Reinforcement Learning for Robust and Adaptive Quadruped Locomotion

Ashutosh Gupta

***Abstract*— Legged robots have the potential to operate in complex and cluttered environments that are difficult for wheeled systems. Achieving reliable locomotion on uneven terrain requires whole-body control capable of regulating both motion and posture while maintaining stability under uncertainty. Traditional optimization-based controllers, such as Model Predictive Control (MPC), have demonstrated effective performance but rely on simplified dynamics and are sensitive to model mismatch. In this work, we develop a reinforcement learning-based controller for blind whole-body quadruped locomotion, trained using only proprioceptive sensing. The learned policies enable a quadruped to traverse challenging terrain, including stairs, slopes, steps, and low-friction surfaces. The policies are trained in simulation using robot observations and a task-driven reward function on a variety of generated terrains. We deploy the controller on a Unitree Go2 platform and demonstrate robust locomotion behaviors on hardware. We also validate the approach in simulation on a Spot quadruped. These results highlight the potential of reinforcement learning to enable scalable and robust whole-body locomotion in real-world environments. Code and demo videos can be found here - https://github.com/osudrl/quadruped_locomotion.**

***Index Terms*— Reinforcement learning, Legged robots, Whole-body locomotion**

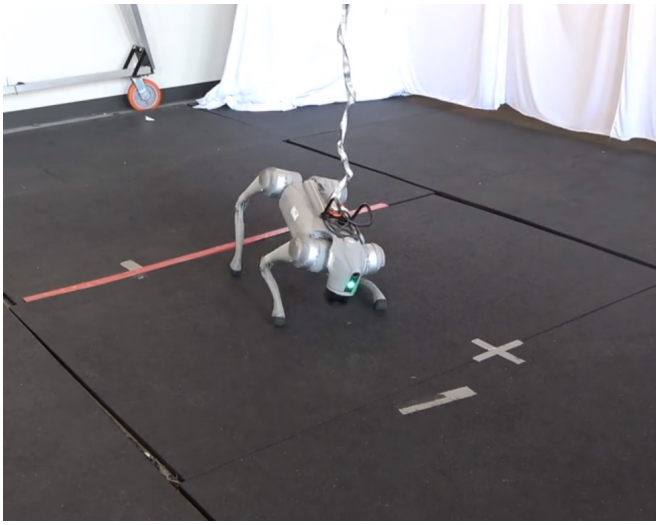


Fig. 1: Deployment of the learned whole-body controller on a Unitree Go2 quadruped. The policy changes the robot's height and body orientation based on user joystick commands while maintaining stable locomotion.

I. Introduction

Robots are increasingly being deployed for real-world applications in various fields such as construction [1], agriculture [2], disaster response [3], and payload transport [4]. Legged robots offer a distinct advantage in these domains, as they are designed to walk, crawl, and jump across unstructured and challenging terrains that are generally inaccessible to wheeled robots [5]. Despite these strengths, legged locomotion remains a difficult control problem. The system dynamics are nonlinear and demand controllers that balance precision with robustness [6].

Much of the existing research in legged locomotion has focused on developing optimization-based control strategies. Among these, Model Predictive Control (MPC) has emerged as a leading approach due to its stable, safe, and robust locomotion capabilities [7]. Recent works have demonstrated dynamic and adaptive whole-body control for quadrupeds with different modes that can effectively handle both system and environmental uncertainties [8], [9], [10]. Although MPC has made significant progress in legged locomotion, it still faces key limitations, such as sensitivity to model mismatch, slow online optimization, and difficulty in managing high-dimensional models. Moreover, MPC-based controllers rely on numerous manual design choices, including foot trajectory planning, cost function design, and prediction horizon tuning, which require significant domain and system knowledge. To address these challenges, MPC-based approaches often use simplified models and simpler cost functions to design a viable controller, which limits their applicability to relatively simpler and less dynamic environments.

Reinforcement Learning (RL) offers a compelling alternative. RL approaches have demonstrated impressive legged locomotion capabilities across diverse, complex, and natural environments [5], [11], [12], [13]. These methods do not require an explicit system model. Instead, they learn control policies in simulation by interacting with the environment and optimizing a reward signal that captures the task objectives [14]. The resulting policies can directly map proprioceptive observations to joint actions in a closed-loop manner [15], [16]. However, RL methods also come with their own set of challenges, such as the simulation-to-reality (sim-to-real) gap, high data sampling requirements during training, and limited safety guarantees in the learned policies. Despite these challenges, recent progress has shown promising results in scaling RL for dynamic locomotion tasks [17].

Robust operation on uneven terrain requires whole-body control that can regulate base velocity, height, and orientation. In this work, we develop a blind RL-based controller for quadruped whole-body locomotion, relying exclusively on proprioceptive sensing without any exteroceptive perception of the environment. Our controller enables traversal of complex terrain, including stairs, slopes, steps, and low-

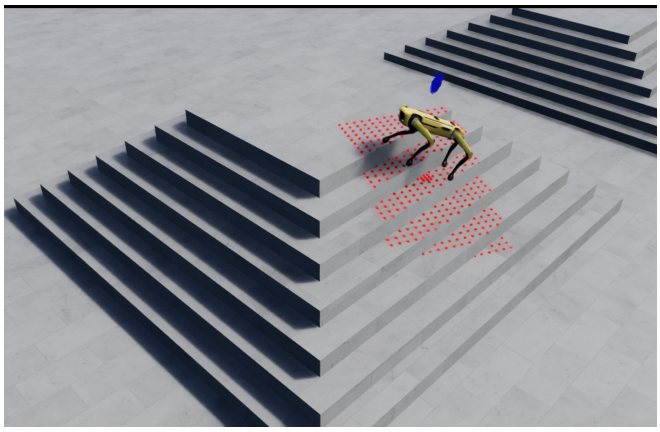


Fig. 2: Simulation of the learned whole-body controller on a Spot quadruped traversing a staircase in Isaac Sim [18]. The robot successfully adjusts its posture and foot placement using only proprioceptive feedback.

friction surfaces. We present our results in simulation and our progress toward reliable deployment on Unitree Go2 hardware (Figure 1), with similar progress on a Spot robot in simulation (Figure 2).

