2024 International Conference on Advanced Robotics and Intelligent Systems (ARIS 2024)
National Taiwan University, Taipei 106319, Taiwan, August 22-24, 2024

Deep Reinforcement Learning in Mixture of Experts Control System for Blind Wheeled-Legged Quadrupedal Locomotion

William Zhang, Ke Wang

***Abstract*—The advances in wheeled-legged robots in recent years have led to their proliferation in human environments. However, these robots still face significant challenges regarding non-flat terrain. Stair climbing is particularly difficult for these robots and restricts their functionality in human facilities. We implemented a goal conditioned deep reinforcement learning algorithm to develop a position-based controller that allows Wheely, a wheeled-legged quadrupedal robot, to blindly climb stairs up to a record breaking 10 cm in height or 33% of Wheely's maximum body height. By training the reinforcement learning algorithm on a single environment each time, we create specialized policies that excel at their own specific terrain type. Through the independent training, we create a controller with a mixture of experts architecture that does not need to compromise between different environments, leading to a more optimal policy for each scenario. To maintain robustness, a selector neural network selects the policy based on past observations, allowing the robot to function independently in dynamic environments.**

I. Introduction

The creation of mobile robots have allowed robots to become increasingly involved in human life. The two primary terrestrial locomotion methods available to humans are wheeled and legged locomotion. Wheels allow for greater speed and efficiency on flat surfaces, while legged locomotion provides adaptability for rugged terrain. Therefore, combining wheeled and legged locomotion into one mechanism would improve both flat and uneven terrain traversal [1], [2], [3]. However, wheeled-legged locomotion is more dynamically complex compared to either wheeled or legged locomotion alone, resulting in greater difficulty in the development of control [4]. Thus, deep reinforcement learning is a promising method for developing ways to control such robotic systems, allowing for robust performance and adaptability [5], [6].

In this work, we studied the problem of stair climbing for wheeled-legged quadrupedal robots and explored the usage of a mixture of experts system to address negative policy generalization behavior which allows the robot to maintain expert level robustness in all terrain as opposed to a more jack of all trades policy approach. To select the appropriate policy to use, we created another neural network to determine the correct policy to use based on past observation states. In simulation, we used Wheely, an economic wheeled-legged quadrupedal robot, as the primary platform for the reinforcement learning. Our research combines deep reinforcement learning with a mixture of experts system to enhance the capability and adaptability of mobile robots in complex environments.

William Zhang is with Harvard University, School of Engineering and Applied Sciences, Allston, MA 02134, USA (phone: (610)-908-2333; e-mail: williamyzhang@seas.harvard.edu).

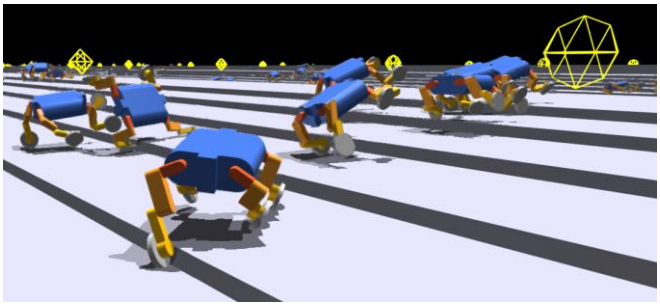

Figure 1. Wheely Training Stair Climbing

II. Literature Review

A. Model Predictive Control

Model predictive control (MPC) has proven to be a powerful tool for controlling legged and wheeled-legged robots, enabling robust locomotion and stability. MPC uses predictive models of the robot's kinematics and dynamics for control, allowing robots to adapt to various terrain and maintain stability. MPC is used in both legged and wheeled-legged robots and is shown to excel [7], [8], [9], [10]. Despite these advantages, MPC has several drawbacks: computational complexity, detailed modeling requirements, and difficult parameter tuning. MPC requires the online solving of optimization problems, something computationally taxing for high-dimensional systems like mobile robots [4], [11]. This necessitates powerful onboard computing capabilities which is costly and can lead to sluggish response times. Another issue is MCP's reliance on accurate models of the robot and environment. Inaccurate models or unmodeled dynamics can significantly degrade MPC performance, leading to suboptimal control [11]. Furthermore, tuning MPC parameters, such as prediction horizon and cost function weights, requires significant expertise and trial-and-error [11].

