Extended Abstract

Motivation Robotic navigation in cluttered environments requires not only global path planning, but also fine-grained joint-level adaptation to avoid collisions, preserve stability, and achieve task goals. Classical approaches like inverse kinematics often fail under tight spatial constraints due to their lack of environmental awareness and limited flexibility in high-degree-of-freedom (DoF) systems. This project investigates whether we can train an environment-aware whole-body controller that autonomously maneuvers through obstacles using reinforcement learning (RL). Our goal is to develop a joint-space control policy that learns from expert demonstrations and dense simulation feedback—enabling reactive, contact-averse behaviors without relying on full handcrafted models or manually tuned planners.

Method We formulate the problem as a continuous control task and train a joint velocity policy using Proximal Policy Optimization (PPO) in simulation. The agent receives privileged input, including joint states, end-effector pose, and obstacle positions, and outputs smooth velocity commands across all DoFs. Expert trajectories, collected via teleoperation, serve as behavioral references and shape the reward signal. The reward encourages goal-reaching while penalizing positional and rotational error and proximity to obstacles. To improve gradient flow near contact points, we test both a binary collision penalty and a smoother Gaussian proximity penalty. This setup promotes learning stable whole-body coordination strategies that generalize across diverse scenes.

Implementation Our training pipeline is built atop MuJoCo and Brax, with a modular environment defined through `dm_control` and Hydra-based YAML configurations. A UR5e robotic arm on a floating base operates in procedurally generated arenas filled with random obstacle layouts. A custom teleoperation interface using a 3D SpaceMouse and keyboard enables intuitive demonstration capture. We run PPO over 1024 parallel environments for 600 time steps per rollout, with a batch size of 128 and learning rate 3×10^{-4}. Reward weights were tuned empirically to prioritize smoothness and collision avoidance.

Results In 128 held-out environments, our PPO policy with Gaussian penalties achieved a 58.1% success rate—more than double that of the naive IK controller (21.1%) and a PPO variant without collision-aware rewards (24.6%). It also demonstrated smoother trajectories, lower energy usage, and better imitation of expert behavior. Visual rollouts confirm the controller's ability to adapt joint angles dynamically to maneuver around narrow passages, with fewer collisions and less erratic motion than baseline policies.

Conclusion Our results show that reward-shaped RL combined with expert priors can produce adaptable, collision-aware control policies for navigating dense environments. The modular simulation and training framework supports scalable experimentation and provides a foundation for future work on student policy distillation under partial observations (e.g., RGB), sim-to-real transfer, and energy-constrained deployment. This project offers a promising step toward intelligent whole-body control in real-world cluttered settings.

Environment-Aware Whole-Body Controller: A Simulated Approach Towards Training Joint Movements Through Barriers

Jeff Liu
Department of Computer Science
Stanford University
jeffliu0@stanford.edu

Abstract

This project investigates the development of a reinforcement learning-based whole-body controller capable of navigating cluttered environments using precise joint-level control. Traditional motion planning approaches often struggle with high-degree-of-freedom systems due to computational inefficiency and limited adaptability in dynamic or constrained spaces. To address these challenges, we propose a learning-based controller trained entirely in simulation, leveraging expert demonstrations to enable environment-aware behavior.

Using a modular MuJoCo framework, we collect expert trajectories via human teleoperation and train a privileged observation policy with full access to simulator state. The policy is optimized using Proximal Policy Optimization (PPO) and guided by a reward function that encourages goal-reaching, smooth joint motion, and collision avoidance. Results have shown that the trained policy can generalize across diverse obstacle configurations and reliably execute control strategies. This work establishes a foundation for robust, learning-driven whole-body control and provides infrastructure for future extensions involving partial observability and sim-to-real transfer.

1 Introduction

Robots operating in unstructured, cluttered environments must not only plan high-level paths, but also execute low-level joint motions that account for physical constraints, dynamic obstacles, and narrow passages. Traditional motion planning pipelines, which often rely on trajectory optimization or inverse kinematics (IK), can become computationally expensive and brittle in high-degree-of-freedom (DoF) systems. These approaches also tend to lack adaptability when faced withwith previously unseen configurations or subtle environmental variations, making them less suitable robust real-timeime deployment.