The remainder of this paper is structured as follows. Section II provides an overview of the key reinforcement learning concepts relevant to this work and prior research on legged locomotion. Section III describes the proposed approach for developing a whole-body RL locomotion controller. Section IV presents the experimental setup, along with quantitative and qualitative evaluations of the learned policies. Finally, Section V summarizes the findings and concludes the paper.

II. Background

A. Reinforcement Learning

Reinforcement learning (RL) is a learning framework where agents learn through trial and error when interacting with the environment. At each step, the autonomous agent takes an action based on the current state to transition to the next state and receives a reward signal. The RL objective is to learn a behavior that maximizes the expected cumulative reward, defined as the value function [19]. An RL problem is formulated in the mathematical framework of MDPs (Markov decision process) as a tuple of (S, A, R, P, γ) consisting of the state space, action space, reward function, transition probability, and the discount factor [20]. Solving an MDP results in a policy π that maps a state $s \in S$ to an action $a \in A$.

Deep reinforcement learning (DRL) integrates RL with deep neural networks to approximate policies and value functions [21], [22]. DRL enables learning in high-dimensional, nonlinear systems without requiring accurate analytical models of robot dynamics [23]. Furthermore, the ability to simulate many parallel environments on GPUs allows efficient data collection and rapid policy improvement [18]. However, DRL has little to no formal guarantees for finding an optimal policy, suffers from the sim-to-real gap, requires a training environment, and often requires millions of interactions before a useful policy is learned [24].

For continuous control problems such as quadruped locomotion, actor-critic methods have become a widely used class of RL algorithms [14]. Proximal Policy Optimization (PPO) [25] is one of the most common algorithms in this domain due to its simplicity and reliability. PPO learns an actor, which represents the control policy and is updated to increase the likelihood of actions that lead to higher advantages, and the critic, which estimates the state value to provide a learning signal for the actor. PPO optimizes a clipped surrogate objective that constrains the policy update step. This clipping mechanism helps avoid large, destabilizing updates and results in more stable training performance. Due to these properties, PPO has become a standard approach for training locomotion controllers [17].

Building on this foundation, the next section reviews relevant RL-based approaches for legged locomotion and whole-body control.

B. Learning for Quadruped Locomotion

Much of the recent progress in quadruped locomotion has been driven by DRL approaches. Several works target high-speed or agile motion, showing that model-free policies can produce running and sprinting behaviors on natural terrain [15]. These controllers are typically trained in simulation using domain randomization and then transferred to hardware, yielding robust base-velocity tracking across uneven, sloped, or otherwise challenging ground [17]. Reward design has also been studied in depth. For example, energy and style-aware objectives have been shown to induce natural gait patterns and desirable footfall timings without explicitly prescribing contact schedules [26]. Early learning-based controllers often relied on a parameterized gait space where the policy commanded only a limited set of motion variables, such as step frequency or foot trajectory. While these controllers could produce stable periodic behavior, their fixed structure limited performance in more complex and cluttered environments [12]. More recent work has therefore shifted to learning joint-level actions directly, which enables richer and more adaptive behaviors [14].

Another active research direction has focused on addressing the gap between simulation and real-world deployment. Some approaches leverage rapid online adaptation to changing dynamics, enabling policies trained in simulation to transfer effectively to different terrains and hardware conditions [16]. Other research specifically targets locomotion over complex terrain using only proprioceptive sensing, demonstrating that blind control can remain robust even when the environment is unpredictable or unstructured [11]. In contrast, additional methods incorporate exteroceptive feedback such as stereo vision or elevation mapping to anticipate upcoming terrain and improve navigation performance across diverse outdoor environments [5], [27]. Recent developments have also extended locomotion beyond simple velocity tracking. Some approaches incorporate whole-body