B. Reinforcement Learning

In contrast to MCP, reinforcement learning has emerged to quickly and simply create robust locomotion policies for legged robots. This approach has notably been implemented on ANYmal, Unitree A1, and Cassie, with Rudin et al. creating a novel simulation method that allows for the simultaneous simulation of thousands of robots in parallel, resulting in robust policies within 20 minutes [1], [12], [13], [14]. Another work involving Cassie and reinforcement learning shows RL's ability to create robust and stable locomotion behaviors [15].

Ke Wang, is with Dyson Technology LTD, Malmesbury SN160RP UK.

Moreover, hierarchical RL frameworks have been utilized to integrate high-level navigation planning with low-level locomotion control by Lee et al. [5]. Their work on autonomous navigation for wheeled-legged robots combines model-free RL techniques with privileged learning, enabling smooth transitions between walking and driving modes [5]. Reinforcement learning has also allowed legged robots to gain advanced locomotion skills such as those necessary to conquer parkour challenges involving jumping, climbing, and crouching [16], [17], [18]. Furthermore, reinforcement learning provides great freedom in locomotion behavior, resulting in more complex and creative locomotion methods [18]. Another example of reinforcement learning being implemented on legged robots is the work on MIT Mini Cheetah by Margolis et al., allowing the robot to run up to 3.9m/s [6], [19].

III. Methodology

In our setup, there are three key components, the PPO algorithm, the supervised learning algorithm, and Isaac gym. An overview of Wheely's training is seen in Figure 2, 3 and 5 which shows the structure of RL training, deployment, and terrain selector training respectively. Training consists of two parts as locomotion and terrain selection are trained separately while both locomotion and terrain selection are used in tandem during deployment.

A. Reinforcement Learning Simulator

We chose to use Isaac Gym as our physics simulator because of its RL focused design. Isaac Gym boasts an incredibly RL friendly GPU based parallelism capabilities that allow for rapid training [20]. Compared to other simulators such as MuJoCo or Pybullet, the speed afforded by Isaac Gym's architecture enables much faster training and iteration [21], [22].

B. Learning Algorithm

We used proximal policy optimization (PPO) to train our locomotion policies, specifically the open-source PPO by the ETH Robotics Systems Lab [12]. For training, we used a single desktop workstation equipped with a Nvidia 4060 GPU and 16 GB of memory.

C. Terrain

Our training environment consists of 10m x 10m squares organized in a 20x10 grid of individual environments. While we only used one environment type during training, we still used the 20x10 grid format for easier monitoring as anything smaller would quickly become crowded and difficult to inspect. Each row of the grid has the same difficulty level with the difficulty increasing as the row number increases. We tested four types of environments: stairs, smooth slopes, rough slopes, and flat planes. Each environment had two 1x10 m sections of flat terrain for the robot to spawn and end on. Between the start and end areas are continuous specialized terrain: stairs, smooth slopes, and rough slopes. While the robot could begin anywhere on the 10 meter wide horizontal axis in the starting area, to promote robustness in turning and lateral locomotion, we set the end point to be a circle 0.2 m in diameter. This end point is always centered at the midpoint of the horizontal axis of the environment. The robot always spawned facing perpendicular to the horizontal axis with its head pointing toward the opposite side of the environment to promote turning behavior.

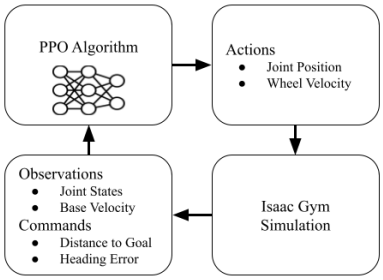

Figure 2. Reinforcement Learning Training Process

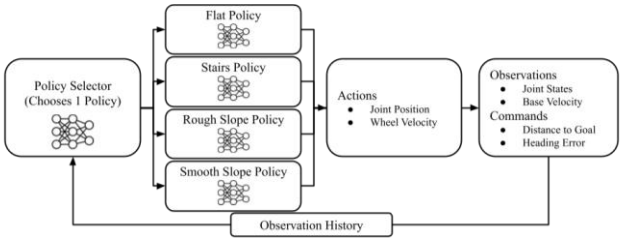

Figure 3. Deployment Structure

1) Stairs: The stair environments had two versions, ascending and descending. The stairs were structurally identical with the only difference being vertical displacement being reversed. Each step was 1m in length.

