Enhancing Explosive Motion in Humanoid Robotics with Imitation and Reinforcement Learning

Master Thesis

submitted to

Institute of Control Theory and Systems Engineering

Faculty of Electrical Engineering and Information Technology

Technische Universität Dortmund

by

Shubhankar Kulkarni

Maharashtra, India

Date of Submission: September 01, 2025

Responsible Professor:
Univ.-Prof. Dr.-Ing. Prof. h.c. Dr. h.c. Torsten Bertram

Academic Supervisors:
Apl. Prof. Dr. Frank Hoffmann
M.Sc. Martin Krüger

"*To the experiences we never expected, and the paths that were redirected. To the friends and family who we found along the way.*"

Acknowledgement

This thesis has been a significant journey in terms of patience, hard work, and dedication. Thinking about this topic and researching it has been a delight, honestly. Receiving constant support from my family and friends has been a great source of motivation. Not only did it encourage me to keep striving toward my goal, but it also provided me with a continuous reminder that I am not alone in this journey.

Learning about patience, perseverance, and hard work is something I gained while working on this thesis. I know it may not seem like much, but these are values that I will carry with me for the rest of my life. I would like to thank my family for their unwavering support and encouragement throughout this process, without which I would not even be where I am today. Their belief in me has been a constant beacon of light, guiding me through the challenges and uncertainties that come with such a significant undertaking. I would also like to thank Apl. Prof. Dr. Frank Hoffmann for his guidance and support throughout this process. His insights and expertise have been invaluable, and I am grateful for the time and effort he invested in helping me refine my ideas and improve my work. I would also like to thank Martin Krüger for his support and guidance throughout this process.

I would like to extend my gratitude to all my friends and my girlfriend, who have been there for me during this journey. On the days when I felt overwhelmed or discouraged, they were always there to lend a listening ear or offer words of encouragement. Having a supportive network has made this journey not only bearable but also enjoyable. I am grateful for the moments of laughter, the shared experiences, and the unwavering belief they have shown in me.

I will be forever grateful that I stumbled upon this topic, which helped shape my understanding of the world and my place in it. It has been a transformative experience that has not only deepened my knowledge but also broadened my perspective. I am excited to see where this journey will take me next, and I look forward to continuing to learn and grow in the years to come.

Abstract

Explosive motion, defined by quick, high-energy movement, is a fundamental aspect of human performance in activities such as sprinting, jumping, and throwing. Such motions are predicated upon the ability to produce and transfer force quickly through coordinated action of numerous joints and muscle activations. For humanoid robots, the ability to perform explosive motion can help to significantly advance their usefulness in tasks that entail quick and forceful motion, such as running, jumping, or throwing.

To apply such movement in humanoid robots, however, is a challenging task that involves biomechanical complexity, optimization of movement, and stable control. Such challenges are addressed by this thesis through the optimization and creation of humanoid motion planning approaches to explosive tasks, or the standing long jump. Traditional MoCap approaches are inclined to have a hard time capturing the subtlety of explosive movements and thus prove to be of limited help in training dynamic behavior. In order to transcend such limitations, this thesis incorporates MoCap data with machine learning techniques for a richer descriptive analysis of explosive motion.

RL is used for optimizing humanoid motion plans so that varied behaviors can be explored and the most efficient ones can be found that can create explosive movements. The experiments are carried out in the MuJoCo simulator, which offers systematic testing and verification within a controlled environment.

The results demonstrate that RL largely increases the accuracy, efficiency, and robustness of humanoid explosive movements. The findings are important to the design and control of humanoid robots in how they are able to execute human-like high-power movements.

Contents

Nomenclature

$\mathbf{c_1}$	Entropy Coefficient
$\mathbf{J}(\pi)$	Cumulative reward
$\mathbf{S}[\pi_\theta]$	Entropy Bonus
$\mathbf{L_t^{CLIP+S}}(\theta)$	Clipping Surrogate Loss with Entropy
$\mathbf{L_t^{CLIP}}(\theta)$	Clipping Suggorate Loss
$\mathbf{L}_\pi$	Actor Loss
$\mathbf{L_V}$	Critic Loss
$\mathbf{W}$	Weight Parameter for Critic

Greek symbols

α	Learning Rate
β	L2 Regularization Factor
δ	Temporal Difference Residual
ϵ	Clipping Ratio
η	Energy Efficiency
γ	Discount factor
κ	Curriculum factor
λ	Generalized Advantage Estimation Parameter
ν	Scale Factor for Control Input
ρ	Step Size
ζ	Entropy Decay Rate

Abbreviations and Acronyms

AMP	Adversarial Motion Priors
BC	Behavior Cloning

CoM	Center of Mass
DTW	Dynamic Time Warping
GAE	Generalized Advantage Estimation
GRF	Ground Reaction Force
IL	Imitation Learning
MDP	Markov Decision Problem
MoCap	Motion Capture
MSE	Mean Squared Error
MuJoCo	Multi Joint dynamics with Contact
PD	Proportional Derivative
PPO	Proximal Policy Optimization
RL	Reinforcement Learning
TD	Temporal Difference
TRPO	Trust Region Policy Optimization

Usage of generative AI models Explanations for the usage of generative AI models and its notation:

The bottommost level at which the identification is presented regarding the possible uses of generative AI models are subchapters of the 2nd order (e.g., 1.1.1, which may also appear without numbering), as otherwise, the identification would disrupt the reading flow due to frequent occurrences. Algorithms used for implementing generative AI models are mentioned at least in the text or provided as pseudo-code to facilitate appropriate identification.

1 Introduction

Humanoid robots have witnessed significant advancements since their inception, with ongoing efforts focused on improving performance, agility, and adaptability. A key challenge in this domain is pushing the boundaries of humanoid capabilities, particularly in terms of dynamic and energy-efficient motion, while maintaining control robustness and reliability in real-world scenarios.

Modeling and simulating humanoid motion provides a powerful approach to address these challenges. However, the complexity of humanoid systems, including high degrees of freedom, underactuation, and nonlinear dynamics, makes accurate modeling non-trivial. Errors in modeling or control design can produce unrealistic or inconsistent simulated behaviors, reducing the applicability of the resulting controllers in physical systems.

In this work, we adopt a learning-based approach to expand the capabilities of humanoid motion control, with a focus on high-energy, explosive movements. At the core of this method is a Reinforcement Learning (RL) framework, where the humanoid agent learns control strategies through trial and error by interacting with the environment and receiving reward feedback. Over time, the agent discovers policies that map state information to effective motor actions.

This study investigates how humanoid robots can learn dynamic behaviors, such as jumping or rapid directional changes, that require precise torque coordination and efficient energy use. Rather than relying solely on reference motion data, we emphasize flexible learning that allows the emergence of novel, optimized behaviors suited to the robot's morphology and the task at hand. By leveraging biomechanical insights, such as ground reaction forces, momentum transfer, and arm-swing mechanics. This framework encourages both performance and human-like motion patterns.

Through this approach, we aim to develop humanoid controllers that are more capable, adaptable, and energy-efficient, enabling effective operation in complex and dynamic environments.

1.1. Motivation

Explosive motion refers to rapid, high-energy behaviors such as jumping, sprinting, kicking, and sudden changes in direction. Such motions are fundamental to human physical capability and are critical in diverse domains, ranging from athletic performance and competitive sports to emergency response, military operations, and complex manipulation tasks. Achieving this requires both efficient force production and precise temporal and spatial coordination across multiple joints and limbs. Replicating these behaviors in humanoid robots represents a major milestone, signaling progress toward greater agility, responsiveness, and practical utility in unstructured environments.

Designing controllers capable of producing explosive motion remains a fundamental challenge due to hardware limitations, such as torque saturation and motor latency, as well as safety constraints and inherent instability in high-speed maneuvers.

Traditional motion imitation approaches rely heavily on Motion Capture (MoCap) data as a reference. While MoCap-guided imitation ensures realistic motion, it often constrains the diversity of behaviors the agent can explore. This can lead to overly conservative and repetitive motions, limiting the policy's ability to generalize to novel situations or adapt to task variations. Directly tracking reference trajectories may also prevent the emergence of more efficient or robust solutions that deviate from expert demonstrations.

Enabling a humanoid to learn explosive behaviors beyond simple imitation allows it to discover novel strategies for force generation, trajectory optimization, and energy-efficient execution. Learning-based frameworks that incorporate feedback, adaptability, and reward-driven optimization can enable robots to outperform the original MoCap reference by leveraging their physical embodiment more effectively.

This leads to two core research questions that guide this work:

- How can RL and Imitation Learning (IL) be leveraged to develop effective strategies for explosive motion in humanoid robots?
- What methods can be used to stimulate and optimize high-energy motion patterns in simulated humanoid environments?

Integrating RL techniques enables policies to be optimized not only for imitation fidelity, but also for robustness, versatility, and task success under varying environmental and physical conditions. This holistic learning approach allows humanoids to dynamically adapt explosive motions to new tasks, terrains, or constraints, resulting in more intelligent and capable robotic agents.

1.2. Thesis Overview

This thesis investigates the application of RL to develop control strategies that improve upon locomotion behaviors derived from MoCap data. Specifically, we focus on enhancing the quality and realism of explosive motions, such as a standing long jump performed by simulated humanoid agents.

Chapter 2 reviews foundational work in RL for motor control and its evolution into IL techniques. This chapter highlights the motivations for using RL in physics-based animation and discusses the benefits of leveraging MoCap data to guide humanoid behavior through imitation.

Chapter 3 introduces the theoretical and algorithmic background necessary to understand this methods. We explain the principles of Proximal Policy Optimization (PPO), the RL algorithm employed in this work, and describe reward design and policy architecture components that underpin successful learning in physics-based simulations.

Chapter 4 details the experimental framework, including the humanoid model, MoCap preprocessing pipeline, MuJoCo environment configuration, and training setup. Special emphasis is placed on state-action representation, policy network design, and simulation parameters that influence training stability and performance.

Chapter 5 presents the core contribution: a method for enhancing explosive motions using a refined reward function and imitation-guided RL policy. We introduce a reward shaping scheme tailored to promote powerful, accurate, and biomechanically plausible jumping behaviors. The policy is trained to match reference trajectories while optimizing for improved outcomes, such as increased jump distance and better landing stability.

Chapter 6 evaluates the learned policy qualitatively and quantitatively. We analyze the effects of the reward formulation, compare generated motions to ground-truth MoCap data, and assess physical realism, coordination, and motion fidelity. This chapter also evaluates generalizability to other motion types and humanoid models and examines factors affecting policy transfer and robustness.

Chapter 7 presents a summary of key improvements in jumping motion and outlines potential directions for future work, including enhancements to the current framework and strategies to further improve the quality and robustness of explosive motions.

Overall, this thesis demonstrates how RL, augmented with imitation learning and reward engineering, can be used not only to replicate, but also to enhance complex human-like behaviors. The findings indicate that this framework holds promise for advancing realistic character animation, robotic control, and biomechanics-informed motion synthesis.

2 Related Work

This chapter surveys the body of research most relevant to present study, encompassing physically plausible motion synthesis, RL for humanoid agents, and the biomechanics of explosive motions such as standing long jump. In particular, we review methods for motion imitation using MoCap data, that integrate physics-based simulation and learning-based control to achieve naturalistic humanoid behavior. Special attention is given to RL techniques, most notably PPO and their application to high-dimensional control problems.

Beyond core imitation frameworks, we examine reward shaping strategies aimed at improving both task performance and stylistic fidelity, with studies highlighting the impact of pose, velocity, end-effector, and Center of Mass (CoM) tracking, as well as auxiliary terms such as symmetry, energy efficiency, and take-off dynamics. Finally, we draw upon the field of biomechanics to contextualize explosive motion generation. Research on human jumping mechanics provides insight into key kinematic and kinetic phases: Squat, Take-off, Flight, and Landing. This informs both reward design and policy evaluation in the current work. By synthesizing these perspectives, this chapter establishes the theoretical and methodological context in which the proposed framework is situated, identifies the limitations in existing literature that this study assesses.

2.1. Physics-Based Humanoid Motion Imitation

Physics-based motion imitation has emerged as a central technique in developing humanoid agents capable of performing realistic movements. Early approaches relied on kinematic imitation, but these often lacked robustness to dynamic perturbations and environmental interactions. A breakthrough came with DeepMimic Peng, Abbeel, et al. (2018), which integrates MoCap sequences into a RL framework. In DeepMimic, a humanoid agent receives a reward that encourages it to closely track the reference motion while simultaneously adapting to external perturbations and follow a goal at the same time. This method allows the agent to reproduce human motion while recovering from disturbances such as pushes or slips, demonstrating the advantage of combining imitation with physically plausible control.

Subsequent works extended the imitation paradigm using Adversarial Motion Priors (AMP) Peng, Ma, et al. (2021). This employs a discriminator network that distinguishes between real MoCap data and simulated behavior. This adversarial approach allows agents to learn from diverse motion datasets, promoting natural transitions between different motions. HumanMimic Tang et al. (2024) builds on this idea with Wasserstein adversarial IL, enabling humanoids to perform natural locomotion and transitions while emphasizing smoothness and continuity.

Other hybrid approaches such as MIMO Miller et al. (2023), combine model-based optimal control with learning-based imitation. MIMOC integrates a physics informed dynamics model with motion priors to produce agile and robust behaviors that are less sensitive to modelling errors. Similarly, frameworks like I-CTRL Yan et al. (2025) uses constrained RL to ensure that retargeted human motions are physically feasible on humanoid robots. Recent advances also explore diffusion-based models Pearce et al. (2023) for imitating human behavior, capturing stochastic and multimodal distributions, and cross-embodiment skill transfer Liu et al. (2024) to generalize behavior from humans to robots with different morphologies.

The emergence of benchmarks such as LocoMuJoCo Al-Hafez et al. (2023) provides standardized datasets and evaluation protocols for locomotion and imitation tasks. These resources facilitate rigorous comparisons across IL methods and promote progress in physics-based character control.

2.2. Reinforcement Learning

High dimensional humanoid control is challenging due to the large action space and unstable dynamics of bipedal agents. PPO Schulman et al. (2017) is widely adopted in this context because it balances sample efficiency with training stability, constraining policy updates to remain within a trust region and preventing catastrophic performance drops.

Studies in locomotion have shown that purely RL based approaches can produce natural walking bahaviors without explicit trajectory tracking. Heess et al. (2017) trained humanoids to walk and run using raw proprioceptive inputs, enabling generalization to novel terrains. Similarly, combining AMP with PPO results in rich human-like behaviors, including jumps, flips, and rolls(Merel et al. (2017)).

Heirarchical RL has been used to decompose complex tasks into simpler sub-tasks. Yang et al. (2020) proposed a multi-expert framework where specialized policies handle different motion primitives, coordinated by a high-level policy. Other recent studies explore constrained and adversarial RL methods Yan et al. (2025), Tang et al. (2024), providing stability and improved realism when learning dynamic skills, such as jumping or locomotion under physical constraints.

2.3. Reward Shaping

The design of reward functions is crucial in achieving high-quality motion imitation. Standard rewards in Deepmimic style frameworks combine pose tracking, velocity matching, end-effector alignment and CoM regulation. Penalizing deviations in CoM ensures upright posture and prevents tipping during dynamic maneuvers.