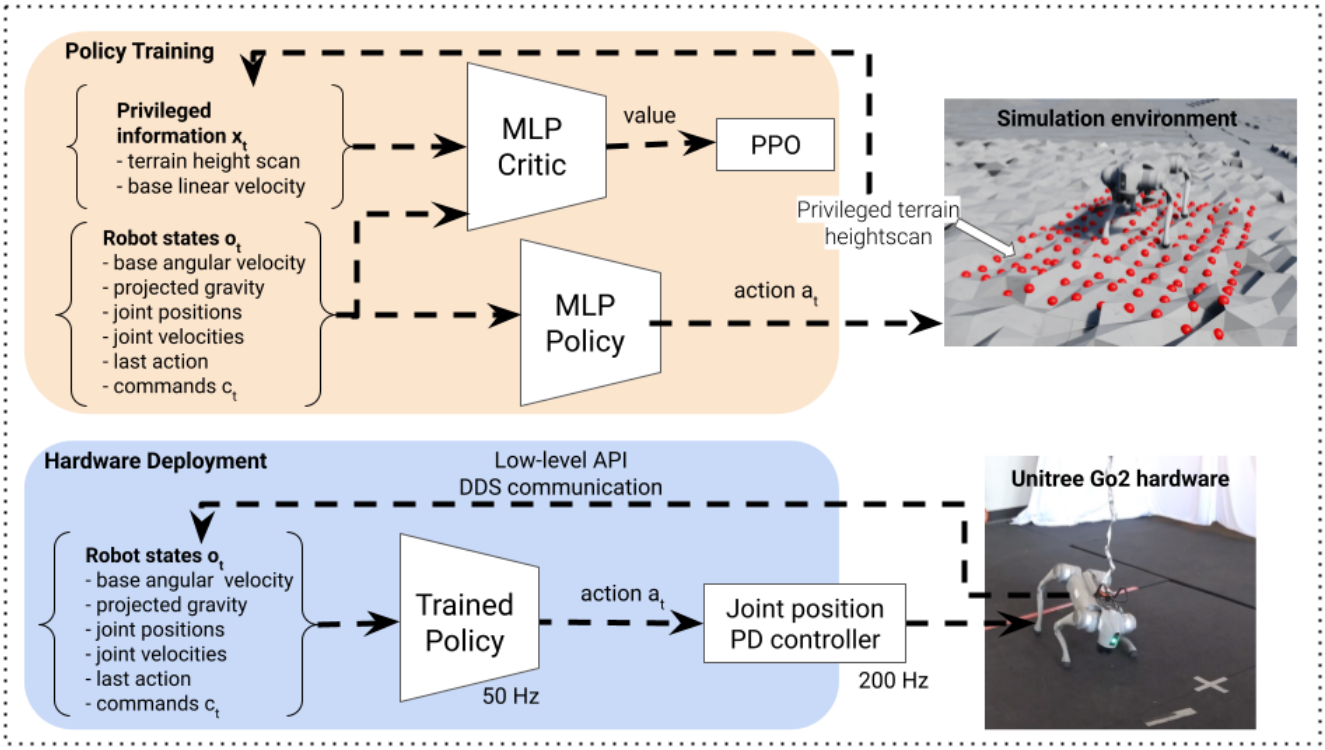


Fig. 3: Overview of our learning and deployment pipeline for blind whole-body quadruped locomotion. During training (top), the policy is optimized in simulation using PPO within an asymmetric actor–critic setup, where the critic receives privileged terrain information not available to the policy. The actor observes only proprioceptive state inputs and outputs joint position setpoints. During hardware deployment (bottom), the trained policy runs onboard the Unitree Go2 robot at 50 Hz and communicates with the low-level API via DDS to execute locomotion behaviors in real time.

coordination to navigate confined 3D spaces and tight terrain features [12], while others explore more dynamic behaviors such as parkour motions involving jumping and rapid configuration changes [13].

Learning-based quadruped control has further expanded toward loco-manipulation and contact-rich behaviors. Such approaches combine navigation and object interaction, enabling quadrupeds to push, manipulate, or carry items using their bodies or legs [28], [29]. These advances illustrate the adaptability and progression of learning-based methods for solving complex control challenges in nonlinear, high-dimensional legged systems. The demonstrated success of these methods motivates our use of deep reinforcement learning for whole-body quadruped locomotion in this work.

III. Methods

This section presents our formulation and implementation details for the whole-body quadruped locomotion controller. We describe the observation and action spaces, reward structure, environment setup, training curriculum, policy architecture, and learning framework used in this work. Figure 3 shows an overview of our method.

A. Observation and Action space

The policy uses only proprioceptive sensing. The observation space includes the robot's base angular velocity (ω_b), the projected gravity vector (g_b) to capture the robot's body orientation independent of heading. Joint positions (q) and velocities ($\dot{q}$) from all 12 actuated joints, with three joints per leg. During training, zero-mean noise is added to these observations to improve robustness to real sensor noise. Lastly, the previous action (a_{t-1}) to help stabilize control across consecutive steps. So the robot observation vector is defined as $o_t = (\omega_b, g_b, q, \dot{q}, a_{t-1}, c_t)$.

Where c_t is a command vector of seven elements $c_t = (v_x, v_y, \omega_z, b_s, h, r, p)$. The first three terms represent the desired base linear velocities in the x and y directions and the desired yaw rate. The term b_s is a stand mode bit that switches between walking and standing behavior, reducing undesirable stepping when near-zero commands are given. The remaining terms (h, r, p) specify target base height, roll, and pitch, enabling explicit control of whole-body posture.

To aid stable value learning, we adopt an asymmetric actor-critic formulation. Along with the policy's observation space, the critic also has access to privileged information. For example, the Go2 robot does not directly measure base linear velocity (v_b) on hardware, so this signal is excluded from the actor but included for the critic. This allows the policy to remain hardware-feasible while the critic benefits from more informative inputs for accurate value estimation. For terrain tasks, we further provide the critic with a clean height scan (s_t) of the surrounding terrain using a grid of

simulated raycaster sensors around the robot's base. The actor does not receive this information, maintaining a blind control consistent with our hardware deployment constraints.

The action space consists of joint position targets for all 12 joints (a_t). The policy outputs joint targets relative to nominal positions that are scaled and added to the nominal configuration. So the final joint target is computed as $q_{target} = scale * a_t + q_{nominal}$. These target joint positions are then tracked using a PD controller that generates torques applied in simulation.

B. Rewards

The design of the reward function plays a crucial role in shaping the learned behavior of the RL policy. We construct a dense reward composed of multiple terms grouped into four categories: task rewards, style rewards, auxiliary rewards, and regularization penalties. Table V explains all the reward terms used and the weights in the training.

1) Task rewards: Task rewards directly measure how well the policy achieves the primary control objectives. These include linear and angular velocity tracking terms that penalize deviation from the commanded base motion in the x, y, and yaw directions. In addition, height tracking and body attitude tracking rewards encourage the robot to maintain the commanded base height, roll, and pitch.

2) Style rewards: Style rewards are used to shape the qualitative characteristics of locomotion without explicitly contributing to the main task objective. For quadruped control, these rewards encourage smooth and coordinated gait patterns. A trot-based gait reward promotes synchronous motion of diagonal leg pairs when the velocity command is nonzero. A foot air-time reward encourages sufficient swing duration for each leg to support base progression, while an air-time variance penalty discourages asymmetric leg behaviors. Finally, a feet-contact reward encourages stable standing behavior when near-zero velocity is commanded by rewarding a larger number of feet in ground contact.