2) Smooth Slopes: Similar to the stair environments, the slopes also had ascending and descending variants. Each slope's height level was changed linearly across the length of the environment, excluding the spawn and end sections. The angle of the incline ranged from 0° to 14.5° depending on the difficulty.

3) Rough Slopes: The rough slope was made in the same format as the smooth slope, except the height of the slope was given an extra randomized height a ± 5cm variation.

4) Flat Plane: The flat plane was of the same dimensions as the other environments and had the same task formulation.

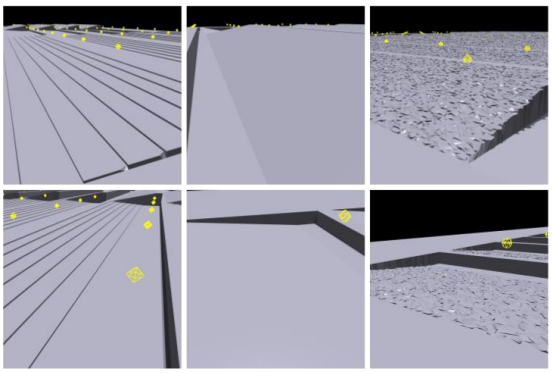

Figure 4. *Terrain Configurations*
Top Row: Stairs, Smooth Slope, Rough Slope (All Ascending)
Bottom Row: Stairs, Smooth Slope, Rough Slope (All Descending)
The target point is marked with a yellow wireframe sphere

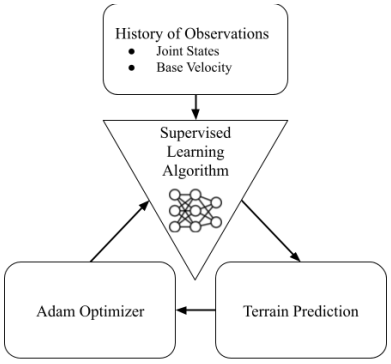

Figure 5. Selection Training Process

D. Terrain Selection

The RL policy is selected using a selector network trained by a supervised learning algorithm that takes in previous observation states and makes a prediction using them about the terrain type. The terrain selection takes in the previous 5 seconds of robot state data to predict its terrain type. Based on this prediction, the RL policy corresponding to the terrain is selected and run. The algorithm learns using the Adam optimizer [23].

E. Task Formulation

We modeled our RL task off of Rudin et al.'s position based task formulation that allowed for greater complexity in behavior [18]. Each episode lasts 60 seconds with the primary objective being to move the robot from the start area to the end point.

1) Observations and Commands:

We used a completely blind observation set with only base inertial information and joint states given. The algorithm is also given two commands, a boolean value to indicate whether the robot has reached the target location and a heading error value between the robot's heading and the target. No other information is provided to the algorithm to model real world deployment.

2) Actions:

The RL algorithm outputs a vector of joint position and velocity targets. In simulation, these values are then converted by a PID controller to torque control. Joint positions are given for leg joints while velocity targets are provided for the wheels.

3) Rewards:

We modified and added to the rewards given in the framework by Rudin et al. [12]. Most notably, we added the proximity, position, and heading rewards. Proximity is the distance between the robot base and the target point. This encourages the robot to get as close to the point as possible even if it is not able fully arrive at the target. Position reward is a positive reward for when the robot is able to arrive within 25cm of the end point. This gives a flat reward determined by the position reward coefficient. Similarly, if the robot is able to trigger the success termination (distance between robot and goal < 0.1), any negative termination rewards related to episode length are canceled and an extra reward is given.

TABLE I. REWARD DEFINITION

Name	Formula	Coeff.
Proximity	$-\sqrt{(x_{robot}-x_{goal})^2-(y_{robot}-y_{goal})^2}$	0.01
Position	If $-\sqrt{(x_{robot}-x_{goal})^2-(y_{robot}-y_{goal})^2}<0.25$ Reward +1	10
Heading	$-\lvert\theta_{robot\ to\ world}-\theta_{robot\ to\ goal}\rvert$	1
Positive End	If $-\sqrt{(x_{robot}-x_{goal})^2-(y_{robot}-y_{goal})^2}<0.1$ Termination Penalty = 0 Reward +1	10

4) Curriculum:

We trained using a set of 2048 robots, equally distributed among the first 8 difficulties. This is largely inspired by the game-like curriculum by Rudin et al. which we modified to be based on position instead of speed [12]. If the robot successfully reached the end point, it would be promoted to the next level in difficulty, and if the robot failed to reach the end point, it would stay at the difficulty level or be demoted if it has not traveled more than half of the distance to the end point. After completing the highest level, the robot is returned to a random lower level to prevent forgetting.