Recent advances in reinforcement learning (RL) have shown promise in enabling robots to learn complex behaviors directly from interaction, especially in continuous control domains. RL-based controllers can offer fast inference once trained and, more importantly, can learn to adapt to environmental nuances without requiring detailed models or hand-tuned heuristics. However, developing such controllers for whole-body control in cluttered scenes remains challenging due to the intricacy of joint coordination and the need for fine-grained collision avoidance.

In this work, we focus on training an environment-aware, learning-based whole-body controller that enables a robotic arm to navigate through dense barrier configurations while maintaining stability and minimizing collisions. We construct a modular simulation environment in MuJoCo to support dynamic obstacle layouts and diverse scene configurations. Expert demonstrations are collected using

human teleoperation via a 3D SpaceMouse, capturing realistic and adaptable joint-level behaviors. These demonstrations serve as supervision for training a teacher policy using Proximal Policy Optimization (PPO), with access to privileged simulator states.

A key aspect of this project is the design of a modular and extensible simulation infrastructure that enables rapid experimentation with different reward structures, robot configurations, and scene layouts. By decoupling the robot, scene, and object definitions into separate modules, the framework supports flexible, scalable training environments. This design is especially valuable for reinforcement learning, where diverse and randomized environments improve policy generalization. The resulting controller demonstrates consistent performance across previously unseen configurations, highlighting the potential of learning-based methods for practical, low-level robotic control in constrained spaces.

The contributions of this project include:

1. A flexible MuJoCo simulation framework for generalized reinforcement learning training in simulated environments (as part of the pipeline for whole-body controller training).
2. An expert demonstration pipeline (including keyboard and 3D SpaceMouse server components) for complex teleoperation and joint control behaviors in simulated spaces.
3. An RL policy trained on privileged state information that enables collision-aware navigation and provides a foundation for later exploration of optimal barrier and environment representations when distilling the policy under partial observations.

The results with the policy trained on full-state information demonstrate that reinforcement learning can produce reliable whole-body control strategies suitable for dense, and potentially real-world environments. This work lays the groundwork for future exploration of vision-based control under partial observability and eventual sim-to-real deployment.

2 Related Work

Learning-based whole-body control has advanced rapidly across robotics, particularly in locomotion, manipulation, and unified policy learning. However, most prior work focuses on dynamic movement or object interaction, rather than precise, collision-aware joint control for navigating cluttered ground environments.

- **UMI-on-Legs** Ha et al. (2024) combines real-world gripper demonstrations with simulation-based training to enable quadrupeds to perform object manipulation tasks like pushing and tossing. The system relies on partial visual inputs and trains whole-body policies to replicate learned manipulator behaviors. While effective for mobile manipulation, the task settings involve open spaces and fixed goals, with no obstacle-dense navigation. The controller is not required to adjust joint configurations in response to spatial constraints, which is central to this project's objective.
- **Extreme Parkour** Cheng et al. (2023) trains quadrupeds to execute dynamic stunts—jumps, flips, handstands—using end-to-end reinforcement learning from RGB-D input. Their work demonstrates high-speed, agile locomotion driven by perception, but the environments are mostly open. The focus is on motion stability and expressiveness, not low-speed, fine-grained control. Our work differs by addressing precise movement in tight spaces, where reactive joint control is critical for avoiding collisions.
- **HOVER** He et al. (2025) presents a unified whole-body controller for humanoids by distilling multiple skills into a single policy. Their system handles diverse tasks like walking and reaching using full-body proprioception and full-state access. While general and robust, the environments are free of clutter, and the policy is not tested in constrained spaces. Our focus is narrower but deeper: enabling stable navigation in dense, obstacle-rich layouts under spatial and sensory limitations.
- **Whole-Body Control Through Narrow Gaps** Wu et al. (2024) trains quadrotors to fly through tight gaps using vision-based policies. The policy maps raw pixels to motor commands with temporal consistency. Though both works address navigation under spatial constraints, their domain is aerial and underactuated, where joint-level articulation and ground contact are not relevant. Our setting introduces additional challenges like limb collisions, joint limits, and static stability.

- **Deep Whole-Body Control** Fu et al. (2022) develops a unified policy for quadrupeds with manipulators, coordinating locomotion and arm movements through reinforcement learning. The system operates in open environments and focuses on multi-task coordination, not obstacle avoidance. Their policy is not tested in cluttered spaces where precise spatial adaptation is required. In contrast, our work targets joint-level control specifically for navigating narrow environments with high collision risk.