3) Auxiliary rewards: Auxiliary rewards support additional desirable characteristics that improve stability and execution quality. A foot clearance reward encourages each leg to lift sufficiently high during swing, improving obstacle avoidance on rough terrain. A slip penalty reduces undesirable feet sliding behavior. We also include a joint-position deviation penalty to discourage large deviations from a nominal configuration, and a base-motion penalty to penalize high base velocities in undesirable directions for stability.

4) Regularization penalties: Regularization terms are included to ensure smooth and physically feasible behavior. These penalties act on joint torques, velocities, and accelerations to prevent excessive forces and rapid motor actuation. We further penalize large changes in consecutive actions to reduce jitter and encourage smoother control outputs over time.

C. Environment setup

We train our policies in the Isaac simulator [18] using 4096 parallel environments. The control policy operates at 50 Hz, while the simulator and low-level PD controller run at 200 Hz. Each rollout is sampled for a duration of 20 seconds. During training, an episode is terminated if the robot enters a failure state. A failure state is defined as contact between the robot base and the ground, or excessive body tilt exceeding 150° in either roll or pitch, as illustrated in Figure 4.

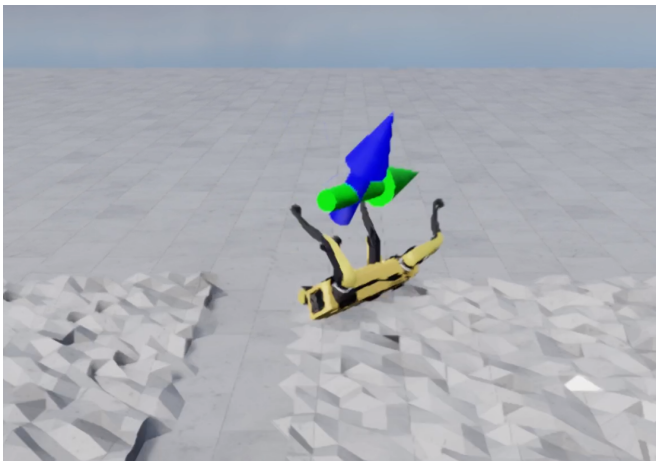


Fig. 4: Example of a failure termination state in simulation, where the robot loses balance and the base contacts the ground after rolling over uneven terrain.

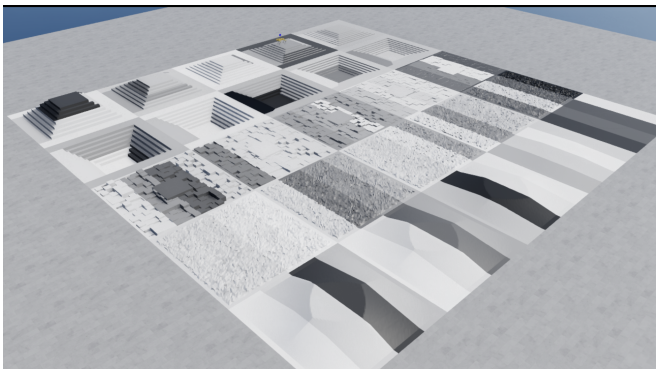


Fig. 5: Examples of procedurally generated terrains in Isaac Sim, including stairs, slopes, steps, and rough surfaces. Color variations indicate different surface friction levels, where lighter regions correspond to lower friction and darker regions to higher friction.

Two environment configurations are used depending on the policy mode. For flat-ground whole-body control, the robot is spawned on an infinite flat plane. For terrain locomotion, a diverse set of procedurally generated terrains is used to encourage robustness to diverse terrains. These include up and down pyramid staircases, steps, random rough terrain, and pyramid slopes with varying inclines, as shown in Figure 5. The terrain surfaces additionally incorporate randomized coefficients of friction to simulate both slippery and high-friction ground conditions.

To facilitate transfer from simulation to the real robot, we employ dynamics randomization during training. The randomized parameters include body friction coefficients,

TABLE I: Dynamics randomization parameters and sampling ranges. Percentage values indicate deviation from the nominal value.

Element	Range
Body static friction	$(0.3, 1.0)$
Body dynamic friction	$(0.3, 0.8)$
Body mass	$(-25, 25)\%$
Joint stiffness	$(-25, 25)\%$
Joint damping	$(-25, 25)\%$
Joint position	$(-50, 50)\%$
Base force	$(-10, 10)\ N$
Base push velocity	$(-1.5, 1.5)\ m/s$
Ground friction	$(0.1, 2.0)$

body mass, joint stiffness and damping, and initial joint positions. External disturbances such as random pushes and base velocity perturbations are also applied to improve robustness to unexpected interactions. The parameter ranges used during training are summarized in Table I.

D. Training Curriculum

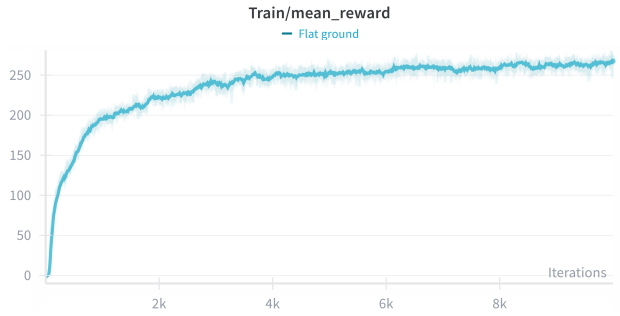
For flat-ground whole-body control, we train the policy across the full range of command inputs without a curriculum. However, for terrain locomotion, we adopt a two-stage curriculum designed to gradually increase task difficulty as the policy improves.

In both stages, terrains are parameterized by a difficulty level ranging from 1 to 10. Each terrain type is configured differently across these levels; for instance, stair height increases with level for staircase scenarios, and slope gradients increase for inclined surfaces. During training, an environment can be promoted or demoted between levels based on the agent's progression under the commanded velocity. This adaptive mechanism ensures that the policy is exposed to challenges that match its current capability.

