward terms and the same set of shaping rewards reformulated as PBRS reward terms.

A. System Overview

The MIT Humanoid is an 18 degree-of-freedom robotic platform designed by the Biomimetic Robotics Lab [23]. For learning running locomotion, we fix the arm joints at nominal angles and reduce control of the system to only the legs, a total of 10 degrees of freedom.

The policy network consists of a single neural network that outputs joint position targets $\mathbf{a} \in \mathbb{R}^{10}$ to the system, similar to prior work [11], [24]. The torques are calculated as

$$\boldsymbol{\tau} = K_p(\mathbf{a} - \mathbf{q}) + K_d(\dot{\mathbf{q}}), \tag{6}$$

where $K_p = 30$ Nm/rad and $K_d = 5$ Nms/rad are fixed proportional and damping gains respectively, and $\mathbf{q}$ are the joint angles.

The observations $\mathbf{s} \in \mathbb{R}^{38}$ are listed in Table I, and are affected by uniformly sampled noise. The phase ϕ is a simple clock with constant growth at one Hz, which we found helpful for the policy to settle into a periodic gait, although the final gaits observed are not limited to this frequency.

TABLE III: Training Environment Hyperparameters

Hyperparameter	Value
Value loss coefficient	1.0
Clipping ϵ	0.2
Entropy coefficient	0.1
Learning rate	1e-5 (adaptive)
Discount factor	0.99
λ	0.95
Steps/env	24
Policy network size	[256, 256, 256]
Critic network size	[256, 256, 256]
Activation	ELU

B. Baseline Rewards

For general locomotion, we define a set of baseline rewards as in Table II. The first two, linear velocity and angular velocity tracking, are the only task-related rewards; all other terms are generic regularization terms to encourage smoothness, efficiency, and discourage joint limit violations.

For the baseline rewards, Δt is the controller timestep, $\sigma = 0.5$ is a scaling parameter, $\beta_\tau = 0.8$ and $\beta_q = 0.9$ act as soft-stop limits to discourage reaching the joint and actuator constraints, and $\boldsymbol{\tau}_{max}$ and q_{max} are torque and joint limits of the system respectively.

C. Shaping Rewards

We choose three shaping rewards commonly used as costs in the humanoid locomotion literature [12], [25], [26]: we regularize the *orientation* (R_{ori}), *height* (R_{h}), and *joints* (R_{j}), with their respective reward terms defined in Table II. A nominal desired height $z_{des} = 0.6$ m is a hand-chosen height target, $\hat{g}_x$, $\hat{g}_y$ are the components of the gravity vector in the body frame, and q_i^j is the i^{th} joint type on leg j. The subscript denotes the specific joint, with q_a, q_p, and q_y representing the abduction/adduction, pitch, and yaw joints specifically, and the superscript refers to the leg (left/right) the joint is part of. The reward R_{joint} serves to regularize the yaw joints about zero and encourage symmetry between the ab/ad and pitch joints of the legs. We use squared-exponential functions to define our reward functions, as is common in RL literature [11], [15], [22], [24].

We can put these "direct" shaping rewards in their potential based forms trivially as

$$P_{\text{s}}(\mathbf{s}_k, \mathbf{s}_{k+1}) = \gamma R_{\text{s}}(\mathbf{s}_{k+1}) - R_{\text{s}}(\mathbf{s}_k), \tag{7}$$

for some arbitrary shaping reward R_{s} and potential discounting γ.

D. Implementation Details

The locomotion policy is trained in the NVIDIA Isaac-Gym framework open-sourced by Rudin, Hoeller, Reist, and Hutter [21] with the PPO-Clip algorithm [27] and the hyperparameters shown in Table III. The agents are trained on a computer equipped with an Intel i9-10850K processor and NVIDIA RTX 3060 GPU. The simulation is run at 1000 Hz, with a control frequency of 100 Hz. Each training run includes 4096 agents and is run for 1000 policy iterations, and we see convergence within around 30 minutes of wall-clock time.