In summary, while many prior works tackles dynamic motion, object interaction, and generalist control, none directly address precise, environment-aware whole-body navigation through cluttered spaces using articulated ground robots. Our project aims to fill this gap by combining expert-guided demonstrations, modular simulation, and reinforcement learning to train a controller that adapts joint behavior in response to dense, physical obstacles.

3 Method

We present a reinforcement learning framework for training a whole-body controller capable of navigating cluttered environments through joint-level adaptation. Our approach is developed entirely in simulation and consists of three core components: (1) a modular MuJoCo-based environment with Hydra configuration support for flexible scene generation, (2) expert trajectory generation via human teleoperation, and (3) a policy training pipeline using Proximal Policy Optimization (PPO) through MuJoCo Playground Zakka et al. (2025), combined with Brax Freeman et al. (2021) for parallelized training. The system is designed to promote collision-aware, goal-directed motion and serves as a foundation for future extensions under partial observability.

3.1 Simulation Environment

To support diverse and reconfigurable scene layouts for whole-body controller training, we design a modular simulation framework using the MuJoCo physics engine and the `dm_control` suite. A key focus of our approach is enabling dynamic environment construction to support obstacle avoidance, spatial adaptation, and generalization.

We initially experimented with a monolithic MuJoCo XML configuration that statically defined the robot, gripper, and environment in a single file. While sufficient for basic visualization and fixed-scene control, this setup lacked flexibility for varying obstacle layouts, robot instantiations, and scene transitions—limiting its usefulness for reinforcement learning.

To address this, we restructured the environment into a modular architecture composed of four base classes: `BaseEnv`, `BaseScene`, `BaseRobot`, and `BaseObject`. These abstractions encapsulate distinct functionalities and enable component-wise reuse across different tasks and layouts. Each base class is responsible for:

- `BaseEnv`: Integrates robots, objects, and scene definitions into a unified simulation interface. It initializes physics, loads MJCF models, and anchors robots and objects using mocap bodies and attachment sites. It handles simulation stepping, resets, and observation collection.
- `BaseScene`: Defines the static simulation world, loads the MJCF scene file, configures cameras, and maintains references to simulation state. It provides the spatial context in which the robot and objects are placed.
- `BaseRobot`: Encapsulates a robotic arm (with optional mobile base and gripper), loading MJCF components and initializing joint and actuator references. It supports Cartesian control via inverse kinematics using the `mink` solver, handles gripper actuation, exposes robot state, and includes geometry IDs for collision detection.
- `BaseObject`: Represents individual objects in the scene. It loads object models from MJCF, stores metadata such as size, pose, and rotation limits, and provides methods to retrieve or set object state and geometry information for collision handling.

Dynamic instantiation of these modules is controlled through Hydra-based YAML configuration files, allowing flexible specification of object placements, robot parameters, and layouts without modifying code.

Our robot model consists of a UR5e arm mounted on a floating base (restricted to XYZ translations), equipped with a UMI gripper. As shown in Figure 1, it operates in a bounded arena with randomly placed or procedurally generated obstacles. These obstacles act as physical barriers that the robot must avoid while maneuvering its end-effector toward a target pose. During training, contact forces, object states, and joint data are extracted from the simulator to support environment-aware control learning.

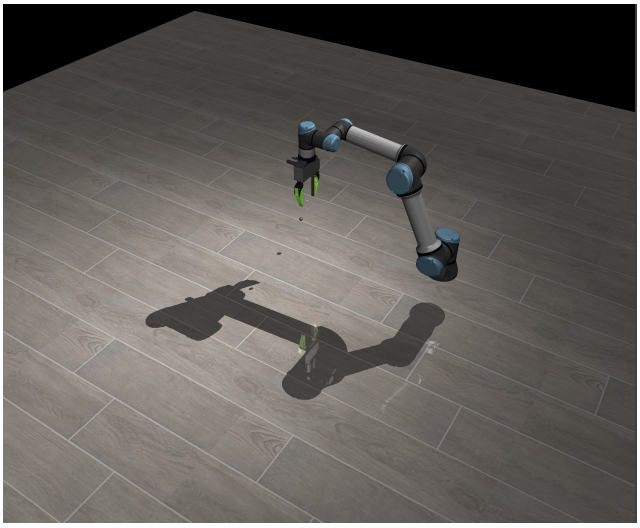

Figure 1: Training environment built in MuJoCo, showing a UR5e robotic arm on a floating base. Modular components are configured using Hydra.