Recent works incorporate biomechanically inspired reward terms such as energy efficiency, symmetry, and phase-aware motion tracking. Phase-aware rewards, in particular, guide the agent to execute take-off, flight, and landing phases with proper timing, improving performance for ballistic movements like jumps. Advanced imitation frameworks, including diffusion-based models, have demonstrated the potential to further refine structures by modeling stochastic and multimodal action distributions Tang et al. (2024).

2.4. Biomechanics of the Standing Long Jump

A biomechanical perspective provides critical insights into the dynamics of explosive human movements. The standing long jump as been extensively analyzed to understand how joint coordination, ground reaction forces, and limb sequencing contribute to performance Baldwin et al. (2009).

A 3D kinetic and kinematic analyses was conducted in Wen-lanwu et al. (2012), showing that proper utilization of ground reaction forces and momentum transfer from the legs significantly enhances jump distance. The emphasis the role of arm swing and take-off velocity in increasing horizontal displacement was done in Ashby and Heegaard (2002). These studies inform the decomposition of jumps in distance phases: Squat, Take-off, Flight, and Landing. This can be incorporated into reward functions to encourage human-like and landing stability.

2.5. Research Gaps

Despite substantial progress in motion imitation, RL-based humanoid control, and robustness techniques, few studies explicitly address explosive, ballistic movements such as the standing long jump. Existing methods often focus on locomotion or generic motion sequencing without optimizing for biomechanical performance.

This work addresses these gaps by:

- Integrating biomechanically grounded reward functions that explicitly model jump phases and propulsion mechanics.
- Evaluate policy robustness under mass and torque perturbations, capturing real world variability in agent morphology and actuation.

- Combining motion imitation and RL to produce humanoid agents that are both realistic in motion and effective in performing explosive tasks.

This combination of biomechanics-informed design, robustness training, and motion imitation represents a novel contribution in the domain of simulated humanoid control.

2.6. Summary

This chapter reviewed literature on physics-based motion imitation, RL for humanoid control, reward shaping, and biomechanical insights for explosive movements. This review situates this work within the broader context of humanoid simulation research. The following chapters presents the background, methodology of our approach, detailing environment setup, policy architecture, reward design, training procedures tailored to the standing long jump.

3 Background

3.1. Introduction to Reinforcement Learning

RL is a framework for decision-making in which agent interacts with the environment to learn optimal behaviors through trial and error. An agent receives feedback in the form of states and rewards and uses this to improve its decision-making strategy over time. The RL problem is formalized as a Markov Decision Problem (MDP), defined by the tuple $(\mathbf{S}, \mathbf{A}, p, \mathbf{R}, \gamma)$, where $\mathbf{S_t}$ is the set of possible states the agent can occupy, $\mathbf{A_t}$ is the set of possible actions the agent can take, $p(s'|s,a)$ is the transition probability function which calculates the dynamics, $r(s,a)$ is the reward function, $\gamma \in [0,1)$ is the discount factor for future rewards [Sutton and Barto (2014)].

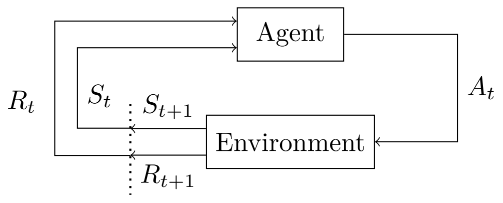


Figure 3.1.: Basic RL Workflow

At each timestep t, the agent observes a state $s_t \in S_t$, takes an action $a_t \in A_t$ according to a policy $\pi(a_t|s_t)$, receives a reward r_t, and transitions to a new state s_{t+1}. The discount factor is majorly used for disregarding rewards in the distant future in comparison to the immediate future. The goal is to learn the policy π that maximizes the expected cumulative reward, known as the return:

$$\mathbf{J}(\pi) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{T} \gamma^t r_t\right]$$

where T is the length of the horizon.

RL algorithms typically operate using either deterministic or stochastic policies. A deterministic policy maps each state to a specific action, consistently selecting the same action every time the state is encountered. This can simplify learning and improve stability, but may limit exploration and adaptability in complex or uncertain environments.

In contrast, a stochastic policy maps each state to a probability distribution over possible actions. At each timestep, the policy samples an action from this distribution, introducing variability into the behavior. This stochasticity promotes exploration, helps the agent discover diverse strategies, and can be particularly advantageous in environments with noise or partial observability.

Stochastic policies are often preferred in continuous control settings, such as humanoid motion, where diverse behaviors and smooth exploration are essential for learning robust and generalizable control strategies. We use stochastic policies during the training of the algorithm and deterministic policies during evaluation.

3.2. Value Functions

Value functions are used to provide an estimate on the performance of the policy stated by cumulative rewards. The state value function, denoted by $V(s)$ measures the long-term expected reward associated with being in state s. This is given by:

$$V(s) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{T} \gamma^t R_t | S_t = s\right]$$

Appropriately, the action value function, denoted by $Q(s, a)$ measures the long-term expected reward associated with performing the action a after being in the state s. This is given by:

$$Q(s, a) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{T} \gamma^t R_t | S_t = s, A_t = a\right]$$

The state value function can be used to represent action value functions, which is given by:

$$Q(s, a) = \mathbb{E}\left[R_{t+1} + \gamma V(S_{t+1} | S_t = s, At = a)\right]$$

These value functions can be estimated from experience through various learning algorithms. One of the most widely used and effective approaches is Temporal Difference (TD) Learning, which updates value estimates by bootstrapping from subsequent estimates.

3.3. TD Learning

TD Learning is a class of model-free RL methods that learn value functions directly from experience by bootstrapping, updating estimates based on other learned estimates. Unlike Monte Carlo methods, which wait until the end of an episode to compute returns, TD methods update value estimates at each time step. The simplest form, TD(0), updates the state value function using the following update rule:

$$V(s_t) \leftarrow V(s_t) + \rho \left[r_t + \gamma V(s_{t+1}) - V(s_t)\right]$$

where $\rho < 1$ is the step size. Here, the quantity:

$$\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

is known as the **TD error**, and represents the discrepancy between the estimated value and the observed return. It drives the learning update. In TD learning, the target for updating $V(s_t)$ is the one-step bootstrapped return:

$$\delta_t = r_t + \gamma V(s_{t+1})$$

This contrasts with Monte Carlo targets, which are based on complete returns from a full episode. TD learning can be extended to account for multi-step returns using a trace-decay parameter λ, leading to the TD(λ) algorithm. It combines n-step updates into a single estimate, balancing bias and variance. Eligibility traces keep a memory of past states to distribute the TD error backward over time.

The update with eligibility trace $e(s)$ is:

$$V(s) \leftarrow V(s) + \rho \delta_t e(s)$$

where $e(s)$ decays over time and is incremented when state s is visited. In actor-critic algorithms, TD learning is often used to update the critic's value function, while the actor updates the policy using advantage estimates derived from the TD error or Generalized Advantage Estimation (GAE). The algorithm used can be found at Algorithm 3.3.1

Actor-critic methods consist of two components: the actor, which updates the policy parameters in the direction suggested by the policy gradient, and the critic, which evaluates the current policy by estimating the value function. The critic provides feedback (advantages) to the actor to improve learning efficiency and stability.

The advantage estimates computed through TD learning with GAE serve as the foundation for updating the policy in actor-critic methods. By reducing variance and maintaining bias-efficiency tradeoffs, GAE improves the stability of policy updates. This leads to the class of *Policy Gradient Methods*, where the policy is directly optimized to maximize expected return.

Algorithm 3.3.1.: TD Learning with GAE

```
1: Initialize value function $V(\theta(s))$
2: for each episode do $\delta_t = r_t + \gamma(V(s_{t+1}) - V(s_t))$
3:     Initialize advantage estimate $\hat{A}_T = 0$
4:     for each timestep $t = T-1, T-2, \dots, 0$ do
5:         Compute GAE recursively:
               $$\hat{A}_t = \delta_t + \gamma\lambda\hat{A}_{t+1}$$
6:     Compute Value Targets: $\hat{V}_t = \hat{A}_t + V(s_t)$
7:     Update Value function by minimizing:
               $$\mathbf{L_V} = \frac{1}{T}\sum_{t=0}^{T-1}(V(s_t) - \hat{V}_t)^2$$
```

3.4. Policy Gradient Methods

Traditional value-based methods like Q-learning optimize policies indirectly by estimating value functions and then extracting the optimal policy from them. In contrast, **policy gradient methods** directly optimize the policy $\pi_\theta(a|s)$ by maximizing the expected cumulative return.

The goal is to find the optimal policy parameters θ that maximize the following objective:

$$J(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{\infty}\gamma^t r_t\right]$$

Using the **Policy Gradient Theorem**, the gradient of the objective can be written as:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(a_t|s_t) \cdot \hat{A}_t\right]$$

Here, $\hat{A}_t$ is an estimator of the advantage function, which measures how much better an action is compared to the average action in state s_t.

This formulation allows the use of stochastic gradient ascent to update policy parameters in the direction that improves performance. However, the variance of the gradient estimates can be high. As introduced in previous sections, methods like **TD learning** and **GAE** are used to reduce this variance by efficiently estimating $\hat{A}_t$.

Despite these improvements, basic policy gradient methods such as REINFORCE suffer from high variance, unstable training, and poor sample efficiency. These limitations motivated the development of actor-critic architectures and constrained policy optimization methods like <u>T</u>rust <u>R</u>egion <u>P</u>olicy <u>O</u>ptimization (TRPO).

PPO is a modern, simplified alternative to TRPO that maintains stable updates by clipping the policy objective to prevent large policy changes. It balances the benefits of performance improvement and training stability in a computationally efficient way.

3.5. Proximal Policy Optimization

As given in Schulman et al. (2017), PPO is a widely used on-policy RL algorithm that improves the stability and performance of policy gradient methods. It is part of the broader actor-critic family, where the actor (policy network) selects actions, and the critic (value network) estimates state values.

PPO builds upon the policy gradient framework, using the following elements introduced previously:

- **Policy Gradient Theorem:** Updates are made using gradients of the expected return with respect to policy parameters.
- **TD Learning and Value Functions:** The critic approximates the state value function $V(s)$ using TD learning to serve as a baseline.
- **Advantage Estimation via GAE:** Advantage estimates $\hat{A}_t$ are computed using GAE to reduce variance while preserving useful learning signals.

Instead of directly maximizing the expected return, PPO defines a **clipped surrogate objective** to constrain policy updates:

$$\mathbf{L_t^{CLIP}}(\theta) = \mathbb{E}_t \left[\min \left(r_t(\theta)\hat{A}_t,\ \text{clip}(r_t(\theta),\ 1-\epsilon,\ 1+\epsilon)\hat{A}_t \right) \right]$$

where:

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$$

This ratio measures how much the current policy has deviated from the previous one. The clipping prevents large updates that could destabilize learning.

A PPO algorithm that uses fixed-length trajectory segments is shown below. Each iteration, each of N (parallel) actors collect T timesteps of data. Then we construct the surrogate loss on these NT timesteps of data, and optimize it with minibatch Adam (or usually for better performance [Kingma and Ba (2017)]), for K epochs [Algorithm 3.5.1]

PPO often includes an entropy term in the loss to encourage exploration by preventing the policy from becoming too deterministic too early. This promotes diverse behaviors, which is especially useful in continuous control tasks like humanoid motion.

PPO strikes a balance between learning speed and stability, making it one of the most popular algorithms in deep RL. Its simple implementation and consistent performance across

Algorithm 3.5.1.: Base Proximal Policy Optimization (Actor-Critic Style)

```
1: Initialize policy parameters $\theta_0$ and old policy $\theta_{\text{old}} \leftarrow \theta_0$
2: for iteration = 1, 2, ... do
3:     for actor = 1, 2, ..., $N$ do
4:         Run policy $\pi_{\theta_{\text{old}}}$ in environment for $T$ timesteps
5:         Collect trajectories $\{(s_t, a_t, r_t, s_{t+1})\}_{t=1}^{T}$
6:         Compute advantage estimates using GAE
7:     for $K$ epochs do
8:         for each minibatch of size $M \leq N \cdot T$ do
9:             Compute probability ratio $r_t$
10:            Compute clipped surrogate loss
11:            Perform gradient ascent on surrogate loss.
12:            Optionally update value function and entropy bonus
13:     Update old policy parameters: $\theta_{\text{old}} \leftarrow \theta$
```

a wide range of environments make it a strong choice for continuous control tasks such as humanoid motion imitation. The detailed algorithm can be found at Algorithm 4.6.1

The next section presents the experimental setup used to train humanoid agents. The details of the humanoid model, the state and action space, the MoCap data preprocessing, and the MuJoCo environment configuration are discussed.

4 Experimental Setup

4.1. Research Methodology

To this end, we propose a RL based framework in which a humanoid agent operates within a barrier-free environment. The agent is initially provided with a reference motion using IL, to bootstrap the training process. This expert trajectory ensures biomechanical plausibility in early stages of learning. As training progresses, RL takes precedence, allowing the humanoid to explore and optimize its motion strategies for greater explosiveness and efficiency.

To stimulate and refine high-energy motion patterns, we design a reward structure that encourages long-distance jumps, rapid execution, and post-landing stability. Additionally, we incorporate curriculum learning to gradually shift the learning focus from imitation to performance, enabling more robust and generalizable motion behaviors in complex simulated settings.

4.2. Choice of Learning Algorithm

Given the simplicity of the task-mimicking a single jumping motion, we opted not to use IL algorithms such as Behavior Cloning (BC) or adversarial methods like AMP. While BC provides a straightforward way to imitate expert demonstrations, it tends to perform poorly in long-horizon tasks and suffers from compounding errors when faced with out-of-distribution states. However, in this case, the primary limitation was not model performance, but rather that the jumping motion itself did not require the representational or policy complexity that IL methods are typically designed to handle.

AMP, in particular, is well-suited for complex scenarios involving multiple skills and transitions between motions. Since this work focuses solely on a single, cyclic motion, the advantages of AMP were unnecessary and added additional computational and implementation overhead without proportional benefit.

Instead, we rely entirely on RL with a well-shaped reward function to guide the humanoid agent. This approach is sufficient to reproduce and improve upon the reference motion, while also allowing for enhancements such as improved jump height, distance, and landing stability.

Reward shaping plays a central role in this approach, as it enables not only faithful imitation of the reference MoCap trajectory but also systematic enhancement of the motion based on task-specific objectives. As detailed in Section 5.3, the reward function is designed to go beyond simple replication, encouraging improvements in jump distance, takeoff dynamics, symmetry, and post-landing stability. This makes RL particularly well-suited to this setting, as it allows the agent to optimize behavior through interaction, rather than relying solely on supervised imitation of demonstrations.

4.3. Simulator

For simulating RL in a humanoid, it is critical to select a physics simulator that accurately models both the dynamics and the contact forces acting on a high-degree-of-freedom agent. For the purpose of solving the problem, we adopt the Multi Joint dynamics with Contact (MuJoCo) physics engine as the simulator of choice. MuJoCo is specifically designed for simulating articulated mechanisms and offers several advantages:

- It is optimized for simulating complex robotic systems, including humanoid agents.
- It provides high-fidelity dynamics simulation with stable contact handling, which is critical for locomotion and explosive motion tasks.
- MuJoCo supports a wide range of prebuilt humanoid models and allows for fully custom XML(model file type)-based model design.
- It enables the creation of custom environments tailored to specific research tasks.
- It is computationally efficient and runs primarily on the CPU, which allows for faster simulation and training cycles even on machines with modest graphics hardware.