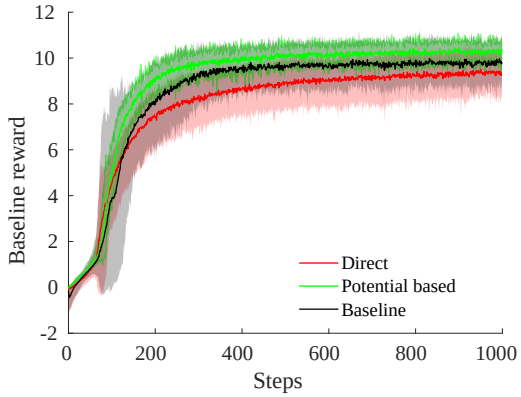


Fig. 3: Values for the total baseline rewards during training for the PBRS, DRS, and baseline policies.

IV. Results

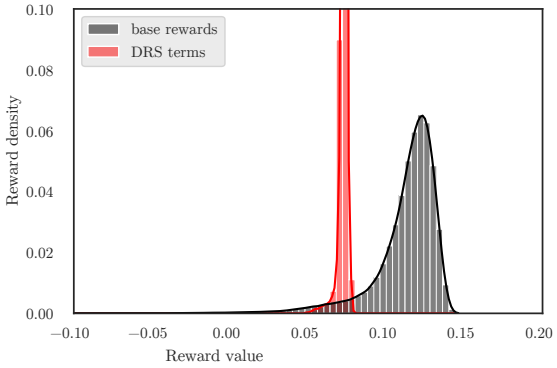
We present here a benchmark with three cases: baseline rewards only, baseline rewards with DRS terms, and baseline rewards with PBRS terms, with accompanying video results and code[1][2]. We first tune the baseline reward weights until reasonable running performance is achieved (see Table II), then keep those weights fixed for all experiments. The weights of the DRS and PBRS rewards are then tuned until a reasonable locomotion policy is found, with the weights set to the tuned values in Table II. Cases are compared with accumulated baseline rewards and not the total rewards, such that the comparisons are not affected by the scaling of shaping rewards. We also visually inspect policies to evaluate the resulting behavior for qualitative differences.

Trajectories are collected over 24 timesteps, equating to roughly 0.2 s of simulation time, and the agents are subjected to randomized impulses, friction, and velocity commands during training.

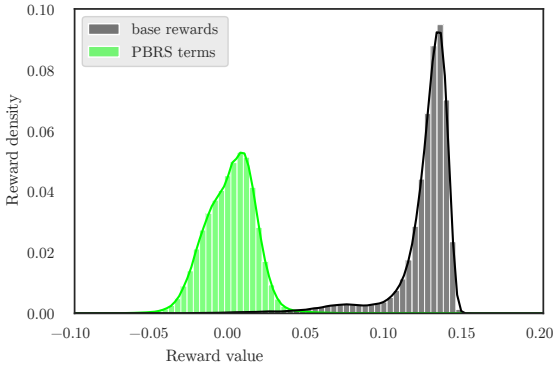
A. Discounting of Potential-Based Rewards

We find that in practice, using discounting for PBRS in (4) can lead to learning instability [28]. To overcome this issue, we set $\gamma = 1$ in the calculation of (4) (though not in calculating the advantage for PPO). While policy invariance is technically sacrificed by doing so, we find that training converges far more stably and quickly with this modification. We do not discuss it further as our findings to this regard closely match those of Grzes and Kudenko [28], who studied in detail how the discount factor in potential-based rewards can affect both the magnitude and sign of the returned value. We confirm their finding, as other studies on PBRS do not report any modifications of the discounting [7], [9], [20], yet we found this to make a significant difference in the effectiveness of PBRS terms.

[1]Video: https://youtu.be/Qvacov9kujQ
[2]Code: https://github.com/se-hwan/pbrs-humanoid



(a) Distribution at iteration 500



(b) Distribution at iteration 500

Fig. 4: Distribution of rewards during an iteration at iteration 500 (well converged). Although the distributions of rewards begins rather spread out, agents tend to quickly fit to dense DRS terms (4a), whereas the distribution of PBRS terms remains centered around zero and with a relatively wide distribution throughout training (4b). See accompanying video for the evolution of this distribution during training.

