Sim2Biped: Enhancing Mobility of a Wheeled Biped via Sim-to-Real Transfer of an RL Controller

Gabrael Levine*
Dept of Computer Science
gabrael@cs.stanford.edu

Selena Sun*
Dept of Computer Science
selenas@cs.stanford.edu

Abstract

Legged robots have been shown to achieve remarkable maneuverability across complex terrains. Recent work in quadrupeds has demonstrated capability in navigating rocky terrain and rapidly adapting its gait to new ground physics. Despite such progress, research focused on wheel-legged locomotion, particularly for wheeled bipedal systems, remains less explored, creating a gap in our understanding of the effectiveness of learning-based control for this morphology. Wheel-legged robots have the advantage of having simpler, more energy-efficient locomotion on smooth terrains, while being adaptable to rough terrains.

In this work, we demonstrate sim-to-real transfer of a reinforcement learning (RL) controller on a wheeled biped (Rhea), trained using Proximal Policy Optimization (PPO) in Nvidia's Isaac Gym simulation environment. Rhea is an agile wheeled biped robot designed to enable embodied AI and human-robot interaction research, designed and built by the authors. In addition to PPO, we use curriculum learning in Isaac Gym to increase the policy's robustness and speed of convergence. The robot is first given a smooth sloped terrain, then rough sloped terrain and discrete steps. Upon sim-to-real transfer, we show that the RL controller is capable of stepping over taller ledges and traverse ramps, as compared to the linear controller. Specifically, the linear controller trips on ledges 21mm tall, whereas the RL controller only fails on ledges 49mm tall. Moreoever, the RL controller successfully scales a ramp with base angle of 15 degrees, whereas the linear controller fails on this task.

Our work shows that reinforcement learning is an effective means of designing controllers for varied terrain. We prove that PPO with curriculum learning is a viable strategy for producing a policy that works on a real-world robot.

1 Introduction

1.1 Motivation

Wheel-Legged Morphology

Agile ground robots have enabled a diverse array of new robotic applications, including inspection [1], construction, warehouse manipulation, and even public safety [2]. Ground robots are a uniquely good form factor for the above tasks, since they are capable of having long battery life, high reliability, and quick navigation.

Up until the past three years, most of legged locomotion has been focused on "leg and foot" locomotion, as seen in Boston Dynamics' Spot robot [2]. More recently, industry has seen the rise of wheeled robots, most notably with Ascento Robotics' wheeled biped [1], and some industry research on wheeled quadrupeds [3]. The motivation of adding wheels to these legged robots is to enable fast and stable movements on smooth ground, while allowing for dynamic legged adjustments in rough or complex terrain. Comparing the 1X wheeled robot (Eve) [4] to the Tesla Optimus robot, we see clear tradeoffs between the two morphologies: Eve is more stable and faster, but is constrained to moving on a flat floor. On the other hand, Optimus is slower, but is (supposedly) capable of stepping over obstacles and going up stairs. By building wheel-legged robots, practitioners aim to combine the benefits of both morphologies.

Reinforcement Learning

We chose to experiment with an RL controller. There are several advantages RL offers over methods like optimal control and imitation learning. Unlike optimal control methods that often rely on precise models of the system and the terrain, RL can learn effective locomotion strategies through trial and error. This flexibility allows the wheeled biped to adapt to varying rough terrain conditions, including unknown or dynamically changing environments. While imitation learning requires expert demonstrations, RL can discover novel and potentially superior locomotion strategies by actively exploring the space of possible actions and learning from the resulting consequences. This capacity for exploration allows the wheeled biped to discover new ways to handle rough terrains, potentially outperforming human-designed strategies. Lastly, classical control approaches often struggle to capture the complex dynamics and nonlinearities associated with rough terrain locomotion.

1.2 Prior Work

RL for Legged Locomotion

We choose to use a state of the art RL method, Proximal Policy Optimization (PPO), which clips the objective function to prevent large policy updates [5]. PPO has been shown to significantly improve the stability and sample efficiency of RL agents.