IV. EXPERIMENTS

After training, Wheely was tested on each terrain for successful traversal, energy use, and speed.

A. Upper Limits

Using the RL policy, Wheely is able to climb stairs that are 10 cm tall or 33% of its maximum body height while being completely autonomous. Compared to the previous open loop, bezier curve based controller, the RL policy doubled the maximum stair height that Wheely is able to surmount and drastically increased its reliability [3] [24]. Wheely is now able to traverse slopes of up to 30° incline, a novel skill developed using deep RL.

TABLE II. TERRAIN DIFFICULTY

Difficulty Level	Stairs	Smooth slope	Rough Slope
	Step Height	Slope Angle	Slope Angle Randomized Height: ±5cm Variation
1	1 cm	0°	0°
2	2 cm	2.87°	2.87°
3	3 cm	4.30°	4.30°
4	4 cm	5.74°	5.74°
5	5 cm	7.18°	7.18°
6	6 cm	8.63°	8.63°
7	7 cm	10.1°	10.1°
8	8 cm	11.5°	11.5°
9	9 cm	13.00°	13.00°
10	10 cm	14.5°	14.5°

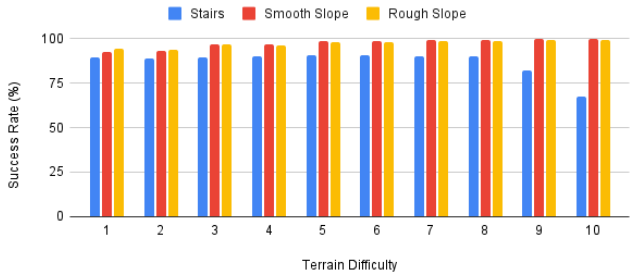

Figure 6. *Success Rate by Terrain Difficulty*

B. Success Rates

Across all terrain types and difficulty, success rate, the percent of robot reaching the goal before the time limit, generally remained greater than or near 90%. On the upper end of Wheely's capabilities, as seen in Fig. 6, Wheely was able to surmount 10cm tall steps with 67% success with success rate climbing to 90% at 8cm tall steps. Compared to the previous maximum step height of 5cm, achieved with an open loop, bezier curve based control system, the reinforcement learning doubled the maximum stair height and greatly enhanced the reliability of Wheely [3]. Smooth and rough slope success rates remained high until 30° of incline where it dropped significantly to near zero as the robot could no longer grip the surface.

C. Energy Consumption

While climbing 10cm high stairs, Wheely exerted an average 84.99 N-m of torque. The whole episode lasted 34 seconds. Over the course of the experiment approximately 2.5 Watts of energy was used by Wheely to reach the goal. Other terrain had similar torque usage with smooth slope climbing averaging 89.37 N-m and rough slope climbing averaging 86.45 N-m.