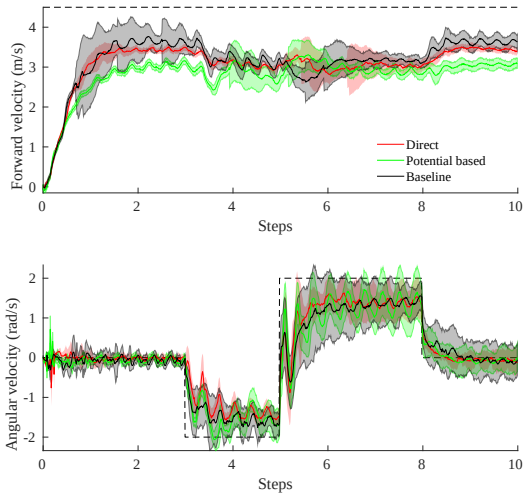


Fig. 5: Velocity tracking plots for the PBRS, DRS, and baseline policies over 50 runs. The dotted line indicates the commanded forward velocity (top) and yaw angular rate (bottom), respectively.

B. Training Benchmark

We run the three cases (baseline, DRS, and PBRS) ten times each with the tuned, nominal weights for all rewards. All three cases converge to behaviors with similar performance within 1000 iterations, as seen in the learning curves shown in Fig. 3. The agent with PBRS terms converges to a policy with slightly higher return, slightly more quickly, though this improvement is relativey marginal. The variance between the ten learning runs, however, is significantly lower, finishing at 0.946, compared to 1.905 when using DRS terms, and 2.025 when using baseline rewards only.

Agents trained with DRS terms converge to policies with slightly lower returns than the baseline; however, when inspecting the policies, we find that baseline policies tend to have abnormal gaits, with legs turned inward. The most likely explanation for this is the greater mediolateral stability it provides, which assists with avoiding termination early on in training. This behavior persists and appears detrimental when turning at high velocities, as shown in Figs. 1 and 5. Both DRS and PBRS terms rectify this issue by regularizing the yaw joints around zero.

In the case of DRS, however, the agent appears to strongly prioritize maximizing the shaping reward terms, which likely conflicts with maximizing the baseline reward. This can be seen when inspecting the reward distribution during training, shown in Fig. 4: the PBRS terms remain centered around 0 and maintain a relatively large spread, even halfway through training, whereas the DRS terms are very quickly maximized, which suggests the agent prioritizes maximizing the shaping rewards. This empirical observation supports the theory behind the policy invariance of PBRS. Because PBRS terms become zero-mean centered as training progresses, their influence on the optimizer decreases, allowing greater returns on the baseline rewards.

Another interesting result is the emergence of natural heel-toe transitions for the trained policies. While none of the rewards explicitly specify this behavior, the touchdown and push-off phases of stance are quite clear. It is possible that the relatively low K_p gains on the joints force the robot into this pattern, but isolating the rewards and environment configurations that can reproduce this behavior is beyond the scope of this work. We present these details as examples of how desirable policies can be both difficult to express and sometimes counterintuitive.

To evaluate the policy itself, we compare the linear and angular velocity tracking performance of the three policies at the limits of the commands, as shown in Fig. 5. We find no significant difference in command tracking on average between the DRS and PBRS policies, but observed that the baseline policy often terminates during the turns, which accounts for the large standard deviation in angular velocity