The field of RL for legged locomotion has seen major advances in recent years. Notably, the Rapid Motor Adaptation algorithm solves the problem of real-time terrain adaptation for quadrupeds, making it robust to rocky, slippery, and deformable terrain [6]. We chose not to use RMA, since we think PPO would suffice for the tasks we were aiming for. There has also been some work to learn quadruped gaits using imitation learning from animal videos [7], but since there is no real-world animal with wheels, this was not viable. Some work has also been done to increase the safety of deploying RL policy on real-world systems, notably by switching between a safe recovery policy and a learner policy [8].

Wheeled-Legged Locomotion

A nontrivial amount of wheeled biped research has been done for the Ascento robot. For example, one study attempted to use LQR for full-body control of the robot [9].

Some work has been done to better model the wheeled biped system. Xin et al. attempted MPC on a wheeled biped, modeling it as a Cart-Linear Inverted Pendulum [11]. Chen et al. used quadratic

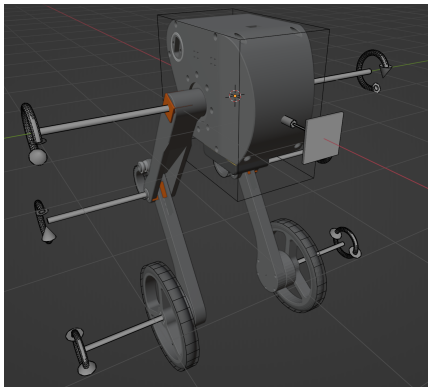
programming to devise a jumping motion path for a wheeled biped, inventing a model called wheeled-spring-loaded inverted pendulum to characterize the biped's dynamics [12].

2 Implementation

2.1 Robot

Rhea (Figure 1a) is a wheeled bipedal robot designed and built by Gabrael Levine (the author). The robot is 3D printed using carbon fiber-nylon (PA-CF) filament. The onboard compute is a Raspberry Pi 4, and the robot is powered via a DeWalt battery. The robot additionally has an RGBD sensor (via an onboard Oak-D Pro W camera) and a microphone, but we didn't use these sensing modalities for our locomotion experiments. Rhea has two legs, and each leg has 2 DoF. There is one motor at the hip, and another motor at the wheel. An approximate straight-line linkage couples the rotation of the knee to the rotation of the hip, roughly constraining the end of the leg to move only in the vertical axis.

(a) Rhea Rendering



(b) Rhea URDF

Figure 1: Rhea Rendering and URDF

2.2 Simulation: Isaac Gym

Nvidia's Isaac Gym is a robust simulation environment designed for robotic reinforcement learning research. It has fairly precise physics, which has alleviated the inaccuracies sim-to-real transfer.

First, we created Rhea's URDF (Figure 1b) by importing the CAD model into Blender. Initially, we added an artificial revolute joint at the knee, since there currently isn't a way to represent the linkage system in a URDF. However, we soon realized that the policy would output torque commands to a joint that didn't exist on the robot, rendering the policy useless. So, we removed the legs entirely, and created a constraint where the wheel would move on a vertical axis proportional to the rotation of the hip motor.

Secondly, we worked with ETH Zurich's legged_gym repository, and wrote configuration files for Rhea (modified code). The action space has dimension four, and contains the actions: [right_leg_position, right_wheel_velocity, left_leg_position, left_wheel_velocity]. The observation space has dimension 21, and contains [base_angular_velocity, gravity_vector, commanded_linear_velocity, commanded_yaw_velocity, joint_positions, joint_velocities, previous_actions].

Third, we adjusted terrain parameters to train on increasingly hard terrain (curriculum learning). The available terrains are: smooth slope, rough slope, stairs up, stairs down, and discrete. For the final policy we deployed on the robot, the terrain proportions were: 40% smooth slope, 30% rough slope, and 30% discrete steps. The curriculum learning terrain was created as follows: we first create a 8x8 grid. The terrains would be randomly distributed throughout the grid at the given proportions. The

slopes and step size of each grid is set to be proportional to the row number:

$$\text{slope} = \frac{\text{row number}}{8 \text{ rows}} * 0.4$$

$$\text{step height} = \frac{\text{row number}}{8 \text{ rows}} * 0.18 + 0.05$$

The slopes of the smooth and rough terrain therefore ranges from 0.05 to 0.4. The sizes of the discrete steps ranges from 0.0725 meters to 0.23 meters. These numbers were given as default values in the original legged_gym repository. A visualization of the terrain can be seen in Figure 2.