4.4. Humanoid Model and Environment

The custom simulation environment contains a humanoid model with 28 degrees of freedom, representing the key joints required for bipedal locomotion and explosive movement. The model has a total mass of approximately 44 kg and includes the hips, knees, ankles, shoulders, elbows, and spine joints, while excluding extraneous joints such as the fingers.

The MuJoCo environment provides full access to the joint positions, velocities, accelerations and applied torques. These are utilized both as observations for the learning agent and for computing the reward function. The root link (pelvis) position and velocity are also accessible and are used to compute jump height and distance. The details of the humanoid model, including joint limits and torque constraints, are specified in Table A.2,

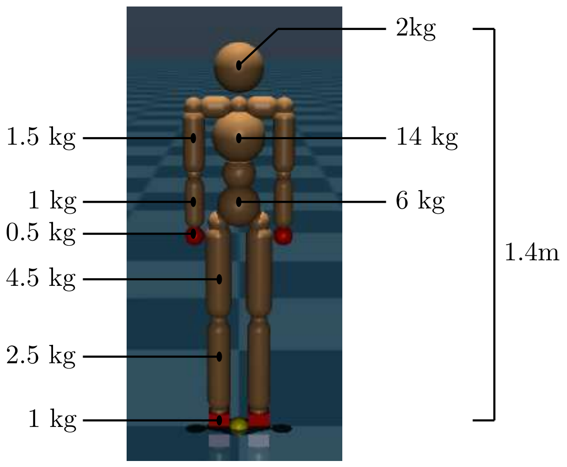


Figure 4.1.: Labeled humanoid showing link masses

Table 4.1.: Humanoid Model Specifications

Specification	Value
Total Mass	44.0kg
Height	1.4m
Number of Joints	28
State Space	132

Table A.3, and Table 4.1.

The simulation is run at a timestep of 0.003 s (i.e., 67 Hz). 5 frames are being skipped during evaluation every time to increase computation times. All physical quantities such as joint angles, velocities, and root orientation are normalized where appropriate before being passed into the policy network.

The environment also supports flexibility in configuration, allowing easy modification of:

- The morphology of the humanoid model (e.g., adjusting segment lengths, mass distribution).
- Torque limits and joint constraints.
- The reward function components and weights.

These features are particularly useful for ablative studies and robustness evaluations, discussed in later sections.

4.5. Task Description

The objective of this experiment is to enhance **explosive motions** derived from human MoCap data. Specifically, we focus on a **standing long jump** as the target behavior. This movement involves transitioning from a squat position into a powerful leap and then landing in a controlled manner. The goal is to not only imitate the provided MoCap trajectory but to improve upon it by increasing the jump distance while maintaining biomechanical plausibility.

4.5.1. Motion Capture Reference

The MoCap data used in this study captures a human subject performing a standing long jump. To make the data compatible with the simulated humanoid, it is retargeted using a standard joint mapping procedure that aligns the degrees of freedom between the MoCap skeleton and the humanoid model.

During training, the agent is provided with a time-indexed reference trajectory that includes target joint angles, end-effector positions (e.g., hands and feet), and the CoM trajectory. These reference signals serve as motion targets and are used to guide the learning process through a carefully designed reward function (see Section 5.3). The agent learns to imitate the reference motion while adapting to the physical constraints and dynamics of the simulation environment.

4.5.2. Observation and Action Spaces

At each timestep, the policy receives an observation vector comprising the following components:

- **Relative joint angles** with respect to the root (pelvis), providing posture information.
- **Joint angular velocities**, offering insight into joint dynamics.
- **Root link (pelvis) linear and angular velocity**, capturing the global motion of the agent.
- **Orientation of the root**, represented as a unit quaternion to preserve smooth rotational continuity.
- **Phase variable**, indicating the normalized progression through the reference motion clip, serving as a weak form of temporal guidance.

This state vector encodes sufficient information for the policy to infer both pose imitation and physical objectives like balance, take-off force, and post-landing stability.

The agent produces continuous-valued actions corresponding to either target joint angles or direct torques, depending on the control mode employed. In the setup, we use a Proportional Derivative (PD) controller to map target joint angles to joint torques. Given a desired joint angle q^i and velocity $\dot{q}^i$, and current values q_i and $\dot{q}_i$, the torque applied is:

$$\tau^i = k_p \cdot (q^i - q_i) + k_d \cdot (\dot{q}^i - \dot{q}_i)$$

Values for k_p and k_d were empirically tuned for each joint group.

Table 4.2.: KP KD Parameters for Joints

Joint name	k_p	k_d	scale
chest	100	100	1
neck	100	10	1
right shoulder	400	40	1
right elbow	300	30	5
left shoulder	400	40	1
left elbow	300	30	5
right hip	300	30	5
right knee	300	30	5
right ankle	200	20	5
left hip	300	30	5
left knee	300	30	5
left ankle	200	20	5

The joints listed in Table 4.2 were further tuned based on their functional role in the motion. Specifically, the *right knee*, *right hip*, *left knee*, and *left hip* joints were assigned k_p and k_d values that were scaled down by a factor of 0.5, relative to their default values. This reduction was introduced to allow for greater compliance and flexibility in the lower limbs during flight phase for motivating a longer flight path for the humanoid.

For the remaining joints, the derivative gain k_d was increased by a factor of 2.0 to provide enhanced damping, which helps stabilize joint movements and reduces oscillations during rapid transitions.

The desired joint angles q^i_{des} were computed using the base pose and the scaled control input as follows:

$$q^i_{\text{des}} = {q^i}_0 + q_i \cdot \nu$$

After computing the desired joint angle, the joint torques were applied using a PD controller. Since the control actions $a \in [-1, 1]$ are normalized, the resulting torques were scaled accordingly to ensure physical feasibility and consistency with actuator limits.

4.6. Training Algorithm

We employ the PPO algorithm within an actor-critic framework to train the control policy. PPO is well-suited for continuous control tasks and offers a favorable trade-off between sample efficiency and training stability.

Both the actor (policy) and critic (value function) networks are implemented as fully connected multilayer perceptrons. The architecture consists of two hidden layers with 512 and 256 units, respectively. The actor network uses ReLU activations to model nonlinear decision boundaries, while the critic network applies a sigmoid activation to bound value estimates, which empirically improves training stability in the setup.

Training proceeds in mini-batches using on-policy rollouts. Although PPO is fundamentally on-policy, we use a temporary replay buffer with a fixed size equal to the number of steps per update (2048), allowing controlled reuse of transition data within a single PPO update cycle. This approach preserves the on-policy nature of the algorithm while improving data efficiency.

To further stabilize learning and enhance exploration, we incorporate the following techniques:

- **Gradient clipping** to prevent exploding gradients and ensure smooth optimization.
- **Entropy regularization** to encourage policy stochasticity, particularly during early training stages.
- **State normalization** to ensure input features remain within a stable dynamic range.
- **Adaptive learning rate scheduling** to dynamically adjust the learning rate and accelerate convergence.

At the beginning of each episode, we reset the environment to obtain the initial state of the humanoid agent. Unlike traditional IL approaches that start from the first frame of the MoCap sequence, we initialize from a default, task-agnostic posture. Further discussion on this choice can be found in Section 6.

The initial state is passed to the policy network, which outputs a stochastic action $a \in [-1, 1]$. This action is then scaled according to the humanoid's actuation limits to conform to the environment's control space.

The scaled action is applied via the environments `step` function, which returns the next state, a scalar reward, and two Boolean flags: `done` and `truncated`, indicating whether the episode has terminated or been truncated due to constraints.

Each transition tuple, composed of `(state, action, reward, next state, done)`, is stored in the replay buffer. The trajectory length is incremented at each step, and once it reaches a fixed horizon of 2048 steps, the actor and critic networks are updated using the collected data. The learning update follows the PPO procedure described in Algorithm 4.6.1.

Algorithm 4.6.1.: Proximal Policy Optimization (PPO)

```
1: Initialize actor and critic networks with parameters $\theta_\pi$ and $\theta_V$
2: for each iteration do
3:     for each environment step until horizon $T = 2048$ do
4:         Observe current state $s_t$
5:         Sample stochastic action $a_t \sim \pi_{\theta_\pi}(a_t|s_t)$
6:         Scale $a_t$ to match action space
7:         Execute action, receive $s_{t+1}, r_t, \texttt{done}_t, \texttt{truncated}_t$
8:         Store $(s_t, a_t, r_t, s_{t+1}, \texttt{done}_t)$ in buffer
9:         T += 1
10:    for $T = 2048$ do
11:        Compute value estimates $V(s_t), V(s_{t+1})$ using critic
12:        $\delta_t = r_t + \gamma V(s_{t+1}) \cdot (1 - \texttt{dw}_t) - V(s_t)$
13:        Initialize advantage buffer $\hat{A}_T = 0$
14:        for each timestep $t = T-1, T-2, , \ldots, 0$ do
15:            $\hat{A}_t = \delta_t + \gamma\lambda(1 - \texttt{done}_t) \cdot \hat{A}_{t+1}$
16:        $\hat{A} \leftarrow \frac{\hat{A} - \text{mean}(\hat{A})}{\text{std}(\hat{A}) + 0.0001}$
17:        Compute TD Targets: $\hat{V}_t = \hat{A}_t + V(s_t)$
18:        for each epoch do
19:            Shuffle the dataset
20:            for each actor mini batch do
21:                Evaluate current policy: $\pi_{\theta_\pi}(a|s)$
22:                Compute Log probabilities $\log \pi_{\theta_\pi}(a)$ and entropy
23:                $r_t = \frac{\pi_{\theta_\pi}(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$
24:                $\mathbf{L_t^{CLIP+S}}(\theta) = \min(r_t\hat{A}_t, clip(r_t, 1-\epsilon, 1+\epsilon)\hat{A}_t)$
25:                $\mathbf{L}_\pi = -\mathbb{E}\left[\mathbf{L_t^{CLIP}}(\theta) + \mathbf{c_1} \cdot \mathbf{S}[\pi_\theta]\right]$
26:                Update Actor via gradient descent
27:            for each critic mini batch do
28:                $\mathbf{L_V} = \mathbb{E}\left[(V(s_t) - \hat{V}_t)^2\right] \beta \cdot \|\mathbf{W}\|^2$
29:                Update Critic via gradient descent
30:        Decay entropy coefficient: $\mathbf{c_1} \leftarrow \mathbf{c_1} \cdot \zeta$
```

The full list of training hyperparameters is provided in Appendix A.1.

4.7. Episode Termination

Episodes terminate under the following conditions:

- The humanoid falls or loses balance (e.g., body parts other than the feet contact the ground).
- The humanoid completes the jump and lands in a stable upright configuration.
- A maximum timestep limit (140 timesteps) is reached.

4.8. Evaluation Metrics

To assess the performance of the trained agent, we track the following metrics:

- **Jump Distance**: Horizontal distance traveled by the center of mass between take-off and landing.
- **Tracking Accuracy**: Mean squared error between reference and executed joint trajectories.
- **Pose Stability**: Root orientation and velocity at landing.
- **Reward Value**: Average return over test episodes.

4.9. Summary

This experimental setup provides a robust framework for simulating and enhancing explosive motions using MuJoCo and RL. The flexibility of the MuJoCo engine, combined with a structured reward function and a MoCap-driven imitation strategy, enables effective learning of dynamic motions such as the standing long jump.

5 Enhancing Explosive Motion

5.1. Overview

In this chapter, we describe the methodology used to enhance explosive motions in a simulated humanoid using RL. Building upon the simulation environment detiled in the Experimental Setup (Chapter 4), we focus on designing motion representations, control strategies, and reward structures tailored to improve the standing long jump, a prototypical explosive motion.

The primary objective is to extend and optimize the MoCap-based jumping behavior, enabling the humanoid to achieve greater distances while maintaining biomechanical plausibility and postural stability. We present the formulation of the task in terms of states, actions, and rewards, each chosen to support learning of improved motion trajectories while preserving key characteristics of the original jump.

5.2. Task Description: Standing Long Jump

The standing long jump is a dynamic, explosive movement that begins from a stationary stance and ends in a stable landing posture. It is composed of four key motion phases:

- **Preparation Phase:** The humanoid lowers its CoM by bending the knees and hips to build elastic energy.
- **Take-off Phase:** The humanoid extends its legs and arms explosively to generate upward and forward momentum.
- **Flight Phase:** The humanoid remains airborne while maintaining symmetry and alignment for optimized trajectory.
- **Landing Phase:** The humanoid re-contacts the ground, absorbing impact and recovering posture.

For a visual description, refer to Figure 5.1

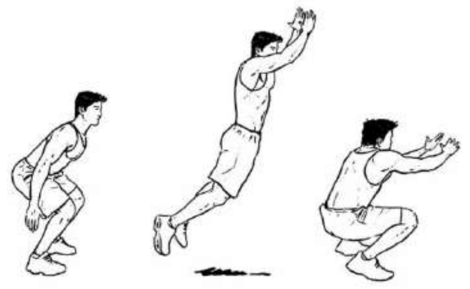


Figure 5.1.: Standing Long Jump (Pintrest n.d.)

This simulation emulates these phases, using MoCap data as a prior for imitation and reinforcement to improve specific motion attributes such as jump distance, height, and symmetry.

5.3. Reward Function

The reward function comprises two components: a motion imitation term to guide the agent toward mimicking reference motion and a task-specific term that promotes further improvement in explosive jump dynamics.

5.3.1. Motion Tracking Reward

To ensure faithful replication of mocap data during the early stages of training, we use a weighted combination of similarity metrics as given in Peng, Abbeel, et al. (2018):

$$r_{im} = w_{pose} \cdot r_{pose} + w_{vel} \cdot r_{vel} + w_{ee} \cdot r_{ee} + w_{CoM} \cdot r_{CoM}$$

where r_{pose}, r_{vel}, r_{ee}, and r_{CoM} refer to pose, joint velocity, end-effector, and CoM tracking, respectively.

These components are defined as:

$$\begin{aligned} r_{pose} &= \exp(-10 \cdot \|q^i - q_i\|^2) \\ r_{vel} &= \exp(-0.05 \cdot \|\dot{q}^i - \dot{q}_i\|^2) \\ r_{ee} &= \exp(-40 \cdot \|p_{ee}^t - p_{ee}\|^2) \\ r_{CoM} &= \exp(-10 \cdot \|p_{CoM}^t - p_{CoM}\|^2) \end{aligned}$$

5.3.2. Jump Enhancement Reward

The rewards are calculated based on distinct phases of the jumping motion, which is segmented into three main intervals:

- Preparation phase ($0 \leq \text{phase} < 0.42$)
- Flight phase ($0.42 \leq \text{phase} < 0.7$)
- Landing and stabilization phase ($0.7 \leq \text{phase} \leq 1.0$)

Each reward is active or emphasized depending on the current phase of motion to encourage proper sequencing and form during the jump. A detailed breakdown of all the rewards can be found in Table 5.1.