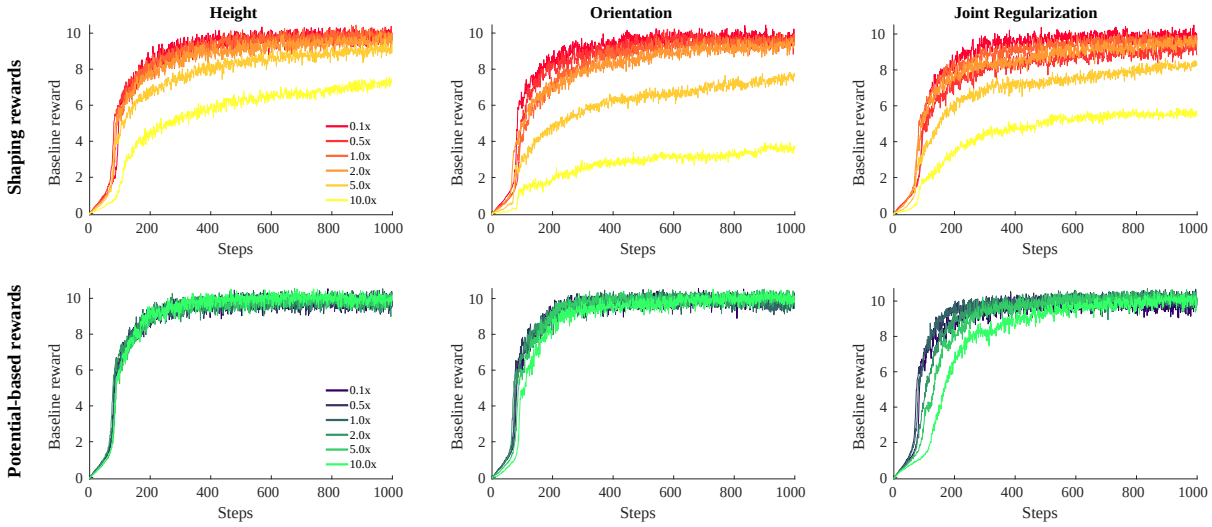


Fig. 6: To compare the robustness of using potential-based shaping against direct reward shaping, we train policies with the baseline rewards in combination with either the height, orientation, or joint regularization reward. From a tuned nominal value, we then sweep across the weights of that reward from 0.1× to 10× the nominal value. The potential-based rewards were far less sensitive to large changes in scaling compared to the shaping rewards.

between $t = 3$ and $t = 10$ s in Fig. 5. While the DRS policy tracks the desired velocity most closely, this metric alone does not account for all the other terms defined as part of the "baseline reward", which overall, is higher for the potential-based policy.

We also compare the average base heights of the DRS and PBRS through the trajectory. As expected, the average height of the DRS policy is 0.596 m, close to the specified desired height of 0.60 m. However, with the same reward used in a potential-based form, the average height of the PBRS is 0.639 m, corresponding to a change of almost 5% of the total height of the robot. While both PBRS and DRS learn comparable policies, the policy appears to be more strongly biased by the DRS terms than PBRS terms. By formulating these rewards in their potential based forms, we retain the advantages of being able to guide the policy towards desirable states while relaxing how much it is affected by the rewards.

C. Sensitivity Analysis

We perform a sensitivity analysis of both DRS terms and PBRS terms by sweeping from 0.1 to 10 times the nominal weights. As shown in Fig. 6, learning with PBRS terms is substantially more consistent across the weights and individual shaping rewards chosen. When the weights of the DRS terms are increased, the policy overfits to the specified reward and sacrifices the performance of the baseline rewards to do so. In particular, rewarding a fixed, upright orientation is particularly detrimental to the baseline rewards. This is unsurprising, given the significant banking and oscillations of the torso that naturally occur during running motions.

By placing shaping rewards in potential based form, the range of weights that can produce desirable behavior is much larger. This significantly eases the burden of iterating on sets of reward weights for an acceptable policy.

V. Conclusion and Outlook

We find that PBRS terms are beneficial for learning on high-dimensional, continuous systems such as legged robots; unlike previous studies [7], [9], [10], which have mostly focused on gridworld or low-dimensional systems, we find that the main benefit is not in accelerated convergence (which in our case is only marginal) but rather on ease of tuning. We note that RL implementations in robotics often only use rewards of the form $r(s_k)$; this type of reward can be trivially converted into PBRS form, and from our findings, we advocate using the PBRS form of rewards when possible.

While we found that PBRS terms are relatively robust to weighting, we also note that these terms are implicitly scaled through the dynamics by the control timescale Δt. In future work, we plan to more closely investigate this relationship, especially in the context of hierarchical RL. Since control timescales are a natural approach to choosing hierarchical levels (typically, higher levels in a hierarchy will reason on a longer horizon, with a larger Δt), it may be possible to automatically assign rewards for different tasks to different hierarchy levels based on their relative scaling.

Another promising avenue in the context of hierarchical RL is to use value functions as PBRS, as originally proposed by Ng, Harada, and Russell [6]. Although finding a good approximation for a value function is often daunting, in a hierarchy of world models, it may be possible to solve a cascade of problems using a hierarchy of simplified world-models, and use the obtained value function to shape the reward of each successive stage.

References

[1] Z. Xie, P. Clary, J. Dao, P. Morais, J. Hurst, and M. Panne, "Learning locomotion skills for cassie:

Iterative design and sim-to-real," in *Conference on Robot Learning (CoRL)*, 2020.

[2] A. Marco, P. Hennig, J. Bohg, S. Schaal, and S. Trimpe, "Automatic lqr tuning based on gaussian process global optimization," in *IEEE international conference on robotics and automation (ICRA)*, 2016.