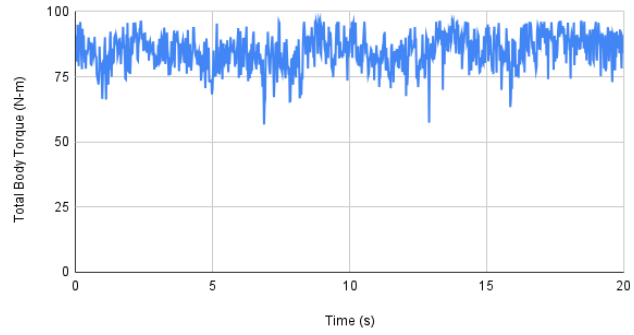

Figure 7. *Torque Usage Climbing 10cm High Stairs*

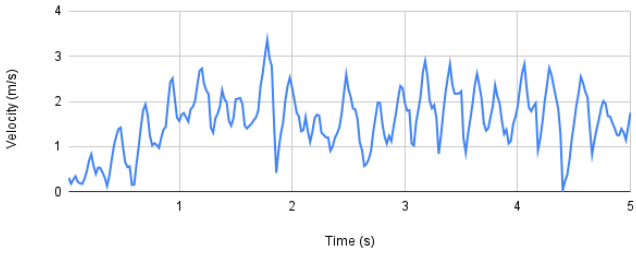

Figure 8. *Velocity Climbing 10cm Stairs*

D. Speed

Wheely's speed on the 10cm steps, as shown in Fig. 8, reflects a unique bounding behavior in which Wheely jumps on its hind legs to "bound" forward. The spikes in velocity are the robot using its legs to jump forward, clearing the stairs. This unconventional locomotion behavior most likely arose from the position-based task formulation which allows for more complex and creative behavior [18]. In testing, Wheely achieved a record maximum velocity of 3.37 m/s with an average velocity of 1.57 m/s, a significant increase from the previous maximum average velocity of only 0.093 m/s [3]. Compared to the pure legged or pure wheeled locomotion of the previous control system, the hybrid RL based control system resulted in a robot 16 times faster than before with no changes to hardware [3].

V. Conclusions

In this work, we developed a new approach for deep reinforcement learning to minimize the need to compromise policies between environments, resulting in a more optimal policy for individual environments. To maintain robustness, we created a mixture of experts control system that selects the specialized policy that best fits the environment. We implemented this to improve stair climbing and general locomotion in non-flat terrain in the blind, wheeled-legged quadrupedal robot Wheely. We achieved a significant milestone of enabling Wheely to climb stairs up to 10 cm in height, 33% of its maximum body height. This accomplishment demonstrates a substantial improvement over previous control systems, which only managed a maximum stair height of 5 cm [3].

A. Limitations and Future Work

While successful in simulation, our RL method has not been tested in the real world which may reveal significant sim-to-real gaps. Furthermore, simulations can never perfectly model the real world; thus, specialized environments trained for within simulation may be at greater risk of not translating perfectly into real-world applications. Future efforts to validate real-world efficacy of this method should be considered.

Acknowledgment

We extend our sincere gratitude to Professor Heng Yang at the Harvard School of Engineering and Applied Sciences for his invaluable insights and advice throughout this research.

References

[1] M. Hutter et al., "ANYmal - a highly mobile and dynamic quadrupedal robot," 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea (South), 2016, pp. 38-44, doi: 10.1109/IROS.2016.7758092.

[2] V. Klemm et al., "Ascento: A Two-Wheeled Jumping Robot," 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 2019, pp. 7515-7521, doi: 10.1109/ICRA.2019.8793792.

[3] W. Zhang and K. Wang, "Wheely: An Accessible Platform for Experimenting with Wheeled-Legged Quadrupedal Robots," 2023 3rd

International Conference on Computer, Control and Robotics (ICCCR), Shanghai, China, 2023, pp. 220-224, doi: 10.1109/ICCCR56747.2023.10193993.

[4] S. Chamorro, V. Klemm, M. de la Iglesia Valls, C. Pal, and R. Siegwart, "Reinforcement Learning for Blind Stair Climbing with Legged and Wheeled-Legged Robots," ArXiv, vol. abs/2402.06143, 2024. [Online]. Available: https://arxiv.org/abs/2402.06143

[5] J. Lee, et al., "Learning robust autonomous navigation and locomotion for wheeled-legged robots," Science Robotics, vol. 9, p. eadi9641, 2024. doi: 10.1126/scirobotics.adi9641. Available: https://arxiv.org/abs/2405.01792

[6] G. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid Locomotion via Reinforcement Learning," 2024 The International Journal of Robotics Research, 43(4):572-587. doi: 10.1177/02783649231224053. Available: https://arxiv.org/abs/2205.02824

[7] M. Gaertner, M. Bjelonic, F. Farshidian and M. Hutter, "Collision-Free MPC for Legged Robots in Static and Dynamic Scenes," 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 2021, pp. 8266-8272, doi: 10.1109/ICRA48506.2021.9561326. Available: https://arxiv.org/abs/2103.13987

[8] Y. de Viragh, M. Bjelonic, C. D. Bellicoso, F. Jenelten and M. Hutter, "Trajectory Optimization for Wheeled-Legged Quadrupedal Robots Using Linearized ZMP Constraints," in IEEE Robotics and Automation Letters, vol. 4, no. 2, pp. 1633-1640, April 2019, doi: 10.1109/LRA.2019.2896721.