Symmetry: The symmetry reward plays a crucial role in ensuring that the humanoid maintains the characteristics of a true standing long jump. To discourage asymmetric behaviors, this reward is composed of two components:

- **Foot Symmetry:** During exploration, the humanoid often discovered strategies where one foot is placed slightly ahead of the other before takeoff. Although this sometimes increases jump distance, it deviates from the desired symmetric motion profile of a standing jump. To penalize such behavior, we define the foot symmetry reward as:

  $$r_{\text{footSymmetry}} = \exp(-10 \cdot e_{\text{foot}})$$

  where e_{foot} is the planar distance between the left and right feet in the x and z directions:

  $$e_{\text{foot}} = \sqrt{(p_{\text{leftFoot},x} - p_{\text{rightFoot},x})^2 + (p_{\text{leftFoot},z} - p_{\text{rightFoot},z})^2}$$

  This encourages the feet to remain aligned, promoting symmetry at the moment of takeoff.

- **Lateral Symmetry:** In some cases, the agent adopted a one-footed push-off, leading to lateral displacement during the jump. Such sideways movement is undesirable in the context of a standing long jump. To discourage this, we define the lateral symmetry reward as:

  $$r_{\text{lateral}} = \exp(-10 \cdot |\text{CoM}_y|)$$

  where CoM_y is the y-component of the CoM. A smaller lateral offset leads to a higher reward, thereby encouraging balanced, forward-directed jumps.

Squat Reward: This reward is active during the **preparation phase** ($\text{phase} < 0.42$), where a deep squat is necessary for effective propulsion. To enable a longer jump, achieving a deeper squat prior to takeoff is essential, as it allows the humanoid to generate more force during the propulsion phase.

Table 5.1.: Enhancement Rewards

Reward	**Value**	**Description**
Height	$r_{height} = \exp\left(-10 \cdot (h^t - h)^2\right)$	Matching height of the Expert CoM to Humanoid CoM
Distance	$r_{distance} = \max\left(d_{MoCap}, d_{humanoid}\right)$	Discrepancy in distance covered by MoCap frame to distance covered by humanoid from the initial position
Symmetry	$r_{symmetry} = 0.6 \cdot r_{footSymmetry} + 0.4 \cdot r_{lateral}$	Foot and Lateral symmetry of the humanoid further explained in Section 5.3.2
Floor Contact	1.0 for both feet, 0.2 for one foot	Rewards the humanoid for the sequence of feet landing
Stability	Phase-dependent combination of foot contact, CoM velocity, and foot placement	Encourages balance and controlled landing and takeoff across phases 5.3.2
Squat	$r_{squat} = exp(-7.0 \cdot (0.5 \cdot e_{\text{hip}} + 0.5 \cdot e_{\text{knee}}))$	Enables a deeper squat for a longer jump 5.3.2
Leg Extension	$r_{legExtension} = exp(-10.0 \cdot \lvert e_{\text{footx}} - p_{\text{targetx}} \rvert)$	Pushes the humanoid to extend the feet during takeoff 5.3.2
Upright	$r_{upright} = 0.3 \cdot r_{torso} + 0.2 \cdot (r_{knee} + r_{hip} + r_{footContact}) + 0.1 \cdot r_{CoMVel}$	Stabilizing the post jump phase 5.3.2
Takeoff Velocity	$r_{takeoff} = 0.5 \cdot v_{forward} + 0.5 \cdot v_{vertical}$	Improve the takoff velocity for maximum distance coverage 5.3.2
Rotation	$r_{yaw} = \exp(-10.0 \cdot e_{yaw})$	Penalize rotation of humanoid around z axis

To encourage this behavior, we penalize deviations in the flexion of the hip and knee joints relative to a reference deep-squat posture. Here, e_{hip} and e_{knee} represent the discrepancies between the current and desired torque values for the hip and knee joints, respectively.

Leg Extension Reward: This reward is active during the **flight phase**, with different target distances set for early ($0.5 \leq$ phase < 0.6) and late (phase ≥ 0.6) sub-phases. A critical factor in increasing jump distance is positioning the feet ahead of the torso to maximize forward reach.

In this context, e_{footx} denotes the average distance along the x-axis between both feet and the CoM, while p_{targetx} is a phase-dependent reference distance. This target distance increases before foot landing, as extending the feet forward not only enhances jump coverage but also facilitates regaining stability upon landing.

Upright Reward: This reward is only activated during the **post-landing phase** (phase $>$ 0.7) to encourage stabilization and recovery after the jump. After the jump is completed, it is crucial for the humanoid to stabilize itself, as achieving stability becomes increasingly challenging with longer jumps.

This stabilization process is encouraged through several components:

- **Torso Reward:** Ensures the torso remains upright and balanced.
- **Knee Reward:** Encourages full extension of the knees upon landing.
- **Hip Reward:** Promotes straightening of the hip flexors following the jump.
- **CoM Velocity:** Penalizes residual movement of the CoM, encouraging it to come to rest.
- **Foot Contact:** Rewards full foot landing for stable support.

Takeoff Velocity Reward: This reward is activated just before takeoff during the **late preparation phase** ($0.3 <$ phase < 0.4). It evaluates the upward and forward velocity of the CoM, which are critical for achieving both height and distance. The reward is computed from normalized components of forward and vertical velocity.

Stability Reward: This reward operates across all three phases, with different logic:

- **Preparation phase (phase < 0.42):** rewards stable contact with the ground.
- **Flight phase ($0.42 \leq$ phase < 0.7):** rewards forward foot placement relative to the CoM.
- **Landing phase (phase ≥ 0.7):** combines foot contact and low CoM velocity to encourage stable landings.

5.4. Curriculum-Based Reward Weighting

To gradually shift the agent's focus from imitation to skill refinement, we define a curriculum factor $\kappa \in [0,1]$ that increases over training for 10 million episodes. The reward weights are adapted dynamically as follows:

$$\begin{aligned}
w_S &= 0.5 \cdot (1.0 - \kappa) + 0.1 \\
w_G &= 0.2 - 0.1 \cdot \kappa \\
w_{\text{stab}} &= 0.2 \cdot \kappa + 0.05 \\
w_{\text{dist}} &= 0.5 \cdot \kappa + 0.3 \\
w_{\text{squat}} &= 0.3 \cdot \kappa + 0.15 \\
w_{\text{leg}} &= 0.3 \cdot \kappa + 0.15 \\
w_{\text{sym}} &= 0.1 \cdot \kappa + 0.05 \\
w_{\text{takeoff}} &= 0.2 \cdot \kappa + 0.1 \\
w_{\text{yaw}} &= 0.2 \cdot (1.0 - \kappa) + 0.05 \\
w_{\text{upright}} &= \begin{cases} 0.3 \cdot \kappa + 0.1 & \text{if phase} > 1.0 \\ 0 & \text{otherwise} \end{cases}
\end{aligned}$$

The final enhancement reward is defined as:

$$\begin{aligned}
r_{\text{enhance}} = {} & w_G \cdot r_{\text{height}} + w_{\text{dist}} \cdot r_{\text{distance}} + w_{\text{squat}} \cdot r_{\text{squat}} + w_{\text{leg}} \cdot r_{\text{legExtension}} \\
& + w_{\text{takeoff}} \cdot r_{\text{takeoff}} + w_{\text{yaw}} \cdot r_{\text{yaw}} + w_{\text{upright}} \cdot r_{\text{upright}} \\
& + w_{\text{stab}} \cdot r_{\text{stability}} + w_{\text{sym}} \cdot r_{\text{symmetry}}
\end{aligned}$$

The total reward is:

$$r_{\text{total}} = w_S \cdot r_{\text{imitation}} + r_{\text{enhance}}$$

5.5. Summary

In this chapter, we presented a comprehensive approach to enhancing explosive motion in a simulated humanoid, focusing on the standing long jump. By leveraging MoCap data for imitation and layering task-specific RL objectives, we shaped behavior that not only replicates human-like movement but also surpasses the original jump performance in terms of distance and stability.

We began by structuring the jump into distinct motion phases: preparation, takeoff, flight, and landing, and designed phase-specific rewards that target critical attributes such as leg extension, symmetry, squat depth, and post-landing stability. A curriculum-based reward weighting scheme was introduced to gradually shift the agent's learning from pure imitation to performance-driven motion refinement.

The reward design balances fidelity to expert motion with the flexibility needed for skill enhancement, enabling the agent to generate robust, symmetrical, and effective jumping behavior. This formulation lays the groundwork for training controllers capable of producing diverse, high-quality explosive motions in simulated humanoids.

6 Results and Analysis

In this chapter, we present a comprehensive evaluation of the learned policy's performance, benchmarked against expert motion data. This analysis integrates both qualitative and quantitative metrics to assess imitation fidelity, training stability, and robustness under various perturbations.

We begin by quantifying how closely the humanoid replicates expert trajectories using Dynamic Time Warping (DTW), Mean Squared Error (MSE), and biomechanical indicators, providing insight into phase-specific similarities and deviations. Next, we analyze training dynamics through convergence curves and ablation testing, isolating the contribution of individual reward components to the final behavior.

We also investigate the underlying factors contributing to the improved jump performance beyond mere replication, seeking to understand how the learned policy leverages the training structure and reward formulation. Finally, we evaluate the robustness of the policy by exposing it to environmental and morphological disturbances, and assess control efficiency and landing precision under these conditions.

The humanoid policy was trained for a total of 30 million episodes, with convergence achieved after approximately 14 million episodes. On average, each episode consisted of 140 simulation steps, which included a small post-landing buffer to ensure additional stability. After 20 million episodes, overtraining effects were observed, causing the policy to fail in completing the initial tasks reliably. The performance of the model can be observed in Figure A.1

All experiments were conducted on a workstation equipped with a 13th Gen Intel Core® i7-13700 processor (16 cores / 24 threads, base frequency 0.8 GHz, turbo boost up to 5.2 GHz, 30 MB L3 cache) with x86_64 architecture. MuJoCo's physics simulation, graphics processing, and visualization were executed exclusively on the CPU, while policy training in PyTorch was performed on an NVIDIA GeForce RTX 3050 GPU (8 GB VRAM, CUDA 12.2). The total wall-clock training time was approximately 13 hours, achieving an average throughput of 641 environment steps per second. No environment parallelization was employed; all simulations were executed in a single MuJoCo instance.

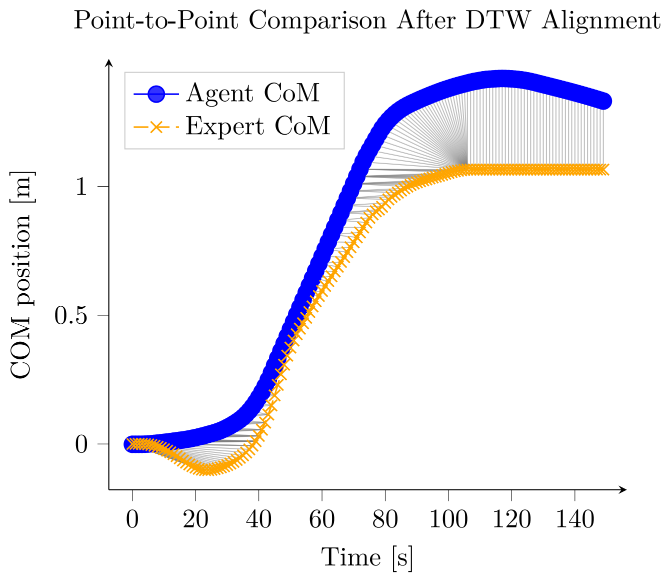


Figure 6.1.: DTW of CoM in comparison to expert

6.1. Policy Performance Validation

6.1.1. Model Imitation Accuracy

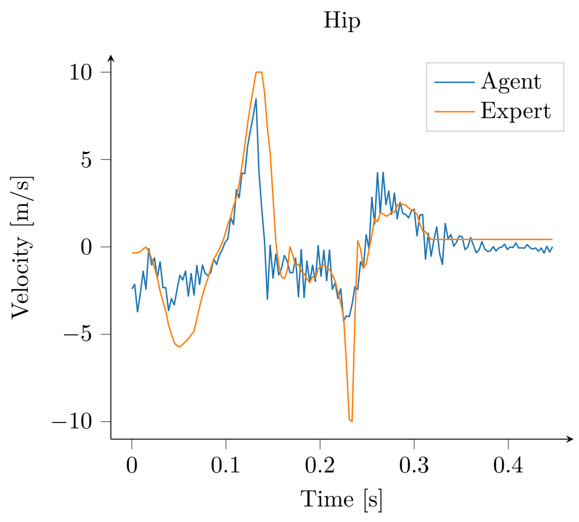
Dynamic Time Warping

To quantitatively evaluate the similarity between the agent's motion and the expert demonstration, we conducted a DTW analysis on the CoM trajectories. DTW computes the optimal temporal alignment between two sequences that may differ in length or speed, making it particularly suitable for analyzing dynamic motions with varying execution rates. It is given by the formula

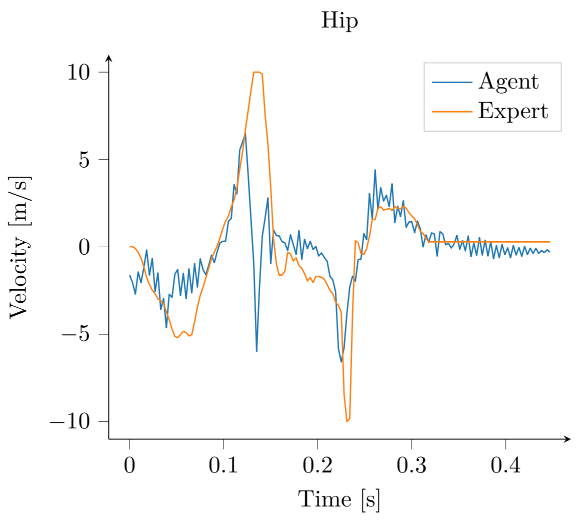
$$\mathrm{DTW}(A,B) = min\sqrt{\sum_{(i,j)} d(a_i,b_j)^2}$$

where $d(a_i, b_j)$ is the distance between points a_i from series A and b_j from series B , and the sum is taken over all points (i, j) in the optimal alignment path. The CoM trajectories of both the expert and the trained agent were extracted and compared on a frame-by-frame basis. A lower DTW distance indicates a higher degree of similarity, implying that the agent successfully replicates the expert's CoM path, whereas larger distances reflect greater deviations in spatial positioning or timing.

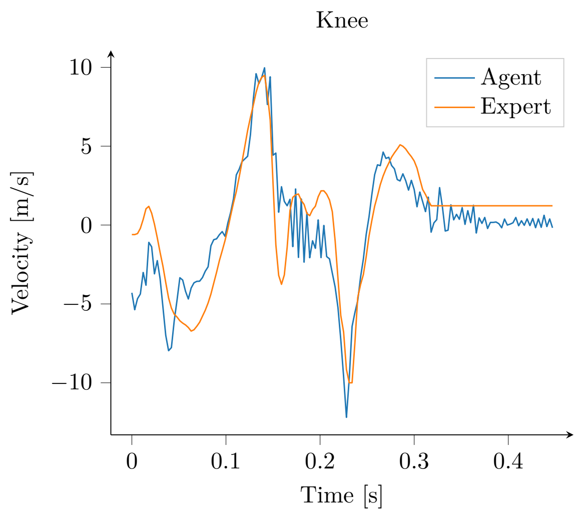
In these experiments, a similarity score of 0.012 was achieved, demonstrating that the humanoid agent closely follows the expert's motion trajectory while adhering to the imposed physical and control constraints. Furthermore, the analysis revealed a significant improvement in the distance covered by the agent compared to the expert, suggesting that the learned policy not only imitates the reference motion but also optimizes certain aspects of the jump, such as force generation and take-off dynamics. This makes DTW-based evaluation a valuable complement to reward-driven performance metrics, offering deeper insights into the fidelity and efficiency of the learned explosive motion.