In Stage 1, friction is held constant, and step heights are limited to $0.2\,\mathrm{m}$ to simplify the initial learning problem. This is especially important for blind traversal, as stairs and steps are the most challenging terrain types without exteroceptive information. In Stage 2, the curriculum introduces friction variation according to Table I, and the maximum stair height is increased to $0.3\,\mathrm{m}$. These changes expose the policy to a wider range of dynamic conditions and terrain properties as training progresses. This structured curriculum enables the policy to first acquire fundamental locomotion skills before adapting to more diverse and demanding terrain.

E. Policy and Learning framework

Following prior works on quadruped control [14], [17], we formulate the whole-body locomotion task using an actor-critic framework, and adopt Proximal Policy Optimization (PPO) [25] as the learning algorithm. The actor represents the stochastic control policy $\pi(a|s)$ and outputs the mean of a Gaussian action distribution, while a learned log-standard-deviation parameter is used for sampling actions during training. The critic models the value function $V(s)$ and estimates expected returns for advantage computation.



(a) Training reward curve for flat-ground whole-body policy.



(b) Training reward curve for terrain locomotion policy.

Fig. 6: Reward learning curves for both trained policies.

Both networks are implemented as multilayer perceptrons (MLPs) with three hidden layers of sizes $[512, 256, 128]$, using ELU activation functions after each layer. The final layers of the actor and critic are linear (no activation), ensuring an unconstrained mapping to policy outputs and value predictions.

We train the actor and critic jointly using the Adam optimizer. At each iteration, the learning rate is adapted based on the KL divergence between the updated policy and the previous one, which encourages stable and conservative policy updates. Training is performed using on-policy rollouts collected in simulation with thousands of parallel environments. The PPO hyperparameters used in our experiments are summarized in Table II.

Figure 6 presents the training reward curves for both the flat-ground whole-body policy and the terrain locomotion policy. The flat-ground controller is trained for $10{,}000$ iterations, and the terrain controller for $20{,}000$ iterations. Training is executed on a single NVIDIA H100 GPU, requiring approximately four hours for the flat-ground policy and eleven hours for the terrain policy to converge.

IV. Experiments and Results

In this section, we describe the experimental setup used to evaluate our RL controllers in simulation. We then present the quantitative performance results and discuss the sim-to-real capabilities of the learned policies.

TABLE II: PPO hyperparameters used for training the whole-body locomotion policies.

Hyperparameter	Value
Learning rate	1e-3
PPO clip range	0.2
Number of epochs	5
Number of mini batches	4
Discount Factor (Gamma)	0.99
GAE Lambda	0.95
Value Function Coefficient	1.0
Entropy Coefficient	0.01
Max Grad Norm	1.0
Steps per environment	24
Number of environments	4096
Episode length	1000

A. Experimental Setup

The learned policies are evaluated in simulation across seven commanded motion types: forward and backward walking along the x-axis, lateral walking along the y-axis, clockwise and counterclockwise yaw rotation, and standing. For velocity tracking experiments, we perform 100 rollouts for each command magnitude in the range $[-1.0, 1.0]\,\mathrm{m/s}$ using increments of $0.1\,\mathrm{m/s}$, with each rollout lasting 20 seconds.

To assess terrain locomotion robustness, we additionally evaluate the policy over long-horizon trials on a variety of challenging surfaces, including stairs, steps, slopes, and rough terrain. During these experiments, velocity commands are randomly sampled, and the ground friction is held constant to isolate terrain difficulty effects.

We report two primary evaluation metrics:

1) Commanded drift: This metric measures the pose deviation from the expected trajectory given the commanded motion on flat terrain. For a commanded velocity of $1.0\,\mathrm{m/s}$ along the x-axis, the robot is expected to travel $20\,\mathrm{m}$ over a 20-second episode with minimal lateral or yaw deviation. The commanded drift is defined as the mean absolute error in final x, y, and yaw positions averaged across all rollouts for each command direction.

2) Mean distance per fall: This metric evaluates the stability of the learned controller on challenging terrains. We test performance across five terrain types: ground, stairs, steps, slopes, and rough surfaces, while keeping ground friction constant. Two evaluation experiments are considered. In the first, the robot is given randomly sampled velocity commands every 3–5 seconds, drawn from the full training range of $[-1.0, 1.0]$ for all three velocity axes, while posture commands remain fixed. In the second experiment, we isolate forward locomotion by sampling only forward velocity commands in the range $[0.1, 1.0]\,\mathrm{m/s}$ at the same interval. A fall event occurs when the robot's body flips over into an unrecoverable state. The mean distance per fall is computed as the total distance traveled divided by the number of falls across all parallel environments.

TABLE III: Mean absolute drift error in final robot pose relative to the commanded motion for different command types.

Command Type	Δx (m)	Δy (m)	$\Delta\theta$ (rad)
Forward walk	1.601	1.953	0.323
Backward walk	0.461	0.592	0.115
Left walk	2.999	1.971	0.488
Right walk	0.431	1.409	0.082
Counter-clockwise turn	0.137	0.167	0.057
Clockwise turn	0.110	0.150	0.406
Stand Still	0.017	0.003	0.017

TABLE IV: Mean distance traveled between fall events for the flat-ground baseline policy (π_{Flat}) and the terrain-trained policy (π_{Terrain}) across different terrains and command settings. Entries marked with * indicate that no fall was observed during the evaluation horizon; reported values therefore represent lower bounds.

Commands	Terrain	π_{Flat} (m)	π_{Terrain} (m)
Random	Ground	5450.482	5037.605*
	Stairs	4.353	790.069
	Steps	6.753	733.968
	Slopes	64.842	979.372
	Rough terrain	2.552	364.938
Forward	Ground	3876.838*	3324.181*
	Stairs	4.211	829.150
	Steps	7.013	717.663
	Slopes	68.421	1029.557
	Rough terrain	3.018	387.814

B. Results

1) Commanded drift: Table III summarizes the drift performance for each commanded locomotion behavior. The stand task exhibits near-zero drift across all directions, indicating that the learned policy remains stable without taking unnecessary corrective steps. This is desirable for applications such as inspection or manipulation, where reliable static positioning is important.