[3] J. Randløv and P. Alstrøm, "Learning to drive a bicycle using reinforcement learning and shaping.," in *ICML*, 1998.

[4] M. A. Müller and K. Worthmann, "Quadratic costs do not always work in mpc," *Automatica*, 2017.

[5] M. Andrychowicz, A. Raichuk, P. Stańczyk, M. Orsini, S. Girgin, R. Marinier, L. Hussenot, M. Geist, O. Pietquin, M. Michalski, S. Gelly, and O. Bachem, "What matters in on-policy reinforcement learning? a large-scale empirical study," in *International conference on learning representations (ICLR)*, 2021.

[6] A. Y. Ng, D. Harada, and S. J. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *Proceedings of the International Conference on Machine Learning (ICML)*, 1999.

[7] A. Malysheva, D. Kudenko, and A. Shpilman, "Learning to run with potential-based reward shaping and demonstrations from video data," in *International Conference on Control, Automation, Robotics and Vision (ICARCV)*, 2018.

[8] A. Harutyunyan, S. Devlin, P. Vrancx, and A. Nowe, "Expressing arbitrary reward functions as potential-based advice," *Proceedings of the AAAI Conference on Artificial Intelligence*, 2015.

[9] S. Devlin and D. Kudenko, "Dynamic potential-based reward shaping," in *International Conference on Autonomous Agents and Multiagent Systems*, 2012.

[10] T. Westenbroek, F. Castaneda, A. Agrawal, S. Sastry, and K. Sreenath, "Lyapunov design for robust and efficient robotic reinforcement learning," in *Conference on Robot Learning (CoRL)*, 2022.

[11] 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*, 2022.

[12] A. Rai, R. Antonova, S. Song, W. Martin, H. Geyer, and C. Atkeson, "Bayesian optimization using domain knowledge on the atrias biped," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[13] J. Siekmann, Y. Godse, A. Fern, and J. Hurst, "Sim-to-real learning of all common bipedal gaits via periodic reward composition," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[14] Z. Li, X. Cheng, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath, "Reinforcement learning for robust parameterized locomotion control of bipedal robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[15] X. B. Peng, E. Coumans, T. Zhang, T.-W. E. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," in *Proceedings of Robotics: Science and Systems (RSS)*, 2020.

[16] K. Green, Y. Godse, J. Dao, R. L. Hatton, A. Fern, and J. Hurst, "Learning spring mass locomotion: Guiding policies with a reduced-order model," *IEEE Robotics and Automation Letters (RAL)*, 2021.

[17] D. Reda, H. Y. Ling, and M. van de Panne, "Learning to brachiate via simplified model imitation," in *ACM SIGGRAPH Conference Proceedings*, 2022.

[18] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *International Conference on Learning Representations*, 2016.

[19] E. Wiewiora, "Potential-based shaping and q-value initialization are equivalent," *Journal of Artificial Intelligence Research*, 2003.

[20] S. Devlin and D. Kudenko, "Theoretical considerations of potential-based reward shaping for multi-agent systems," in *International Conference on Autonomous Agents and Multiagent Systems*, 2011.

[21] 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 (CoRL)*, 2022.

[22] X. B. Peng, P. Abbeel, S. Levine, and M. van de Panne, "Deepmimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Transactions on Graphics*, 2018.

[23] M. Chignoli, D. Kim, E. Stanger-Jones, and S. Kim, "The MIT humanoid robot: Design, motion planning, and control for acrobatic behaviors," in *IEEE International Conference on Humanoid Robots*, 2020.

[24] J. Siekmann, K. Green, J. Warila, A. Fern, and J. Hurst, "Blind Bipedal Stair Traversal via Sim-to-Real Reinforcement Learning," in *Proceedings of Robotics: Science and Systems (RSS)*, 2021.

[25] H. Dai, A. Valenzuela, and R. Tedrake, "Whole-body motion planning with centroidal dynamics and full kinematics," in *IEEE International Conference on Humanoid Robots*, 2014.

[26] G. Garcia, R. Griffin, and J. Pratt, "MPC-based locomotion control of bipedal robots with line-feet contact using centroidal dynamics.," in *IEEE International Conference on Humanoid Robots*, 2021.

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

[28] M. Grzes and D. Kudenko, "Theoretical and empirical analysis of reward shaping in reinforcement learning," in *International Conference on Machine Learning and Applications*, 2009.