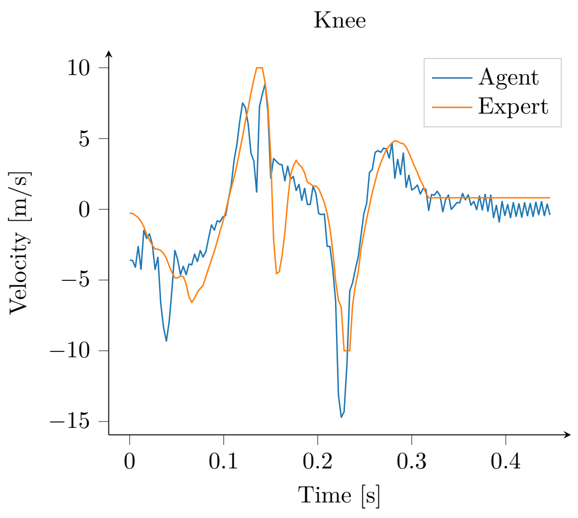
(a) Left Hip Velocity Tracking



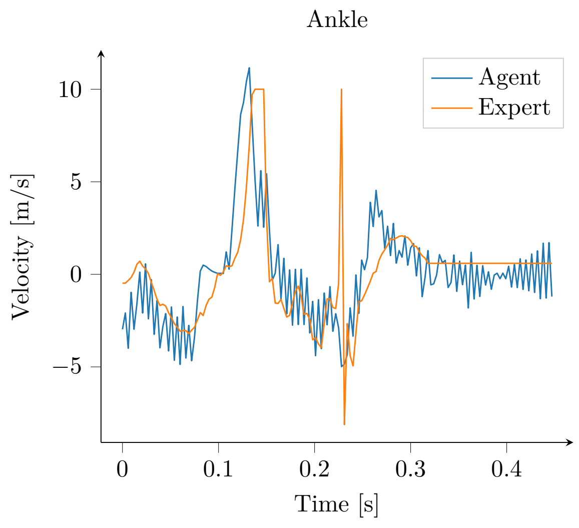
(b) Right Hip Velocity Tracking



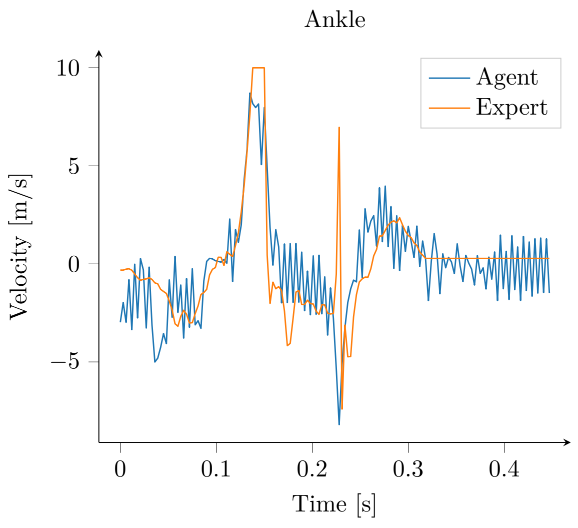
(c) Left Knee Velocity Tracking



(d) Right Knee Velocity Tracking

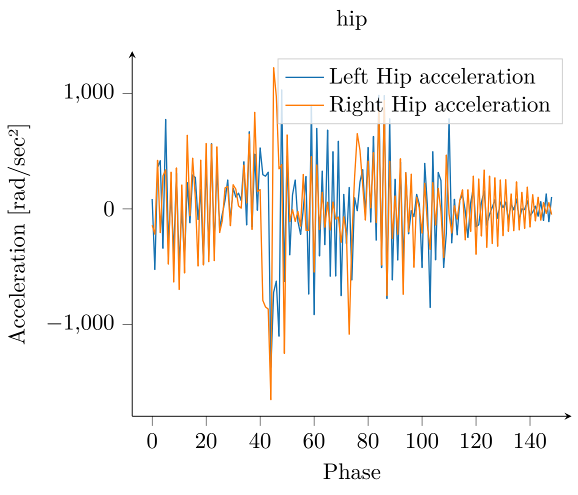


(e) Left Ankle Velocity Tracking

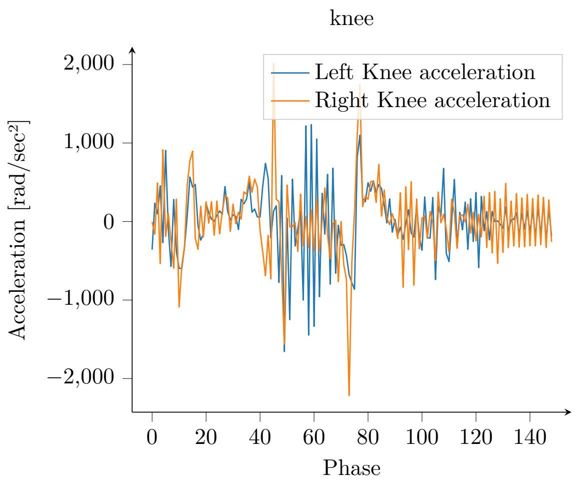


(f) Right Ankle Velocity Tracking

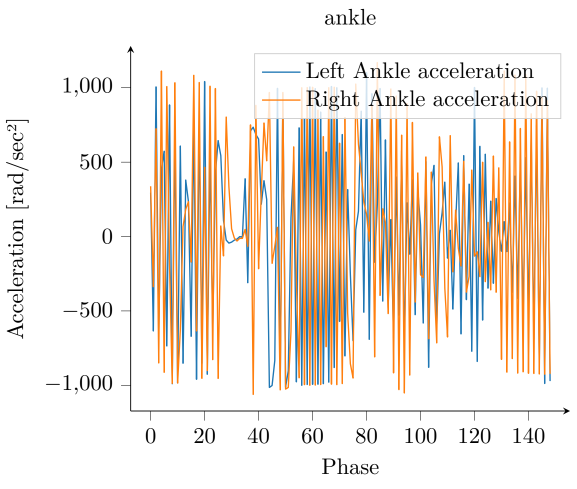
Figure 6.2.: Comparison of joint velocity for the leg joints.



(a) Hip acceleration



(b) Knee acceleration



(c) Ankle acceleration

Figure 6.3.: Comparison of joint acceleration in both the legs.

Trajectory Smoothness

Smoothness of motion was evaluated using derivative-based metrics, specifically joint accelerations and jerk (the third derivative of position). These measures were compared between expert and humanoid trajectories, supported by visual overlays of joint angle profiles. The results indicated that, while most joints broadly followed the expert's motion trends, certain joints contributed disproportionately to the humanoid's jump performance. As shown in Figures 6.2 and 6.3, the transition phases (takeoff and landing) exhibited the highest levels of instability, as reflected by increased accelerations and jerk magnitudes.

Acceleration profiles revealed that the humanoid achieved relatively smoother transitions during landing compared to takeoff. In contrast, the takeoff phase was characterized by higher jerk values, reflecting the explosive nature of the movement and the rapid force generation required for propulsion. This asymmetry suggests that the policy has learned to prioritize landing stability, an essential aspect of successful jump execution, while takeoff still shows signs of mechanical roughness.

The joint velocity profiles further corroborated this observation. During takeoff, the humanoid's joint velocities displayed sharper peaks and more abrupt transitions compared to the expert, indicating less refined control in explosive phases. Conversely, the landing phase was marked by smoother velocity profiles, with reduced oscillations and more consistent deceleration, showing improved absorption of ground impact forces.

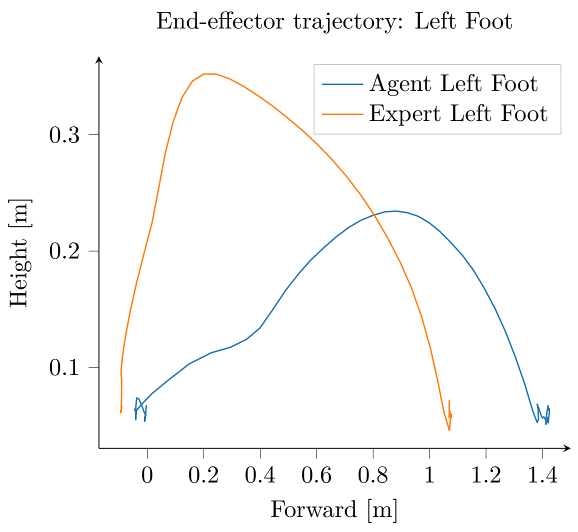
Overall, these findings suggest that the humanoid has mastered the core motion patterns of the jump and has developed a stable landing strategy. However, the persistence of erratic velocity fluctuations and elevated jerk during takeoff highlights an area for improvement, particularly in refining the smoothness of explosive movements and achieving more expert-like propulsion dynamics.

Imitation Error

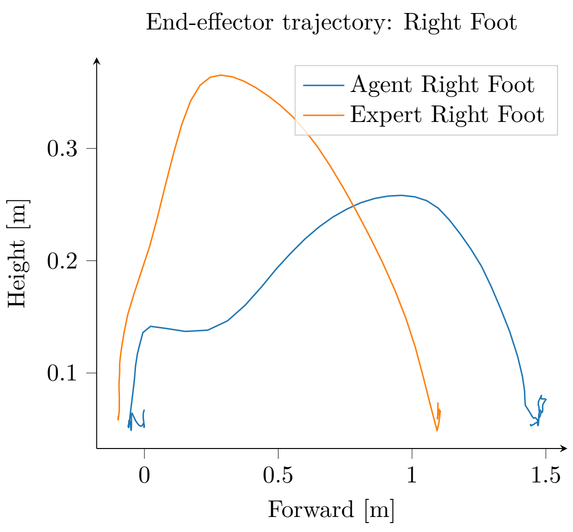
To quantify the imitation error, we computed the average discrepancies between the humanoid and expert trajectories for the CoM, end effectors, and major leg joints. Figures 6.4 and 6.5 illustrate the tracking errors for the end effector trajectories and joint angles, respectively. Since the humanoid was explicitly trained to imitate expert motion, the overall tracking errors remain relatively low, confirming that the policy successfully captured the essential motion patterns of the expert.

The end effector tracking error (Figure 6.4) demonstrates that the humanoid closely follows the experts limb trajectories, with only minor deviations in position across the motion cycle. This level of fidelity is crucial for achieving coordinated and biomechanically consistent landings, as even small inaccuracies in foot placement can destabilize the motion. The joint tracking error (Figure 6.5) further indicates that the humanoid reproduces the experts articulation with high accuracy. While deviations exist, they are localized to joints with high dynamical demands, such as the knees and ankles. Interestingly, the joint error plots also reveal the explosiveness of the humanoids jumps, as the policy learns to rapidly adjust its hip and knee angles to generate the necessary thrust.

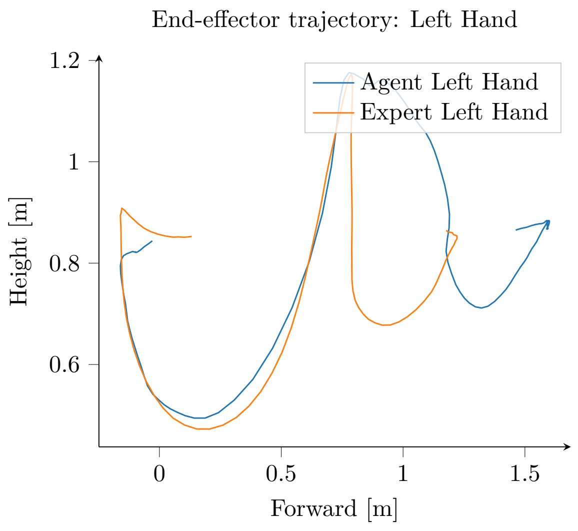
In addition, Figure 6.6 highlights the evolution of the humanoids jumping ability over



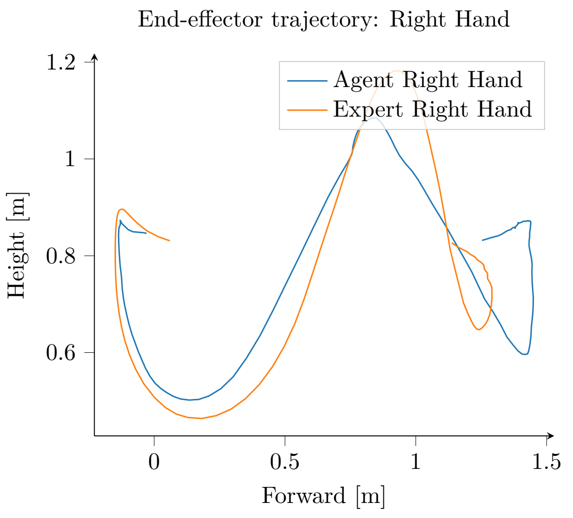
(a) Left Foot tracking error



(b) Right Foot tracking error

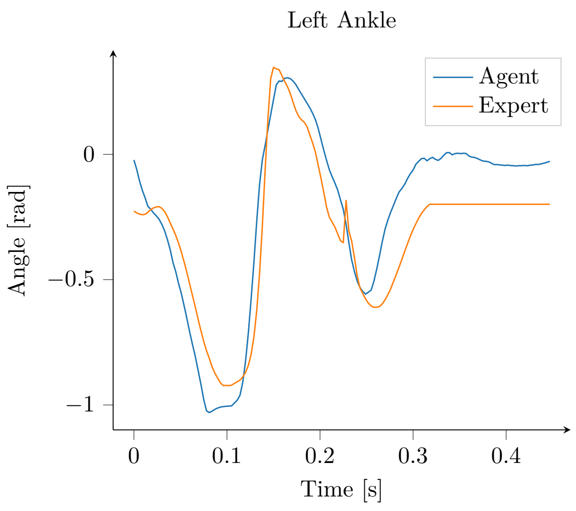


(c) Left Hand tracking error

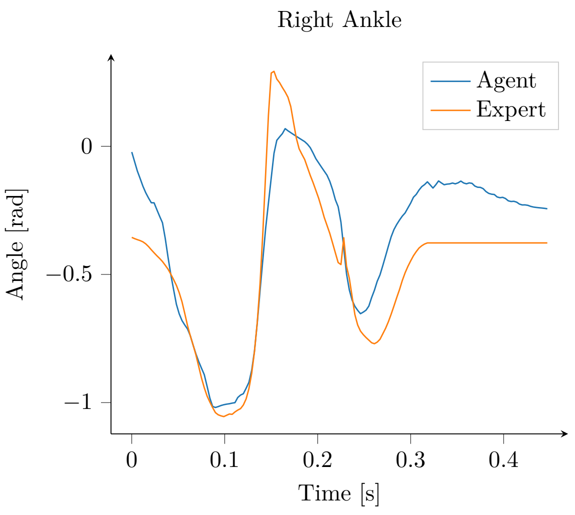


(d) Right Hand tracking error

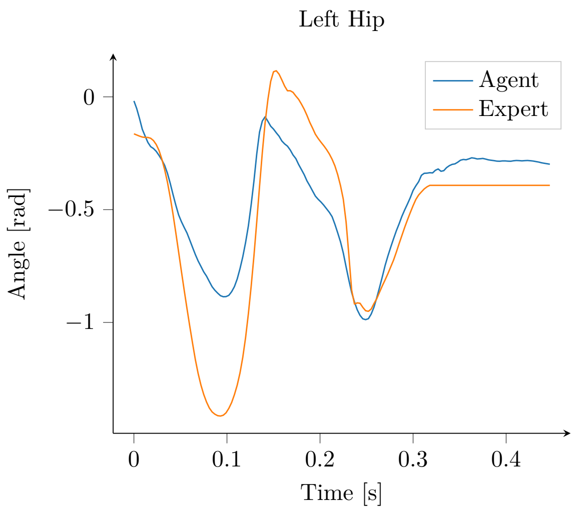
Figure 6.4.: Comparison of motion tracking performance for the end effectors.