This design supports rapid task prototyping, curriculum learning, and robust experimentation across diverse spatial scenarios—ultimately enabling more generalizable control policy development.

3.2 Expert Demonstration Collection

To guide reinforcement learning and facilitate reward shaping, we collect expert demonstrations via a human-operated 3D SpaceMouse and keyboard interface. The SpaceMouse is used to manipulate the end-effector pose, while the keyboard adjusts the floating base. This allows the operator to generate smooth, goal-directed trajectories around obstacles in real time. Each trajectory is recorded as a sequence of end-effector positions and orientations (XYZ and quaternion WXYZ), which are later used to initialize training and contribute to the reward function.

The teleoperation interface is implemented in Python and designed for low-latency, intuitive control. It enables precise joint configuration adjustments in cluttered environments. The resulting trajectories reflect natural collision-avoidance strategies and joint usage patterns, providing strong behavioral priors for early policy learning.

3.3 Policy Learning

We train the controller using PPO, an on-policy reinforcement learning algorithm suited for continuous control tasks. The policy maps simulator state observations to joint-level velocity commands, allowing for smooth motion generation. The action space includes all controllable joints of the robot arm, while the observation space consists of joint positions and velocities, end-effector pose, and obstacle locations and poses.

Reward Function

The reward function is designed to balance task success, safety, and behavioral realism. A binary success reward is granted when the end-effector enters a goal region. Additional penalties apply for positional and rotational error, as well as for proximity to obstacles. The reward encourages the robot to reach the goal while minimizing unsafe or unstable motions.

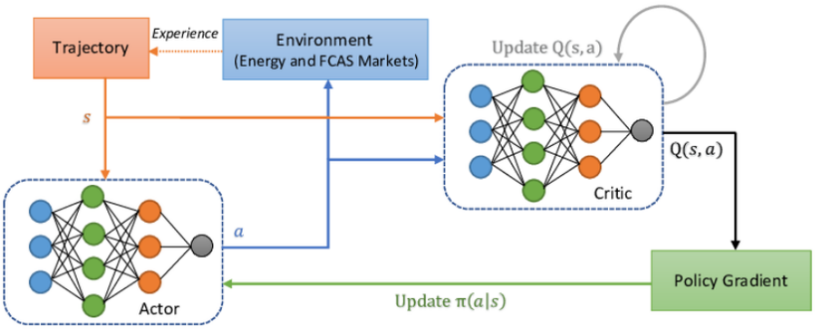

Figure 2: PPO training pipeline. Policies map simulator state to joint velocity commands, with rewards based on task success, imitation, and collision avoidance.

$$r = r_{\text{success}} - w_{\text{pos}} \cdot \text{pos\_err} - w_{\text{rot}} \cdot \text{rot\_err} - w_{\text{col}} \cdot \text{collision\_penalty} \tag{1}$$

The sparse success reward is defined as:

$$r_{\text{success}} = \begin{cases} 1.0 & \text{if pos\_err} < \delta \\ 0.0 & \text{otherwise} \end{cases} \tag{2}$$

We experimented with two variants of the collision penalty: a binary version and a continuous Gaussian-based version. The binary penalty assigns a fixed cost when a contact event is detected, serving as a simple indicator of failure. However, it provides no gradient information near collisions, which limits its effectiveness for learning stable policies in cluttered environments.

To address this, we implemented a continuous Gaussian collision penalty based on the distance between the end-effector and the nearest obstacle. This version penalizes proximity more smoothly, with the penalty increasing as the robot approaches a collision. The function is defined as:

$$\text{collision\_penalty} = \exp\left(-\frac{d^2}{2\sigma^2}\right) \tag{3}$$

where d is the minimum distance to an obstacle surface and $\sigma = 0.8$ controls the penalty spread.

The components in the formulae are defined as follows:

- $\text{pos\_err} = \|\mathbf{x}_{\text{ee}} - \mathbf{x}_{\text{target}}\|_2$: Euclidean distance between end-effector and target.
- $\text{rot\_err} = \frac{2 \cdot \arccos(\min(1.0, |q_{\text{ee}} \cdot q_{\text{target}}|))}{\pi}$: Quaternion-based angular misalignment.
- collision_penalty: Penalty based on either binary contact detection or Gaussian proximity-based formulation.