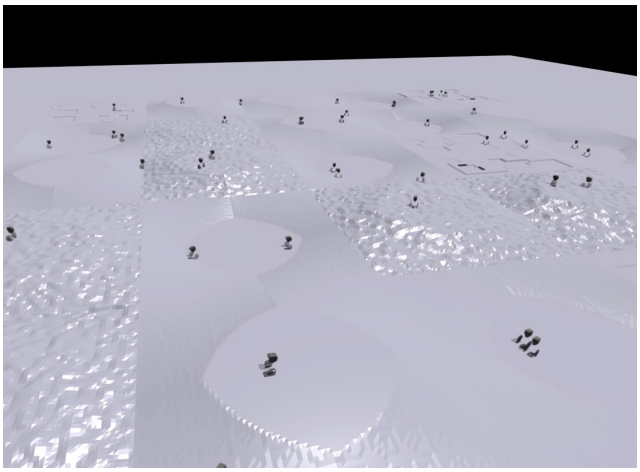


Figure 2: Curriculum Learning Terrain

Lastly, we trained the policy with PPO. The actor and critic networks both used hidden layers of sizes 512, 256, and 128, in addition to an LSTM layer of size 512. The policy was conditioned on desired forward velocity and turning velocity.

2.3 Sim-to-Real Transfer

2.3.1 System Identification

We conducted system identification to build an accurate model of the robot. We first measured the mass of each link of the robot, incorporating it into the URDF model. Then, to account for inaccuracies in our center-of-mass estimate (crucial for balancing), we measured the pitch angle at which the robot was balanced, and compensated for it in our policy deployment code.

2.3.2 Domain Randomization

In keeping with common practice for sim-to-real transfer for locomotion RL, we applied domain randomization to the robot's mass (+-0.25kg) and friction coefficients (0.5-1.5). To further encourage robustness, random velocity perturbations of up to 0.5m/s were applied to the robot every 15 seconds while the policy was training.

2.3.3 Policy Deployment

We ran the policy on the robot's onboard Raspberry Pi 4 computer. The policy inference thread receives observations from and sends actions to the robot's low-level control thread at 50hz. ROS2 messages were used for inter-process communication.

3 Results

3.1 Simulation Results

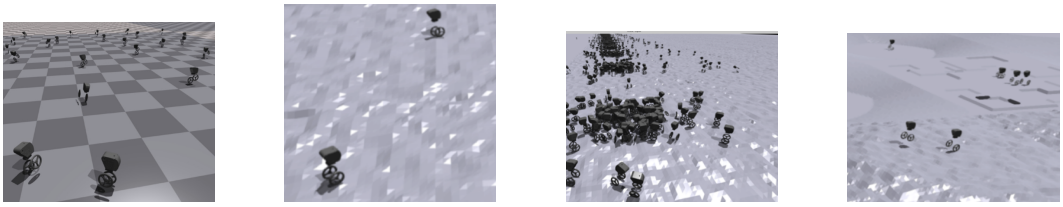
We conducted a series of experiments in Isaac Gym. We first trained Rhea to walk on flat ground, then increased terrain difficulty. For all these experiments, we domain randomized the robot's mass and friction properties.

1) Flat Ground (Figure 3a. Average reward = 9.7, mean episode length = 1021, learning iterations = 2000, number of environments = 1500): We trained Rhea in simulation without any terrain features. This was to prove basic functionality and revealed that we needed to reconfigure the URDF.

2) Rough Terrain (Figure 3b. Average reward = 28.4, mean episode length = 988, learning iterations = 500, number of environments = 8000): We noticed that on flat ground, Rhea learned to stand on one leg, which is undesirable behavior. We then trained Rhea on rough terrain, which mitigated the behavior.

3) Curriculum Learning - Sloped Rough Terrain (Figure 3c. Average reward = 19.1, mean episode length = 956, learning iterations = 500, number of environments = 8000): Curriculum learning enabled Rhea to climb steeper slopes (slope = 0.4).