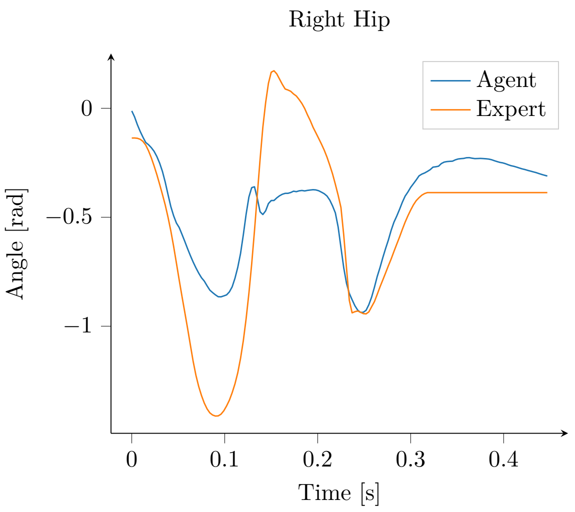
(a) Left Foot tracking error



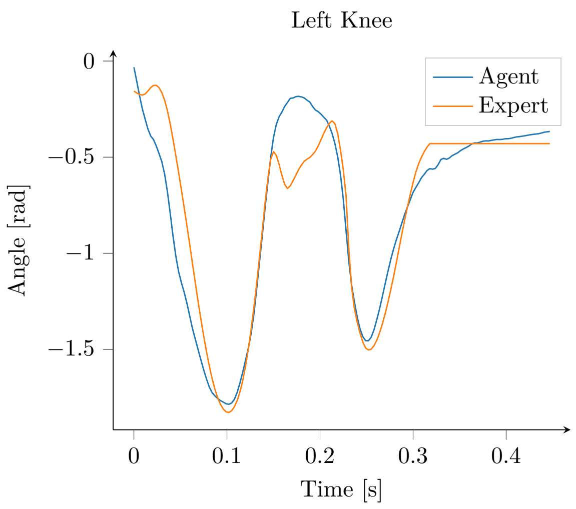
(b) Right Foot tracking error



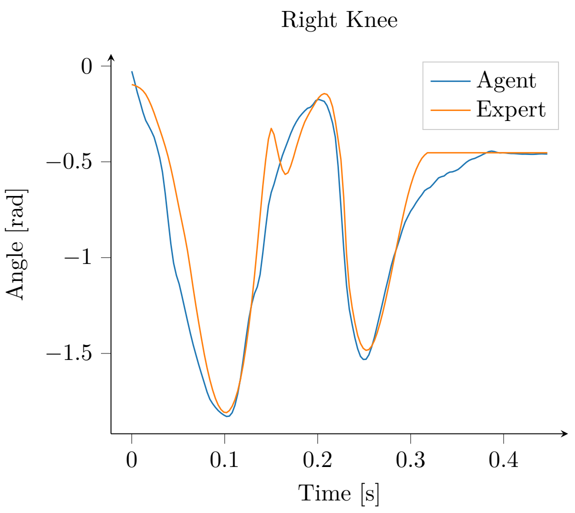
(c) Left Hand tracking error



(d) Right Hand tracking error



(e) Left Hand tracking error



(f) Right Hand tracking error

Figure 6.5.: Tracking major leg joints in comparison to expert

the training period. The CoM trajectory shows that the agent not only maintains strong alignment with the experts vertical (z) and lateral (y) motion but also exhibits a notable increase in forward displacement along the x-axis. This pattern reflects a learned optimization: the policy prioritizes maximizing horizontal distance while maintaining the stability of the CoM. Such behavior is directly attributable to the interplay of reward shaping and imitation learning, which encourages both task completion and stylistic similarity to the expert.

Finally, Figures 6.2 and 6.3 provide further insight into the biomechanics of the learned motion by comparing joint velocity and acceleration profiles. These plots reveal that the push-off phase is primarily powered by the hips and knees, with strong bursts of velocity and acceleration aligning with the experts propulsion strategy. The ankle joints provide fine-tuned adjustments at takeoff and landing, ensuring smoother trajectory control. The coordination across these joints demonstrates that the humanoid not only learned to reproduce the visual appearance of the jump but also captured the underlying dynamics required for stable, powerful motion.

Taken together, these results show that the humanoid policy achieves a balance between motion fidelity and performance optimization: it maintains close tracking of expert kinematics while simultaneously adapting the trajectory to achieve greater horizontal displacement, thereby demonstrating both imitation accuracy and task-specific adaptation.

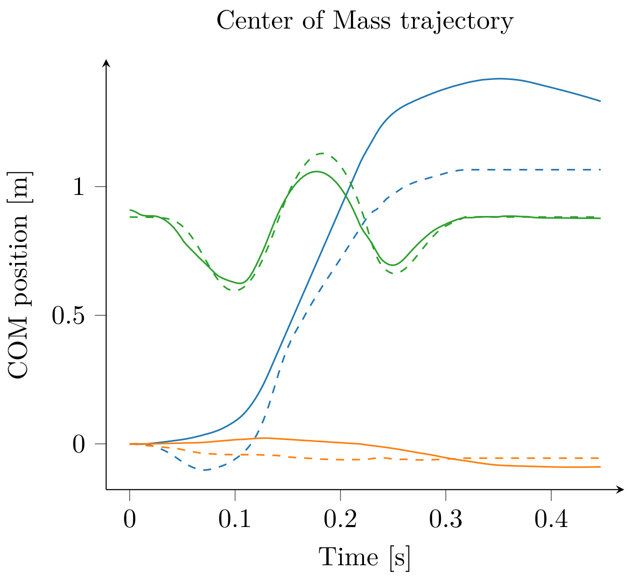


Figure 6.6.: CoM tracking error in comparison to expert. The dotted lines are expert performances, whereas the solid lines are the agents performance. The blue line is the X position of the CoM, the orange line is the Y position of the CoM, and the green line is the Z position of the CoM.

6.1.2. Mass and Torque Perturbation Experiments

To improve policy robustness during training, random perturbations were applied to both the humanoid's body mass and actuator torque parameters. The perturbation scale $[0.8, 1.2]$ corresponds to $\pm 20\%$ deviations from the nominal values, simulating real-world variability such as payload changes, manufacturing tolerances, or actuator performance inconsistencies. These experiments were conducted under three conditions: mass perturbation only, torque perturbation only, and simultaneous mass and torque perturbations.

Mass Perturbation

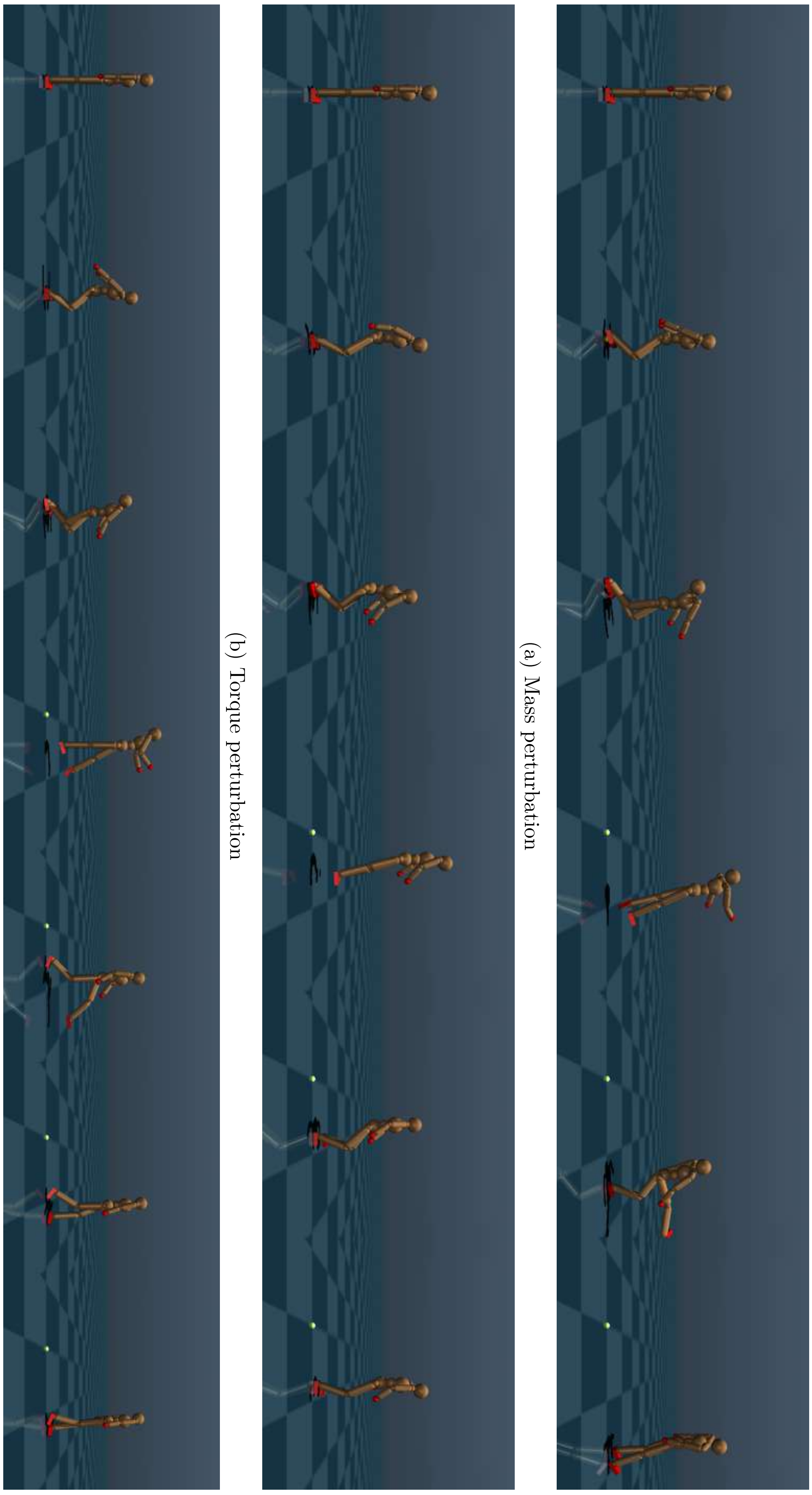
When body mass was varied, the humanoid adapted by shifting its stability strategy towards unilateral reliance on the left leg during takeoff and landing. The resulting motion resembled a one-legged jump, with the left foot acting as a stable anchor on landing, followed by a right-leg push to extend forward displacement. Over time, the policy discovered that oscillating (or "flicking") the legs during the airborne phase improved horizontal distance and aided post-landing recovery. This leg oscillation likely enhanced momentum redistribution, compensating for altered mass distribution and center-of-mass dynamics. Interestingly, the symmetry reward exerted less influence under mass perturbation, as stability became the dominant factor. While this adaptation improved jump distance, it reduced fidelity to the expert reference motion, illustrating the trade-off between task performance and imitation accuracy when mass distribution changes. Refer Figure 6.7a

Torque Perturbation

Under actuator torque variability, the humanoid exhibited a rightward bias in stance while actively preserving foot symmetry and post-landing stability. Torque fluctuations appeared to limit the agent's capacity for consistent force generation, causing the policy to prioritize stability maintenance over distance maximization. As a result, while symmetry and balance were preserved, horizontal displacement gains were minimal within the observed training window. It is plausible that extended training could partially recover distance performance, but in the current experimental timeframe, the stability-over-distance trade-off remained dominant. Refer Figure 6.7b

Combined Mass and Torque Perturbation

When both mass and torque perturbations were applied simultaneously, the humanoid initially demonstrated promising adaptation. For the first $\sim$1,000,000 episodes, it progressively increased jump distance while maintaining landing stability, predominantly relying on the left foot for push-off. The resulting strategy integrated elements from both individual perturbation cases—dynamic leg oscillation from the mass-perturbed policy and controlled, stability-oriented push-off from the torque-perturbed policy. This combined control pattern closely resembled realistic adaptive behavior in environments with both mass and actuation uncertainty. However, after $\sim$1,100,000 episodes, the policy collapsed into a degenerate behavior, holding the left leg fully lifted and ceasing effective motion. This suggests that the combined perturbation magnitude, coupled with the exploration



(a) Mass perturbation

(b) Torque perturbation

(c) Combined perturbations

Figure 6.7.: Motion trail compositions for different mass and torque conditions.

strategy, may have induced overfitting to a suboptimal local minimum. Curriculum learning or progressive perturbation scaling may be required to sustain robust adaptation in such multi-variable uncertainty scenarios. Refer Figure 6.7c.

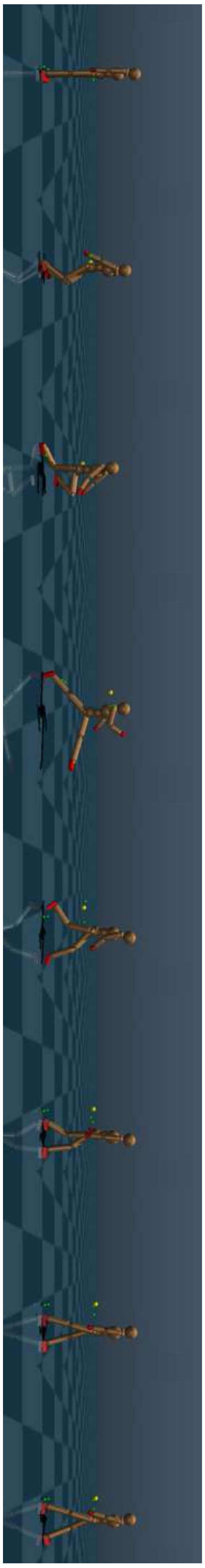
Conclusion
These experiments indicate that introducing physical variability during training drives the emergence of adaptive control strategies, but also alters the trade-off between stability, distance, and motion fidelity. Mass perturbations promoted asymmetrical leg use and improved horizontal displacement, while torque perturbations emphasized symmetry and stability at the expense of distance. Simultaneous perturbations produced realistic adaptive motion patterns but also highlighted the risk of convergence to non-functional behaviors when perturbations are too aggressive. Careful tuning of perturbation parameters and training schedules is therefore essential for achieving both robustness and biomechanical plausibility in humanoid motion control.