For linear walking, we observe higher drift in the forward direction than in backward motion. While the exact underlying cause is not obvious, similar behavior has been noted in prior quadruped studies where asymmetries in platform dynamics and morphology can lead to directional performance biases. In our case, this difference emerged consistently during training and did not significantly impact locomotion robustness. Reducing this discrepancy could be addressed through reward shaping or asymmetric modeling considerations if required in future applications.

Sideways walking generally shows larger drift than for-

ward or backward motion. This is expected, as quadruped leg geometry is naturally optimized for forward-plane movement, and lateral locomotion requires larger hip abduction and less mechanically favorable contact configurations. As lateral command magnitude increases, the accumulated transverse drift becomes more noticeable.

Turning behavior remains well-regulated, with low translational drift during rotation tasks. Angular tracking error remains within acceptable bounds for both clockwise and counterclockwise turning, demonstrating that the policy can execute rotations in place while maintaining approximate heading stability.

We note that small lateral or yaw deviations during forward and sideways locomotion can accumulate over extended horizons due to the stepwise nature of the velocity tracking reward used during training. Incorporating longer-horizon tracking terms or trajectory-aware rewards could further reduce accumulated drift if tighter navigation requirements are desired. However, minimizing drift in non-commanded directions was not a primary objective of this work, and current performance is sufficient for our evaluation.

2) Mean distance per fall: Table IV compares the terrain robustness of the terrain-trained policy against a flat-ground baseline under two evaluation settings: randomly sampled velocity commands and forward-only commands. Across all non-flat terrain types and both command regimes, the terrain-trained controller achieves substantially longer distances between fall events. On slopes, the improvement approaches an order of magnitude, indicating that curriculum-based terrain exposure enables effective balance recovery and adaptation to smooth elevation changes. Similarly large gains are observed on stairs and steps, reflecting more reliable foot placement on sharp geometric transitions despite the absence of exteroceptive sensing. Even on rough terrain, where irregular height variations introduce unpredictable disturbances, the terrain-trained policy consistently maintains significantly higher stability than the flat-ground baseline.

The flat-ground results provide important context for interpreting these numbers. On level ground, both policies exhibit very long fall-free operation, and in several cases, no failure was observed within the evaluation horizon. As indicated by the starred entries in Table IV, these values represent lower bounds on the achievable distance before failure. Notably, the terrain-trained policy maintains comparable stability to the flat-ground baseline on flat terrain, indicating that robustness to uneven terrain does not come at the cost of degraded flat-ground performance.

These results also highlight that a policy trained exclusively on flat terrain does not generalize to uneven environments. The baseline controller typically fails within only a few meters on stairs, steps, and rough terrain, regardless of whether commands are random or forward-only, demonstrating its inability to compensate for terrain-induced perturbations or unexpected foothold changes.

To better isolate the effects of terrain difficulty from the momentum shifts caused by random command sampling, we additionally evaluate both policies under forward-only commands. The terrain-trained controller again outperforms the baseline by a large margin, with slightly higher distances than in the random-command trials. This modest increase is expected, as reducing abrupt command transitions removes a common source of failure. Overall, the similarity between the two evaluation modes suggests that terrain geometry, rather than command variability, is the dominant factor governing locomotion stability for these policies.

Although this evaluation setup introduces variability, it reflects realistic operation in unpredictable environments and provides a meaningful estimate of long-term locomotion resilience. In this context, the reported distances should be viewed as indicative upper bounds on the capability of our blind whole-body controller to traverse challenging terrain before experiencing an unrecoverable failure.

C. Sim-to-Real

We deploy the learned whole-body policies on a Unitree Go2 platform for qualitative evaluation, as shown in Figure 1. The trained actor network runs onboard at $50\,\mathrm{Hz}$ on the robot's Jetson computer and communicates joint commands through the built-in DDS interface to the robot's low-level controller. The policy is commanded using a handheld joystick operated by a human user.

The robot exhibits behaviors on hardware that closely match those observed in simulation, demonstrating a strong degree of sim-to-real transfer under our testing conditions. The controller successfully executes commanded forward, lateral, and rotational motions, as well as height, roll, and pitch adjustments, while maintaining overall balance. We additionally apply external pushes during operation, and the policy consistently recovers without falling, suggesting robustness to moderate disturbances.

All hardware experiments are conducted on flat ground without external perception or terrain sensing. While preliminary, these results validate the blind locomotion capabilities of our approach and provide confidence in extending to more challenging real-world environments.

V. Conclusion

Quadruped robots offer significant advantages in environments that are inaccessible to wheeled platforms, requiring agile and coordinated whole-body control to traverse uneven surfaces safely. RL provides a promising avenue for developing such controllers by leveraging rich interaction in simulation without requiring explicit modeling of complex robot dynamics or contact interactions.

In this work, we developed a blind whole-body locomotion controller for quadrupeds that enables height modulation, posture adjustment, and omnidirectional velocity tracking. The learned policies demonstrate strong performance in simulation with minimal drift during commanded motions and robust traversal across a diverse set of terrain types. Moreover, qualitative hardware deployment on a Unitree Go2 robot shows effective sim-to-real transfer without perception modules, validating the practicality of proprioception-only blind locomotion under real operating conditions.

There are several directions for continued improvement. Residual drift observed during prolonged locomotion could be reduced through longer-horizon rewards or trajectory-aware tracking rewards. Incorporating exteroceptive sensing may further enhance foothold selection and robustness on challenging terrain [30], [27], [5]. Additionally, integrating a fall recovery behavior would expand operational autonomy by enabling the robot to recover after failures [31].

Overall, this work demonstrates the potential of deep reinforcement learning to enable scalable and robust whole-body locomotion for quadruped robots, providing a foundation for more capable operation in real-world environments.