4) Curriculum Learning - Rough Terrain and Small Steps (Figure 3d. Average reward = , mean episode length = 924, learning iterations = 500, number of environments = 8000): Curriculum learning enabled Rhea to step up small ledges (>70mm), as well as traverse sloped terrain.



(a) Experiment 1 (b) Experiment 2 (c) Experiment 3 (d) Experiment 4

Figure 3: Simulation Experiments

We chose one most promising model from experiment 4 to transfer to the robot (torque, position, and velocity shown in Figure 4). The model we chose showed stable locomotion in simulation. Interestingly, it seemed to have learned an oscillating motion. We hypothesize it's to stabilize itself on rough sloped terrains (the oscillating was alleviated when rough slopes were removed from the terrain). The policy also learned to make the robot lift its feet when presented with a ledge. This behavior enabled it to traverse taller ledges than the linear controller.

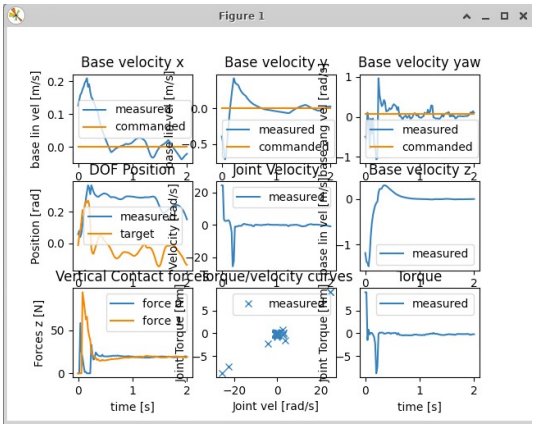


Figure 4: Simulation Results

3.2 Real World Results

We conducted a series of real world experiments on the robot, testing its ability to roll over ledges without falling, and to climb up smooth slopes.

Rolling Over Ledges

These ledges were constructed by stacking magazines (7mm thick) on top of each other for the 7mm, 14mm, and 21mm heights, and textbooks for the 37mm and 49mm heights. We tested each controller at least 3x for each ledge height, both at the same velocity (approx. 0.5m/s).

We define 'success' to be the act of traversing over the ledge at 0.5m/s and stabilizing after the traversal.

Table 1: Performance Rolling Over Obstacles

Obstacle Height	**Linear Controller (Baseline)**	**RL Controller**
0mm (flat ground)	Success	Success
7mm	Success	Success
14mm	Success	Success
21mm	Failure	Success
37mm	Failure	Success
49mm	Failure	Failure

Figure 5: Rhea Successfully Traversing a Ledge, Using the RL Controller

The RL controller performed much better than the linear controller, and was able to roll over ledges more than twice the height that the linear controller could handle. The RL controller only failed at 49mm, when the robot couldn't lift its leg over the textbook. A visualization of the RL controller successfully completing the task can be seen in Figure 5.

Climbing Slopes

We constructed ramps of the following slopes and tested each controller type 3x on each slope.

We define 'success' to be the act of getting to the top of the ramp without falling.

Table 2: Performance on Slopes

Slope	Linear Controller (Baseline)	RL Controller
0 (flat ground)	Success	Success
0.1	Success	Success
0.25	Failure	Success
0.40	Failure	Failure

Figure 6: Rhea Successfully Scaling the Ramp, Using the RL Controller

The RL controller performed better than the linear controller at steeper slopes. The RL controller caused the robot to oscillate while going up the slope, which provided more stability. At sloped 0.40, the robot fell forward. The failure mode of the linear controller was also the robot falling forward.

4 Conclusions

We have proven that the RL controller makes the robot much less likely to fall on slopes and when traversing ledges. This is due to a few behaviors the robot learned in simulation, including lifting a leg momentarily and oscillating. We have successfully demonstrated that using reinforcement learning in simulation, then performing sim-to-real transfer, is an effective way to design a controller more robust to varied terrains. To our knowledge, we've also been the first to demonstrate sim-to-real transfer of PPO on a wheeled biped (or at the very least a 3D printed wheeled biped robot).