We use empirically selected weights: $w_{\text{pos}} = 1.2$, $w_{\text{rot}} = 0.55$, and $w_{\text{col}} = 1.8$, prioritizing collision-free, goal-directed behavior.

Training is performed in parallel across multiple randomized environments, promoting generalization. Rollouts are collected using a fixed time horizon, and policy updates are computed from minibatches. Training proceeds until the success rate stabilizes and collision frequency decreases significantly.

3.4 Evaluation

We evaluate the learned controller in held-out environments containing obstacle configurations not seen during training. A rollout is considered successful if the robot reaches the target pose within a fixed number of steps without colliding with any obstacles. In addition to success rate, we assess performance using trajectory smoothness, average control effort, and deviation from expert

demonstrations. These metrics provide a more comprehensive view of the controller's stability, efficiency, and alignment with human-like behavior.

Qualitative performance is visualized through rollout videos, which highlight how the robot adjusts its joint configuration and end-effector path in response to spatial constraints.

4 Experimental Setup

Task Description. The objective is to train a whole-body controller that enables a robotic arm to reach a target pose while avoiding collisions in cluttered environments. Each episode begins with a random obstacle layout and a fixed target position. The robot must navigate through the scene using joint-level velocity control to align its end-effector with the goal. Episodes terminate when the goal is reached, a (self or ground) collision occurs, or the maximum step count is exceeded.

Data Collection. As mentioned above, to guide training, we collect a dataset of expert demonstrations using a 3D SpaceMouse and keyboard interface. These expert trajectories consist of timestamped end-effector positions and orientations (in XYZ and quaternion format) and reflect human strategies for spatial reasoning and collision avoidance. In total, we recorded 50 trajectories across 5 distinct scene layouts.

Baselines. We compare our learned PPO policy against two baselines:

- **Naive IK Controller:** A simple inverse kinematics controller that directly follows a straight-line path in task space without considering obstacles.
- **PPO Policy with Collision Avoidance:** A learning-based controller trained without adding in relevant reward terms to enable obstacles avoidance.

These baselines help isolate the value of learned whole-body coordination over manually engineered or previously trained working policies.

Training Setup We train our PPO policy in the MuJoCo simulator using Brax to enable efficient vectorized rollouts over 1024 parallel environments. Detailed training configurations are as follow:

Setting	Value
Parallel Environments	1024
Rollout Length	600 time steps
Batch Size	128
Learning Rate	3×10^{-4}
Discount Factor (γ)	0.99
Reward Shaping	See Section 3.3

Table 1: Summary of PPO training settings.

Evaluation Metrics. We evaluate each method using the following metrics:

- **Success Rate:** Percentage of episodes where the robot reaches the goal without collisions.
- **Trajectory Smoothness:** Measured by the mean squared jerk (change in acceleration) across the trajectory.
- **Energy Efficiency:** Cumulative joint torque or velocity magnitudes, reflecting energy usage.
- **Tracking Error:** Average distance between the learned trajectory and expert demonstration in task space.

Each trained policy is evaluated in 128 held-out environments with unseen obstacle configurations to assess generalization performance. Visual rollouts are also recorded for qualitative analysis.

5 Results

As mentioned, we evaluated our PPO-trained whole-body controller against two baselines in 128 unseen, randomized environments. Results demonstrate that the learned policy overall generalizes well to new obstacle layouts, maintaining decent success rates while exhibiting smooth, collision-averse motion.

5.1 Quantitative Evaluation

Table 2 summarizes the performance across all evaluated policies. Our PPO-based controller achieves a success rate of 56.4% under the binary collision reward, and 58.1% under the Gaussian collision reward, significantly outperforming the naive IK controller (21.1%) and the PPO without Collision Avoidance baseline (24.6%). Across both PPO variants, we observe smoother trajectories—as indicated by lower mean squared jerk—and reduced control effort, suggesting more efficient joint coordination. Additionally, the tracking error metric demonstrates that our learned policies closely align with expert trajectories, even though training was conducted via reinforcement learning rather than direct behavioral cloning.