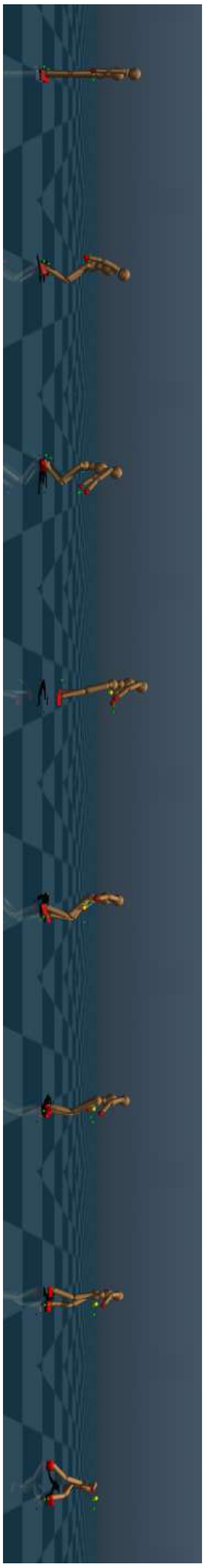
6.1.3. Ablation Testing

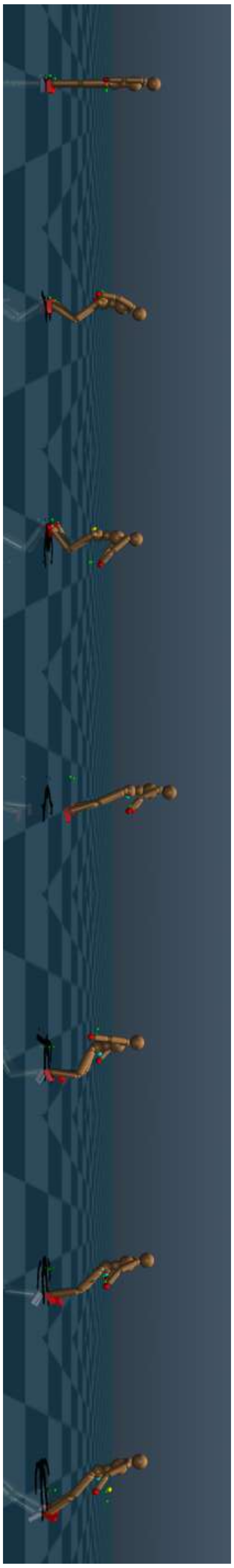
All ablation experiments were evaluated over a duration of 150 episodes. In each test, specific terms in the reward function were systematically omitted to assess their individual contribution to training stability, motion quality, and jumping performance. The goal was to determine whether these terms positively influence the humanoid's ability to improve jump distance and overall motion fidelity. Table 6.1 summarizes the tested ablation configurations and the associated evaluation metrics. Take a look at Figure 6.8 for an overview of the performances of the humanoid.

When the imitation reward was removed, the humanoid exhibited a progressive bias towards the right side during jumps, with excessive flinging of the left leg. This likely emerged from the agent exploiting the reward structure by increasing the mean foot position through extended left leg movement. The resulting strategy prioritized jump distance over stability, producing frequent instability during landing. By evaluation step 23,000, the humanoid briefly discovered a stabilizing tactic—stretching the legs post-landing—but quickly abandoned it. The motion evolved into a "drag foot" strategy, where one foot was placed far ahead and the other dragged forward after landing to extend total measured distance. This clearly indicates that in the absence of imitation guidance, the policy heavily exploits distance-based rewards at the expense of stability and biomechanical plausibility. Refer Figure 6.8a.

Without the height reward, jump apex was significantly reduced, often leading to incomplete motions and a trotting gait rather than a ballistic jump. This conservative height adaptation favored stability but severely limited horizontal displacement. The mismatch between actual and expected jump height, compounded by excessive torque generation in the lower limbs, resulted in inaccurate trajectory estimation and frequent tripping. Refer Figure 6.8b. Removing the stability reward, on the other hand, produced visibly unstable landings, with excessive post-impact oscillations in the legs. Refer Figure 6.8c.


(a) No Imitation Reward


(b) No Height Reward


(c) No Stability Reward

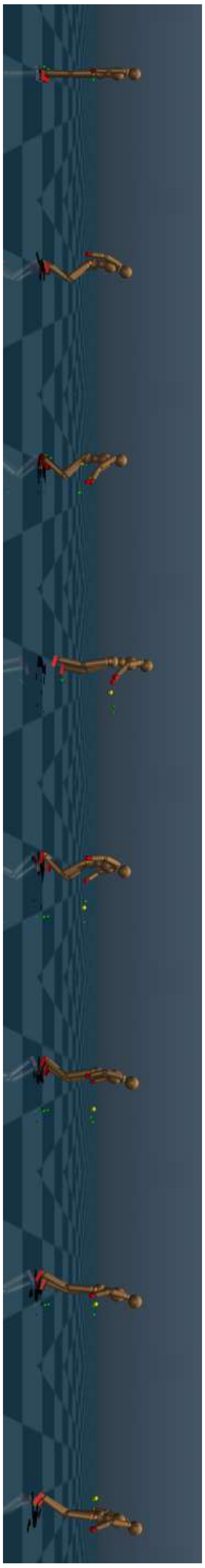
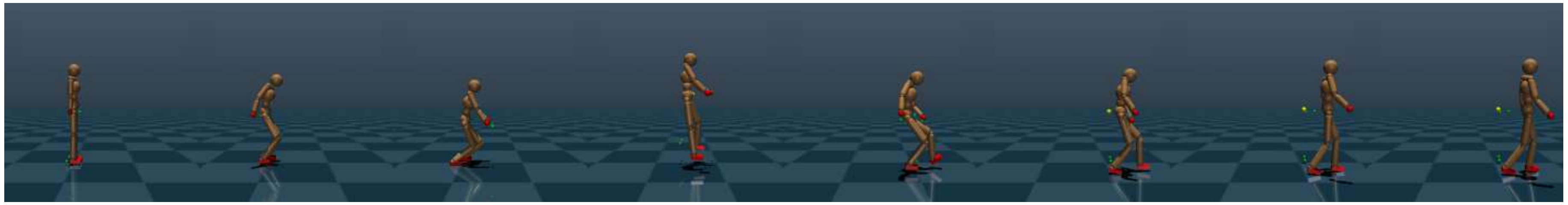
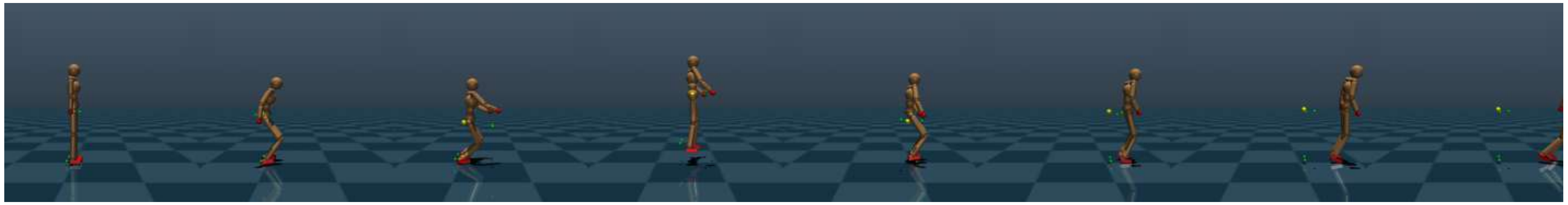
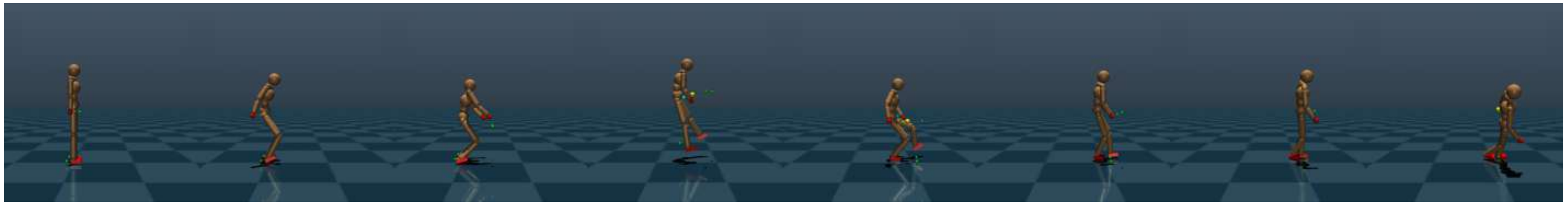

(d) No Distance Reward

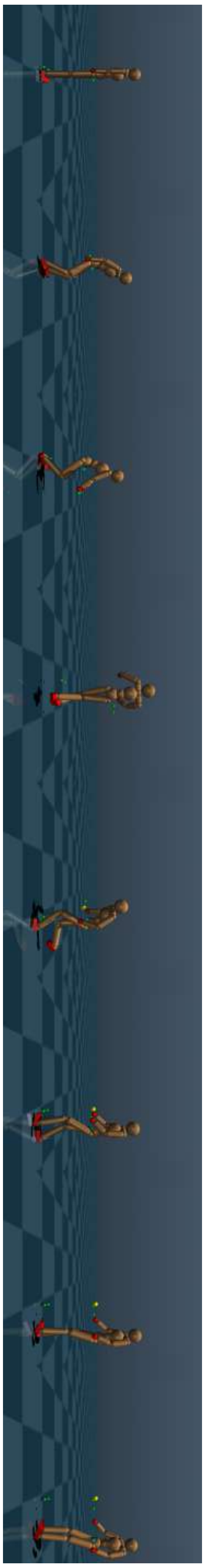
Figure 6.8.: Motion trail compositions for removal of imitation, height, stability, and distance rewards.

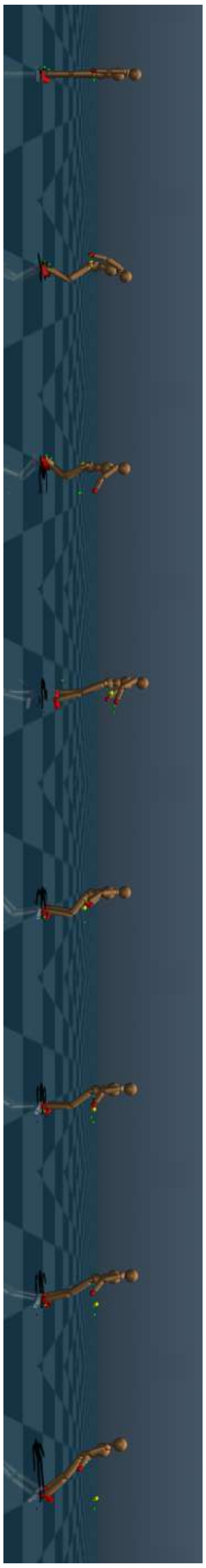

(a) No Squatting Reward

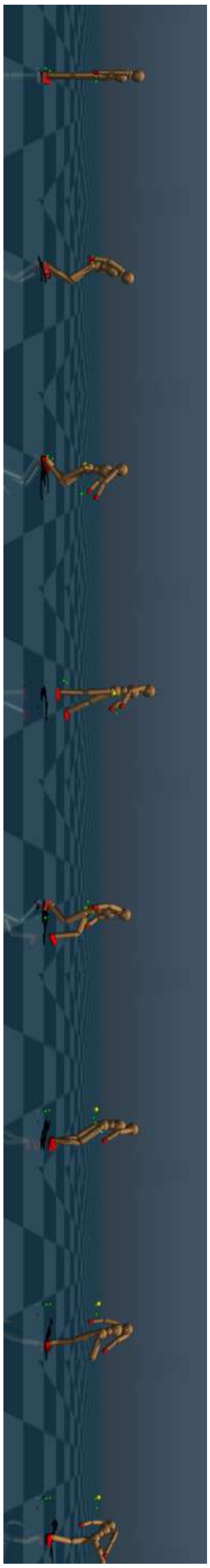

(b) No Leg Extension Reward


(c) No Symmetry Reward

(d) Motion trail compositions for removal of squatting, leg extension, and symmetry rewards.


(a) No Yaw Reward


(b) No Takeoff Reward


(c) No Upright Reward

(d) Motion trail compositions for removal of yaw, takeoff, and upright rewards.

Table 6.1.: Ablation study of additional reward terms for motion enhancement. Each ablation removes a single reward component, and the effects on jump distance, landing success, symmetry score, energy efficiency (η), and imitation mean squared error (MSE) are evaluated relative to the baseline.

Ablation Mode	**Jump Distance (m)**	**Symmetry Score**	**Energy Efficiency (η)**	**Imitation MSE**
Imitation	1.41	0.11	0.086	0.11
Height	0.99	0.17	0.055	0.05
Stability	1.35	0.25	0.07	0.085
Distance	0.47	0.31	0.024	0.098
Squat	0.89	0.6	0.049	0.053
Leg Extend	1.53	0.15	0.108	0.33
Symmetry	1.49	0.112	0.089	0.152
Yaw	1.07	0.115	0.06	0.118
Takeoff	1.32	0.04	0.077	0.25
Upright	2.03	0.177	0.083	0.294

Omitting the distance reward shifted the humanoids focus almost entirely to stability preservation. The agent learned to execute rapid foot movements to remain balanced, sometimes generating unintentional forward displacement. However, across many episodes, horizontal progress was minimal, and jumps became stagnant. Refer Figure 6.8d.

Removing the squat reward degraded explosive force generation during takeoff. While occasional distance increases were observed, these gains were inconsistent and often accompanied by compromised landing stability. The absence of the leg extension reward produced similar effects to height reward removal: reduced aerial phase duration, frequent trotting patterns, and instability during extended forward motion. Refer Figure 6.9a.

Without the symmetry reward, the humanoid diverged significantly from expert motion kinematics. A persistent asymmetrical foot placement emerged, with one leg consistently positioned ahead of the other to maximize measured displacement. This asymmetry negatively affected both landing stability and takeoff dynamics, leading to insufficient torque buildup before the jump. Refer Figure 6.9c

Omitting the takeoff reward resulted in shorter jumps than the baseline, as the agent failed to generate sufficient linear velocity at liftoff. Refer Figure 6.10b. Removing the yaw reward led to a compensatory behavior where the humanoid rotated around its vertical axis to improve stability and extend distance, a behavior inconsistent with the expert demonstration. Refer Figure 6.10a. Finally, without the upright reward, the humanoid frequently toppled post-landing due to inadequate control of the center-of-mass velocity, resorting to erratic leg movements to regain balance—often unsuccessfully. Refer Figure 6.10c.

Conclusion

The ablation study confirms that each reward component plays a distinct role in shaping both performance and motion style. Distance and imitation rewards strongly influence forward displacement, while height, squat, and leg extension rewards are crucial for preserving realistic jump mechanics. Stability, symmetry, upright, and yaw rewards are essential for maintaining biomechanical plausibility and post-landing control. Omitting any single term degrades performance in at least one key dimension, highlighting the necessity of a multi-component reward formulation for effective explosive motion imitation.