4.1 Limitations

Our policy currently doesn't use vision input, despite the presence of an RGBD vision sensor on the robot. It can only detect the presence of obstacles through proprioception (i.e. bumping into things), which sometimes results in a loss of stability. This limits the maximum traversable obstacle height to 40mm.

4.2 Future Work

Our most immediate next step would be to perform additional reward tuning to penalize high-frequency actions. Currently, the robot sometimes kicks a leg repeatedly, which is undesirable behavior, and could be unsafe. Once we retrain the policy, we'd like to conduct real-world experiments on a wider range of terrains, such as traversing over gravel, navigating through tanbark, etc. We'd also like to experiment with giving the robot vision and depth sensor inputs.

5 Contributions and Acknowledgements

5.1 Contributions

5.1.1 Gabrael Levine

1. Created the URDF model for Rhea

2. Implemented IsaacGym training code for Rhea
3. Trained policies
4. Implemented policy deployment code for the real robot
5. Conducted ledge traversal experiments on the real robot
6. Edited the final report

5.1.2 Selena Sun

1. IsaacGym and VNC setup
2. Tuned terrain for curriculum learning
3. Trained policies
4. Conducted ledge and ramp traversal experiments on the real robot
5. Wrote and edited the final report. Conducted lit review and produced figures.

5.2 Acknowledgements

5.2.1 Rafael Rafailov (CS224R project mentor)

Suggested adding LSTMs to the policy, which ended up significantly improving sim-to-real transfer performance.

5.2.2 Wenhao Yu (Google DeepMind)

Provided advice on reward tuning for locomotion RL.

5.2.3 Zipeng Fu (SAIL)

Provided advice on system identification and debugging sim-to-real transfer.

References

[1] Hutter, M., Diethelm, R., Bachmann, S., Fankhauser, P., Gehring, C., Tsounis, V., et al. "Towards a generic solution for inspection of industrial sites." ETH Zurich 2017 11th Conference on Field and Service Robotics (FSR), 2017.

[2] Boston Dynamics. "Products." https://www.bostondynamics.com/products.

[3] Bjelonic, Mark. IEEE Robotics and Automation Letters. Preprint Version ... - Marko Bjelonic, www.markobjelonic.com/publications/files/2020$_r al_b jelonic.pdf. Accessed 13 June 2023$.

[4] 1X Technologies. "Eve." https://www.1x.tech/eve.

[5] Schulman, John, et al. "Proximal Policy Optimization Algorithms." arXiv.org, 28 Aug. 2017, arxiv.org/abs/1707.06347.

[6] Kumar, Ashish, et al. "RMA: Rapid Motor Adaptation for Legged Robots." https://ashish-kmr.github.io/rma-legged-robots, 2021.

[7] Peng, Xue Bin, and Patrick M. Wensing. "Learning Agile Robotic Locomotion Skills by Imitating Animals." https://www.semanticscholar.org/paper/Learning-Agile-Robotic-Locomotion-Skills-by-Animals-Peng-Coumans/1803722f786b901a744bc363c0ebdc51902ceceb.

[8] Yang, Tsung-Yen, et al. "Safe Reinforcement Learning for Legged Locomotion." arXiv.org, 5 March 2022. arxiv.org/abs/2203.02638.

[9] Klemm, Sven, and Michele Morra. "LQR-Assisted Whole-Body Control of a Wheeled Robot with Dynamic Constraints." https://www.semanticscholar.org/paper/LQR-Assisted-Whole-Body-Control-of-a-Wheeled-Robot-Klemm-Morra/0a273213ba54c13bcb083c88625dc68f3276fd3c.

[10] Klemm, Victor, et al. "Ascento: A Two-Wheeled Jumping robot." IEEE 2019 International Conference on Robotics and Automation (ICRA).

[11] Xin, Songyan, and Vijayakumar, Sethu. "Online Dynamic Motion Planning and Control for Wheeled Biped Robots." arXiv.org, 7 Mar 2020. arxiv.org/abs/2003.03678.

[12] Chen, Hua, et al. "Underactuated Motion Planning and Control for Jumping with Wheeled-Bipedal Robots." arXiv.org. 11 Dec. 2020. arxiv.org/pdf/2012.06156.pdf.