REFERENCES

[1] S. A. Prieto, N. Giakoumidis, and B. García de Soto, "Multiagent robotic systems and exploration algorithms: Applications for data collection in construction sites," *Journal of Field Robotics*, vol. 41, no. 4, pp. 1187–1203, 2024. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.22316.

[2] K. Ankit, S. N. Kolathaya, and D. Ghose, "Multi-Agent Collaborative Framework for Automated Agriculture," in *2021 15th International Conference on Advanced Computing and Applications (ACOMP)*, pp. 30–37, Nov. 2021. ISSN: 2688-0202.

[3] D. S. Drew, "Multi-Agent Systems for Search and Rescue Applications," *Current Robotics Reports*, vol. 2, pp. 189–200, June 2021. Company: Springer Distributor: Springer Institution: Springer Label: Springer Number: 2 Publisher: Springer International Publishing.

[4] M. L. Elwin, B. Strong, R. A. Freeman, and K. M. Lynch, "Human-Multirobot Collaborative Mobile Manipulation: The Omnid Mocobots," *IEEE Robotics and Automation Letters*, vol. 8, pp. 376–383, Jan. 2023. Conference Name: IEEE Robotics and Automation Letters.

[5] 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, p. eabk2822, Jan. 2022. Publisher: American Association for the Advancement of Science.

[6] F. Jenelten, J. He, F. Farshidian, and M. Hutter, "DTC: Deep Tracking Control," *Science Robotics*, vol. 9, p. eadh5401, Jan. 2024. arXiv:2309.15462 [cs].

[7] Z. Zhang, H. An, Q. Wei, and H. Ma, "Learning-Based Model Predictive Control for Quadruped Locomotion on Slippery Ground," in *2022 4th International Conference on Control and Robotics (ICCR)*, pp. 47–52, Dec. 2022.

[8] J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. Kim, "Dynamic Locomotion in the MIT Cheetah 3 Through Convex Model-Predictive Control," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–9, Oct. 2018. ISSN: 2153-0866.

[9] S. Xu, L. Zhu, H.-T. Zhang, and C. P. Ho, "Robust Convex Model Predictive Control for Quadruped Locomotion Under Uncertainties," *IEEE Transactions on Robotics*, vol. 39, pp. 4837–4854, Dec. 2023.

[10] M. Neunert, M. Stäuble, M. Giftthaler, C. D. Bellicoso, J. Carius, C. Gehring, M. Hutter, and J. Buchli, "Whole-Body Nonlinear Model Predictive Control Through Contacts for Quadrupeds," *IEEE Robotics and Automation Letters*, vol. 3, pp. 1458–1465, July 2018. Conference Name: IEEE Robotics and Automation Letters.

[11] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science Robotics*, vol. 5, p. eabc5986, Oct. 2020.

[12] Z. Xu, A. H. Raj, X. Xiao, and P. Stone, "Dexterous Legged Locomotion in Confined 3D Spaces with Reinforcement Learning," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11474–11480, May 2024.

[13] E. Chane-Sane, J. Amigo, T. Flayols, L. Righetti, and N. Mansard, "SoloParkour: Constrained Reinforcement Learning for Visual Locomotion from Privileged Experience," in *Proceedings of The 8th Conference on Robot Learning*, pp. 4268–4285, PMLR, Jan. 2025. ISSN: 2640-3498.

[14] G. B. Margolis and P. Agrawal, "Walk These Ways: Tuning Robot Control for Generalization with Multiplicity of Behavior," in *Proceedings of The 6th Conference on Robot Learning*, pp. 22–31, PMLR, Mar. 2023. ISSN: 2640-3498.

[15] G. B. Margolis, G. Yang, K. Paigwar, T. Chen, and P. Agrawal, "Rapid locomotion via reinforcement learning," *The International Journal of Robotics Research*, vol. 43, pp. 572–587, Apr. 2024. Publisher: SAGE Publications Ltd STM.

[16] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "RMA: Rapid Motor Adaptation for Legged Robots," July 2021. arXiv:2107.04034 [cs].

[17] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to Walk in Minutes Using Massively Parallel Deep Reinforcement Learning," in *Proceedings of the 5th Conference on Robot Learning*, pp. 91–100, PMLR, Jan. 2022. ISSN: 2640-3498.

[18] J. Liang, V. Makoviychuk, A. Handa, N. Chentanez, M. Macklin, and D. Fox, "Gpu-accelerated robotic simulation for distributed reinforcement learning," in *Proceedings of The 2nd Conference on Robot Learning* (A. Billard, A. Dragan, J. Peters, and J. Morimoto, eds.), vol. 87 of *Proceedings of Machine Learning Research*, pp. 270–282, PMLR, 29–31 Oct 2018.

[19] D. Lee, N. He, P. Kamalaruban, and V. Cevher, "Optimization for Reinforcement Learning: From a single agent to cooperative agents," *IEEE Signal Processing Magazine*, vol. 37, pp. 123–135, May 2020. Conference Name: IEEE Signal Processing Magazine.

[20] M. van Otterlo and M. Wiering, "Reinforcement Learning and Markov Decision Processes," in *Reinforcement Learning*, pp. 3–42, Springer, Berlin, Heidelberg, 2012. ISSN: 1867-4542.

[21] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[22] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*, pp. 1889–1897, PMLR, 2015.

[23] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 23–30, IEEE, 2017.

[24] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "A brief survey of deep reinforcement learning," *arXiv preprint arXiv:1708.05866*, 2017.

[25] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," Aug. 2017. arXiv:1707.06347 [cs].

[26] Z. Fu, A. Kumar, J. Malik, and D. Pathak, "Minimizing Energy Consumption Leads to the Emergence of Gaits in Legged Robots," in *Proceedings of the 5th Conference on Robot Learning*, pp. 928–937, PMLR, Jan. 2022. ISSN: 2640-3498.