[9] M. Bjelonic, P. K. Sankar, C. D. Bellicoso, H. Vallery and M. Hutter, "Rolling in the Deep – Hybrid Locomotion for Wheeled-Legged Robots Using Online Trajectory Optimization," in IEEE Robotics and Automation Letters, vol. 5, no. 2, pp. 3626-3633, April 2020, doi: 10.1109/LRA.2020.2979661. Available: https://arxiv.org/abs/1909.07193

[10] M. Bjelonic, R. Grandia, O. Harley, C. Galliard, S. Zimmermann and M. Hutter, "Whole-Body MPC and Online Gait Sequence Generation for Wheeled-Legged Robots," 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Prague, Czech Republic, 2021, pp. 8388-8395, doi: 10.1109/IROS51168.2021.9636371. Available: https://arxiv.org/abs/2010.06322

[11] P. M. Wensing, M. Posa, Y. Hu, A. Escande, N. Mansard, and A. Del Prete, "Optimization-based control for dynamic legged robots," arXiv preprint arXiv:2211.11644, 2022. Available: https://arxiv.org/abs/2211.11644

[12] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to Walk in Minutes Using Massively Parallel Deep Reinforcement Learning," in Proc. 5th Conf. Robot Learn., A. Faust, D. Hsu, and G. Neumann, Eds., vol. 164, Proc. Mach. Learn. Res., PMLR, 2022, pp. 91-100. Available: https://proceedings.mlr.press/v164/rudin22a.html.

[13] Unitree Robotics, A1, 2022. [Online]. Available: https://www.unitree.com/a1/.

[14] "Agility Robotics Introduces Cassie: A Dynamic and Talented Robot Delivery Ostrich," IEEE Spectrum, [Online]. Available: https://spectrum.ieee.org/agility-robotics-introduces-cassie-a-dynamic-and-talented-robot-delivery-ostrich.

[15] Z. Xie, P. Clary, J. Dao, P. Morais, J. Hurst, and M. van de Panne, "Iterative reinforcement learning based design of dynamic locomotion skills for Cassie," arXiv preprint arXiv:1903.09537, 2019. Available: https://arxiv.org/abs/1903.09537

[16] X. Chen, A. Ghadirzadeh, J. Folkesson, M. Björkman and P. Jensfelt, "Deep Reinforcement Learning to Acquire Navigation Skills for Wheel-Legged Robots in Complex Environments," 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 2018, pp. 3110-3116, doi: 10.1109/IROS.2018.8593702. Available: https://arxiv.org/abs/1804.10500

[17] D. Hoeller, N. Rudin, D. Sako, and M. Hutter, "ANYmal parkour: Learning agile navigation for quadrupedal robots," Sci. Robot., vol. 9, no. 88, p. eadi7566, 2024. doi: 10.1126/scirobotics.adi7566. Available: https://arxiv.org/abs/2306.14874

[18] N. Rudin, D. Hoeller, M. Bjelonic and M. Hutter, "Advanced Skills by Learning Locomotion and Local Navigation End-to-End," 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Kyoto, Japan, 2022, pp. 2497-2503, doi: 10.1109/IROS47612.2022.9981198. Available: https://arxiv.org/abs/2209.12827

[19] B. Katz, J. D. Carlo, and S. Kim, "Mini cheetah: A platform for pushing the limits of dynamic quadruped control," in 2019 International Conference on Robotics and Automation (ICRA), 2019, pp. 6295–6301. DOI: 10.1109/ICRA.2019.8793865.

[20] 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. Available: https://arxiv.org/abs/2108.10470

[21] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in 2012 IEEE/RSJ international conference on intelligent robots and systems. IEEE, 2012, pp. 5026–5033

[22] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation in robotics, games and machine learning," 2023. Available: http://www.pybullet.org

[23] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," CoRR, vol. abs/1412.6980, 2014. Available: https://arxiv.org/abs/1412.6980

[24] M. Rahme, I. Abraham, M. Elwin, and T. Murphey, "Spotminimini: Pybullet gym environment for gait modulation with bezier curves," 2023. Available: https://github.com/moribots/spot_mini_mini