Table 2: Performance Comparison Across Evaluation Metrics and Baselines

Method	Success Rate (%)	Smoothness ↓	Energy Efficiency ↓	Tracking Error ↓
Naive IK Controller	21.1	0.122	3.98	0.308
PPO Policy without Collision Avoidance	24.6	0.108	3.24	0.172
Our PPO Policy (Binary Collision)	**56.4**	**0.097**	**2.78**	**0.133**
Our PPO Policy (Gaussian Collision)	**58.1**	**0.084**	**2.75**	**0.130**

Among the baselines, the naive IK controller performs poorly in obstacle-dense environments. Since it directly solves for a straight-line trajectory to the target using inverse kinematics without considering obstacles, it frequently results in collisions. Its low success rate and poor smoothness reflect both its lack of spatial reasoning and computational inefficiencies in real-time joint-space solving.

The PPO policy without Obstacle Avoidance baseline performs better, achieving a 24.6% success rate. While it was not explicitly trained for obstacle avoidance, its might have learned motion priors and partial visual input allow for some adaptability. However, due to its limited environment awareness, it often produces erratic joint behavior when navigating tight spaces. This manifests in relatively higher control effort and reduced smoothness, as the policy frequently hesitates in response to obstacles.

In contrast, our PPO-trained controllers demonstrate consistent improvements across all metrics. Training across five procedurally generated scenes and fine-tuning the reward structure yielded policies with stronger generalization. Both binary and Gaussian reward configurations led to relatively more effective goal-reaching behavior, but the Gaussian reward function showed a slight edge. As discussed in Section 3.3, the smooth, continuous penalty from the Gaussian function provides gradient information even in near-collision cases, allowing the agent to learn subtle avoidance behaviors that binary penalties cannot capture. This additional guidance likely helped the policy reduce abrupt corrections, improving both smoothness and success rate.

Overall, our learned policies achieved more consistent success across five unseen environments, more effectively navigating cluttered scenes with obstacle avoidance, reduced joint effort, and strong adherence to expert-like trajectories.

5.2 Qualitative Analysis

As discussed in the quantitative analysis, rollouts using the naive IK controller are generally unsuccessful, brief (often ending early due to obstacle crashes), and exhibit jerky, uncoordinated motion (when crashed or scratched by obstacles). In the rollout videos, this manifests as the robot driving straight into obstacles and then falling over due to failed balance recovery. These behaviors confirm our expectations: the IK controller lacks any obstacle awareness and merely attempts to follow a direct path to the goal, leading to error-prone execution.

The PPO without Collision Avoidance, in contrast, demonstrates some awareness of its environment. In many cases, the agent attempts to navigate through narrow gaps, reflecting its pre-trained motion priors. However, because it was not explicitly trained to perform in cluttered environments, it often

struggles in ambiguous layouts (such as Y-shaped intersections) where it hesitates, producing jittery or indecisive motion. While this Policy without Collision Avoidance offers more competent behavior out of the box compared to naive IK, it lacks the specialized adaptation needed for consistently reliable navigation in dense scenes.

Our trained PPO-based policies show smoother and more stable behavior across a range of obstacle configurations. For example, when following expert-inspired trajectories that would otherwise result in collisions, our controller adjusts by tilting the floating base or reorienting its joints to avoid contact—demonstrating a level of spatial awareness and adaptability not present in the baselines. These policies also handle complex structures like branching corridors more confidently, committing to a path while maintaining stable posture.

However, one limitation observed is the emergence of unrealistic or unsafe joint motions, particularly in tight spaces. In some cases, the robot contorts excessively to avoid penalties, which may not translate safely to real-world hardware. This highlights the need for additional constraints in the reward function, such as an energy penalty term, to regularize motion and prevent over-optimization on avoidance at the cost of physical feasibility.

6 Discussion

This project consists of three core components: (1) implementing a modular MuJoCo environment with Hydra-based scene configuration to support dynamic layout generation, (2) developing a teleoperation server that enables intuitive expert demonstrations using a 3D SpaceMouse and keyboard interface, and (3) training a privileged-observation-based whole-body controller entirely in simulation to perform smooth and efficient obstacle avoidance.

One of the main technical challenges I faced early on was how to flexibly compose different MJCF XML files for the robot, objects, and environment. Initially, I attempted to merge all components into a single monolithic XML file, but this approach was difficult to maintain and not suitable for diverse scene configurations. After studying prior codebases and consulting with PhD students in my lab, I transitioned to using the `dm_control` framework, which allowed me to implement an object-oriented simulation interface. I also integrated Hydra to manage training configurations and enable scalable testing across varied obstacle layouts and robot parameters. This modular simulation setup is extensible and reusable, making it easy for others to quickly build and evaluate new scenarios in MuJoCo.