[27] A. Agarwal, A. Kumar, J. Malik, and D. Pathak, "Legged Locomotion in Challenging Terrains using Egocentric Vision," in *Proceedings of The 6th Conference on Robot Learning*, pp. 403–415, PMLR, Mar. 2023. ISSN: 2640-3498.

[28] P. Arm, M. Mittal, H. Kolvenbach, and M. Hutter, "Pedipulate: Enabling Manipulation Skills using a Quadruped Robot's Leg," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5717–5723, May 2024.

[29] X. He, C. Yuan, W. Zhou, R. Yang, D. Held, and X. Wang, "Visual Manipulation with Legs," in *Proceedings of The 8th Conference on Robot Learning*, pp. 4218–4234, PMLR, Jan. 2025. ISSN: 2640-3498.

[30] Z. Fu, A. Kumar, A. Agarwal, H. Qi, J. Malik, and D. Pathak, "Coupling Vision and Proprioception for Navigation of Legged Robots," July 2022. arXiv:2112.02094 [cs].

[31] Y. Lu, Y. Dong, J. Ma, J. Zhang, and P. Lu, "Learning an Adaptive Fall Recovery Controller for Quadrupeds on Complex Terrains," Dec. 2024. arXiv:2412.16924 [cs].

TABLE V: Summary of all reward terms used for training the blind whole-body locomotion controller.

Category	Reward term	Equation	Weight
Task	Linear velocity tracking	$r_{\text{lin}} = \exp\left(-\frac{\left\|v_{\text{cmd}}^{xy} - v_b^{xy}\right\|_2}{1.0}\right) \cdot \max\left(1.0,\ 1.0 + 0.5\left(\left\|v_{\text{cmd}}^{xy}\right\|_2 - 1.0\right)\right)$	5.0
	Angular velocity tracking	$r_{\text{ang}} = \exp\left(-\frac{\left\|\omega_{\text{cmd}}^{z} - \omega_b^{z}\right\|}{2.0}\right)$	5.0
	Height tracking	$r_{\text{height}} = \exp\left(-\frac{\left\|h_{\text{cmd}} - h\right\|}{0.05}\right)$	5.0
	Body tilt tracking	$r_{\text{tilt}} = \exp\left(-\frac{\left\|r_{\text{cmd}} - r\right\| + \left\|p_{\text{cmd}} - p\right\|}{0.2}\right)$	5.0
Style	Gait reward	$r_{\text{gait}} = \exp\left(-\frac{\Delta t_{\text{air}}^2 + \Delta t_{\text{contact}}^2}{0.1}\right)$	5.0
	Feet air-time reward	$r_{\text{air}} = \sum_{i=1}^{4} \max\left(0,\ t_{\text{air},i} - 1.0\right)$	5.0
	Air-time variance penalty	$r_{\text{var}} = \left(\text{Var}(t_{\text{air}}) + \text{Var}(t_{\text{contact}})\right)$	-1.0
	Feet contact reward	$r_{\text{contact}} = \frac{\text{number of feet in contact}}{4}$ if $\|v_b\|_2 < 0.25$ else 0	5.0
Auxiliary	Foot clearance	$r_{\text{clear}} = \begin{cases} \exp\left(-\frac{\sum_{i=1}^{4}\left(z_i - 0.06\right)^2 \tanh\left(2\left\|v_{\text{foot},i}^{xy}\right\|_2\right)}{0.05}\right), & \|v_{\text{cmd}}\|_2 > 0, \\ 0, & \text{otherwise} \end{cases}$	0.5
	Foot slip penalty	$r_{\text{slip}} = \sum_{i=1}^{4} \mathbb{I}(F_i \geq 1.0) \left\|v_{\text{foot},i}^{xy}\right\|_2$	-0.5
	Base motion penalty	$r_{\text{base}} = 0.5\, v_z^2 + 0.5\left(\left\|\omega_{\text{roll}}\right| + \left|\omega_{\text{pitch}}\right|\right)$	-4.0
	Joint deviation penalty	$r_{\text{joint}} = \left(0.7\left\|q_{\text{leg}} - q_{\text{leg}}^0\right\|_2 + 1.4\left\|q_{\text{hip}} - q_{\text{hip}}^0\right\|_2\right) \cdot \begin{cases} 5.0, & \|v_{\text{cmd}}\|_2 \leq 0\ \wedge\ \|v_b^{xy}\|_2 \leq 0.5, \\ 1.0, & \text{otherwise} \end{cases}$	-1.0
Regularization	Action smoothness	$r_{\text{smooth}} = \left\|a_t - a_{t-1}\right\|_2$	-1.0
	Joint torque penalty	$r_\tau = \left\|\boldsymbol{\tau}\right\|_2$	-5.0×10^{-4}
	Joint acceleration penalty	$r_{\text{acc}} = \left\|\ddot{\mathbf{q}}\right\|_2$	-1.0×10^{-4}
	Joint velocity penalty	$r_{\text{vel}} = \left\|\dot{\mathbf{q}}\right\|_2$	-1.0×10^{-2}

APPENDIX I
REWARD TABLE

Table V lists the reward terms used in the training, along with the equations and weights. Below is the key for the notations used.

v_{cmd}: commanded base linear and angular velocity.
v_b: measured base linear velocity in the body frame.
ω_b: measured base angular velocity in the body frame.
h: robot base height; h_{cmd}: commanded base height.
r, p: measured base roll and pitch; $r_{\text{cmd}}, p_{\text{cmd}}$: commanded roll and pitch.
z_i: vertical position of foot i in the world frame.
$v_{\text{foot},i}^{xy}$: planar velocity of foot i.
F_i: ground contact force magnitude at foot i.
t_{air}, t_{contact}: recent air/contact durations for a foot.
q: joint position; q^0: default joint position.
$\dot{q}$, $\ddot{q}$: joint velocity and acceleration.
τ: applied joint torque.
a_t: policy action at time t.