To support expert data collection, I built a low-latency teleoperation interface using a keyboard and 3D SpaceMouse, allowing for natural, real-time manipulation of the robot in cluttered environments. These demonstrations provide high-quality motion trajectories that not only help shape the reward function but also serve as behavioral references during training. The PPO-based teacher policy, trained using full simulator state, establishes a strong performance baseline that we intend to distill into a student policy with partial observations in future work.

While the current controller performs well in simulation, several limitations remain. Most notably, the policy has not yet been distilled into a student model that operates under realistic, partial observations (e.g., RGB, RGB-D, or point clouds). Furthermore, the simulation lacks domain randomization features critical for sim-to-real transfer. These include modeling sensor and actuator noise, introducing variable damping coefficients, and adding external perturbations or contact forces. Additionally, deployment in real-world environments with complex geometries—like tabletops, chairs, and narrow gaps—requires further robustness evaluation. These improvements will be prioritized in the next phase of development to enhance the generalizability and physical reliability of the controller.

7 Conclusion

This work lays the foundation for a robust, learning-based whole-body controller capable of generalizing across complex terrains and constrained environments. The teacher model training stage not only enables future student policy distillation but also supports future systematic exploration of partial observation strategies for real-world deployment. Additionally, the full pipeline—including a modular, object-oriented MuJoCo simulation environment—provides a reusable infrastructure that supports sim-to-real transfer and serves as a base for broader robotics research moving forward.

However, further work on policy distillation, partial observation representation, and sim-to-real adaptation is needed to make the system more usable.

8 Team Contributions

- **Group Member 1 (only member):** Jeff Liu – all components of the project were completed independently.

Changes from Proposal Due to limited available time during the quarter and the small team size (solo project), the proposed student policy distillation could not be completed before the deadline for this final report (June 8th, 2025).

References

Xuxin Cheng, Kexin Shi, Ananye Agarwal, and Deepak Pathak. 2023. Extreme Parkour with Legged Robots. arXiv:2309.14341 [cs.RO] https://arxiv.org/abs/2309.14341

C. Daniel Freeman, Erik Frey, Anton Raichuk, Sertan Girgin, Igor Mordatch, and Olivier Bachem. 2021. Brax – A Differentiable Physics Engine for Large Scale Rigid Body Simulation. arXiv:2106.13281 [cs.RO] https://arxiv.org/abs/2106.13281

Zipeng Fu, Xuxin Cheng, and Deepak Pathak. 2022. Deep Whole-Body Control: Learning a Unified Policy for Manipulation and Locomotion. arXiv:2210.10044 [cs.RO] https://arxiv.org/abs/2210.10044

Huy Ha, Yihuai Gao, Zipeng Fu, Jie Tan, and Shuran Song. 2024. UMI on Legs: Making Manipulation Policies Mobile with Manipulation-Centric Whole-body Controllers. arXiv:2407.10353 [cs.RO] https://arxiv.org/abs/2407.10353

Tairan He, Wenli Xiao, Toru Lin, Zhengyi Luo, Zhenjia Xu, Zhenyu Jiang, Jan Kautz, Changliu Liu, Guanya Shi, Xiaolong Wang, Linxi Fan, and Yuke Zhu. 2025. HOVER: Versatile Neural Whole-Body Controller for Humanoid Robots. arXiv:2410.21229 [cs.RO] https://arxiv.org/abs/2410.21229

Tianyue Wu, Yeke Chen, Tianyang Chen, Guangyu Zhao, and Fei Gao. 2024. Whole-Body Control Through Narrow Gaps From Pixels To Action. arXiv:2409.00895 [cs.RO] https://arxiv.org/abs/2409.00895

Kevin Zakka, Baruch Tabanpour, Qiayuan Liao, Mustafa Haiderbhai, Samuel Holt, Jing Yuan Luo, Arthur Allshire, Erik Frey, Koushil Sreenath, Lueder A. Kahrs, Carmelo Sferrazza, Yuval Tassa, and Pieter Abbeel. 2025. MuJoCo Playground. arXiv:2502.08844 [cs.RO] https://arxiv.org/abs/2502.08844