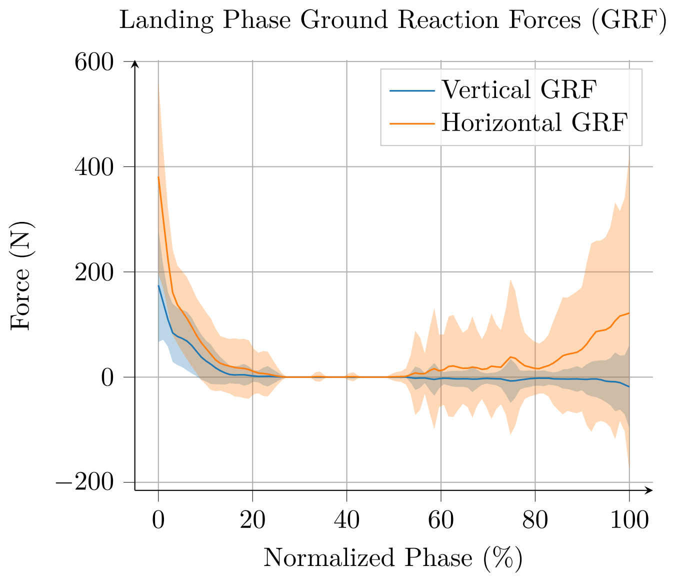


Figure 6.11.: Ground Reaction Forces during landing phase

6.2. Model Validation and Robustness

6.2.1. Adaptability

To evaluate the robustness of the trained policy, we subjected the humanoid to a variety of environmental perturbations, including slippery terrain, inclined platforms, and external pushes during the jump phase. Performance was quantified using several key metrics: the mean and variance of jump distance across trials, symmetry reward and foot contact score at landing, and the percentage of successful landings (defined as trials without falls or collapses). Results showed that while the agent maintained stable performance on flat ground, it was able to adapt effectively to a slippery surface. To test this, we changed the parameter `friction` for the surface from $1, 0.1, 0.1$ to $1, 0.001, 0.001$. The agent was able to stick the landing despite the reduced friction.

However, when the surface was inclined by adjusting the `euler` parameter along the x and y axes to 2 degrees, the humanoid failed to maintain balance, indicating limited proficiency on tilted surfaces. Similarly, when external perturbations were applied during the jump, the humanoid struggled to achieve a stable landing. These results suggest that while the agent has effectively learned the jump motion, it lacks intrinsic robustness and stability under non-ideal conditions. External factors, such as surface inclination or perturbations during the jump phase, significantly influence performance. Although the humanoid can still achieve improved jump distances, its inability to stabilize post-landing highlights that the learned policy relies heavily on ideal conditions for optimal performance.

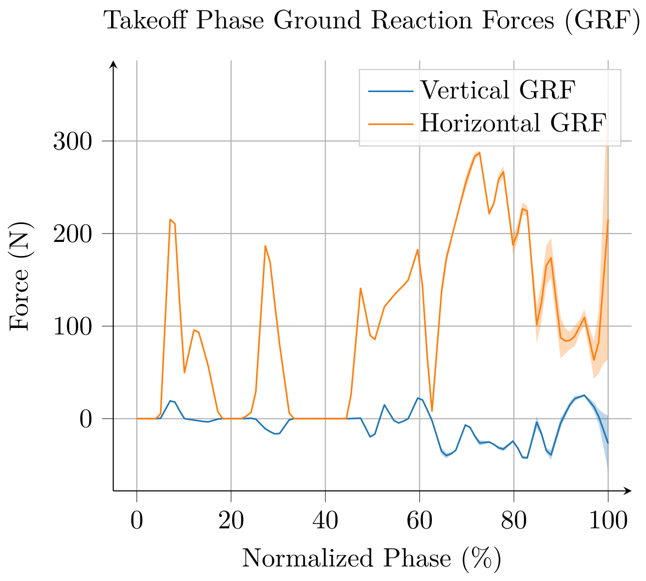


Figure 6.12.: Ground Reaction Forces during takeoff phase

6.2.2. Biomechanical Analysis

To examine the physical plausibility of the learned motion, we analyzed biomechanical indicators across the jump cycle. Ground Reaction Force (GRF)s were measured at takeoff and landing, capturing peak values, average magnitudes, and directional consistency. These measurements ensure that the humanoid generates and absorbs forces in a realistic and physically consistent manner. As discussed in Sections 6.1.1, the humanoid successfully learned to increase jump distance while maintaining a stable landing strategy. Figures 6.11 and 6.12 illustrate the GRF profiles during the landing and takeoff phases, respectively.

During takeoff, the GRF profiles show that the humanoid produces sufficient force to achieve the desired jump height and distance. While vertical lift is not the primary objective of the task, since maximizing horizontal distance is the main goal, the agent nevertheless achieves a reasonable jump height. Importantly, the takeoff torques remain below the joint torque limits, indicating that the motion is biomechanically feasible and does not rely on unrealistic joint actuation. The horizontal component of the GRF is considerably larger than the vertical component, consistent with the policys emphasis on generating forward momentum to optimize horizontal displacement.

The landing phase reveals a different dynamic. At initial ground contact, the humanoid experiences very high impact forces, briefly exceeding the maximum torque capacity of its joints. These sharp peaks are confined to the first instants of contact, where most of the impact energy must be absorbed. Beyond this critical moment, the GRF profiles flatten, showing that the humanoid is able to stabilize itself with relatively modest torque demands. Additional torque is then applied for post-landing stabilization, ensuring balance is recovered after impact.

Overall, the GRF profiles indicate that the humanoid employs a stable and repeatable landing strategy, as evidenced by smooth and consistent patterns across trials. This robustness is further reinforced by the perturbation experiments, where the humanoid maintained comparable landing stability even under challenging conditions. These findings suggest that the agent has internalized biomechanically plausible strategies for both force generation during takeoff and impact absorption during landing, aligning with the underlying task objective of maximizing forward displacement while maintaining stability.

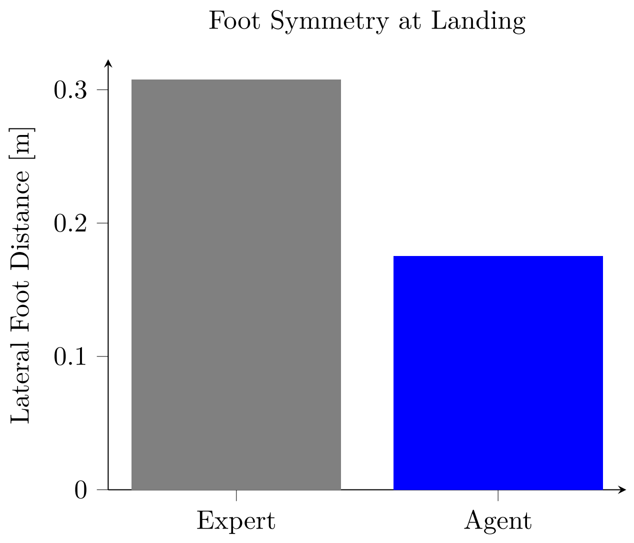


Figure 6.13.: Foot Symmetry during landing phase

6.2.3. Energy Efficiency

The energetic demands of the policy were assessed by computing the total control effort, defined as the sum of squared control signals integrated over time. To contextualize energy use, we derived an efficiency ratio

$$\eta = \frac{p_{\text{forward}}}{c_{\text{energy}}},$$
$$c_{\text{energy}} = \sum_{t=0}^{T} \|\mathbf{u}_t\|^2 \, \Delta t$$

which allows direct comparison between the agent and expert. Results showed that while the agent was capable of achieving comparable jump distances, it typically incurred higher energy expenditure, leading to a lower efficiency ratio. This trade-off highlights a common challenge in learned locomotion policies: effective task completion at the cost of reduced energetic efficiency.

6.2.4. Landing Precision

Landing performance was evaluated to assess both stability and similarity to expert motion. Key metrics included the deviation of each foot from the expected touchdown location, the symmetry error between left and right foot placements, and the temporal sequence of foot contacts during landing. Statistical analysis showed that, while the agent consistently landed within a narrow margin of the target positions, small asymmetries in foot placement and timing remained when compared to the expert. These differences were evident in post-jump stabilization, where the agent required additional corrective steps more frequently than the expert (Figure 6.13).

Visual overlays of foot trajectories highlighted these discrepancies, offering insight into potential improvements in balance and coordination during landing (Figure 6.14). Examination of the jump onset revealed that the humanoid often initiated takeoff slightly earlier than the expert, suggesting a more aggressive push-off strategy. This premature initiation may partially explain the observed variations in landing precision and stability. Conversely, the landing phase frequently began with a marginally delayed foot contact relative to the expert, which likely affected balance recovery.

Overall, these temporal shifts indicate that the learned policy would benefit from refinement to better match expert timing, thereby improving both landing accuracy and post-landing stability.

6.2.5. Mass and Torque Fluctuations

To evaluate the robustness of the learned policy under internal physical variations, we introduce random perturbations in the models actuator gear ratios and body mass distributions. These modifiers emulate real-world scenarios such as actuation inconsistencies, payload handling, or limb morphing.

Setup

For each evaluation episode, actuator gear values were independently scaled by a random factor in the range $[0.8, 1.2]$. This results in varying torque production capabilities across joints. The policy was executed deterministically to assess its adaptability without exploratory noise. We record the joint torques (`actuator_gear`) across all timesteps to compute per-joint variance across episodes. This is also done two fold. We test the robustness of the policy by varying the joint torques, along with train a policy which has a variable joint torque throughout the training of the algorithm. The same is done with the joint masses (`body_mass`). We also measure the forward jump distances and the landing success rate to assess the behavioral impact of internal perturbations. We

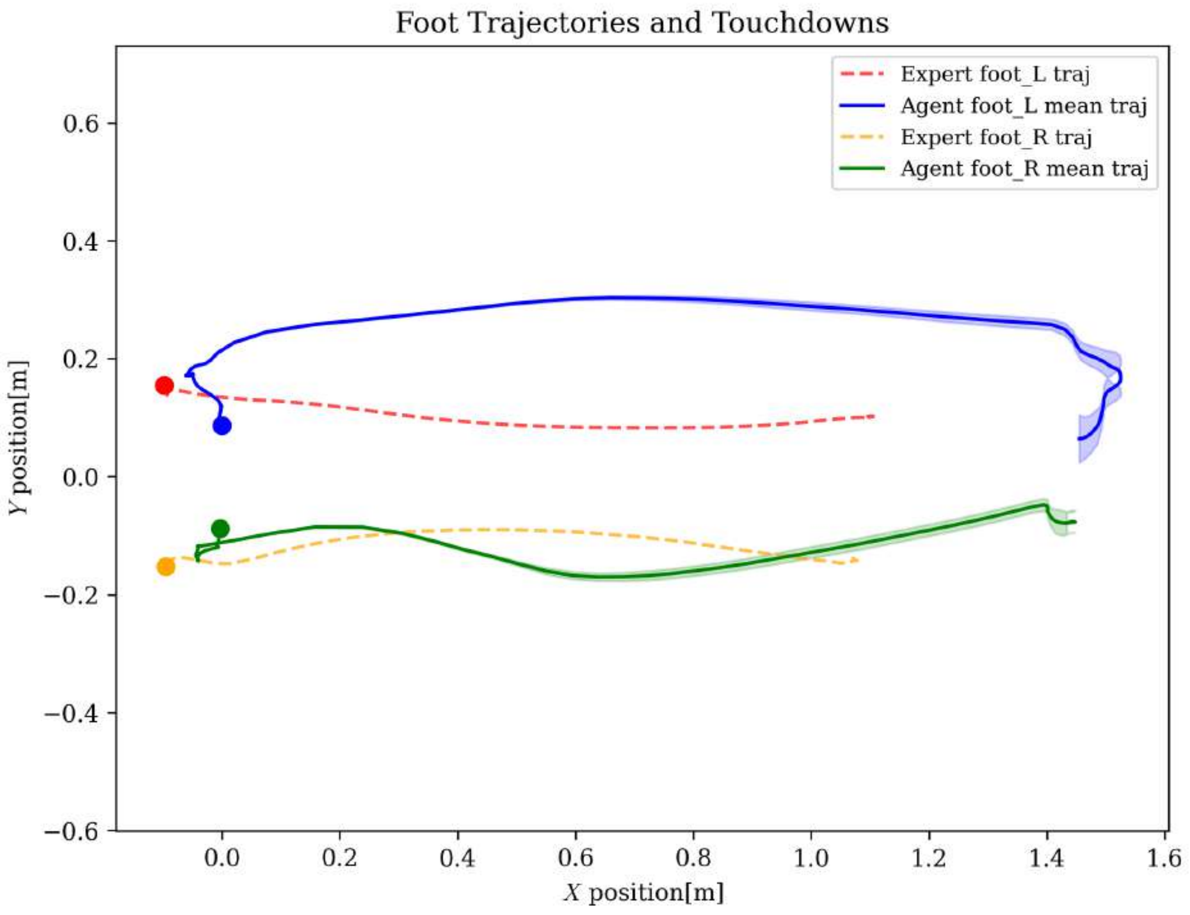


Figure 6.14.: Foot Touchdown positions during landing phase. The first dot signifies the foot takeoff point and the landing points are marked where the trajectories end

also qualitativel observed torque strategies in different jump phases: explosive push-off, mid-air, and landing phase to analyze conpensatory behaviors. Data recorded is for 150 episodes.

Mass Fluctuations

Mass perturbations turned out to have a particularly strong impact on performance, as evidenced by a clear increase in the distance covered by the humanoid. The policy adapted to the altered mass by modifying its landing strategy and creating effective openings for foot placement. With reduced body mass, the humanoid was able to generate proportionally greater joint torques, which translated into longer jumps. This demonstrates that the policy is capable of exploiting mass perturbations to improve forward displacement.

However, this improvement came at the cost of landing stability. While the humanoid achieved a maximum jump distance of 2.5m, significantly higher than the baseline distance of 1.4m, the success rate of stable landings decreased sharply. The mass imbalance reduced the agents ability to absorb impact forces and maintain balance after touchdown. As a result, only 8% of the landing attempts were successful. These findings highlight a trade-off: mass perturbations enhance jump distance but simultaneously compromise landing reliability. Refer Figure 6.15.

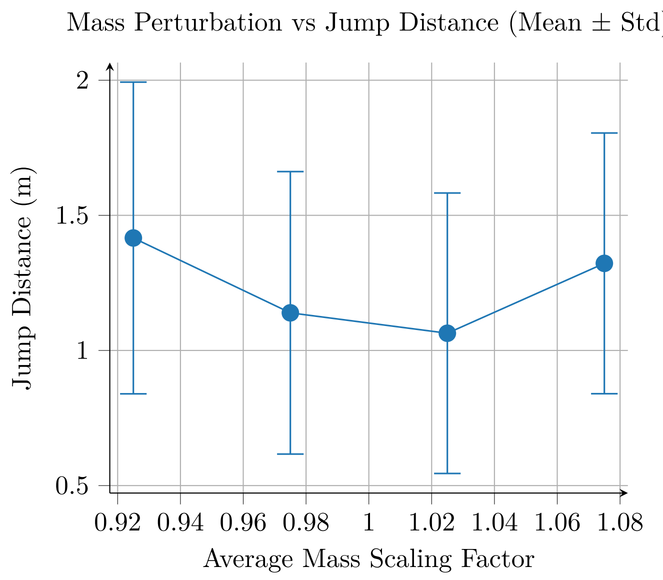


Figure 6.15.: Mass Fluctuation vs Distance covered by the humanoid

Torque Fluctuations

The torque fluctuations had a more nuanced impact on the humanoids performance. The policy was unable to effectively adapt to the randomized gear ratios, which resulted in a marked decrease in jump distance. The average distance covered dropped to approximately 1m, comparable to that of the expert policy. This indicates that the humanoid could not compensate for variability in joint torques, leading to a weaker push-off phase and reduced forward momentum.

Landing stability was also significantly affected. The success rate declined to just 3%, suggesting that the policy struggled to maintain balance under fluctuating torque conditions. The disruptions in actuator consistency appeared to interfere with the learned motion patterns, forcing the humanoid to adopt less effective strategies for both takeoff and landing. This behavior was reflected in erratic joint trajectories and inconsistent foot placements, which further contributed to instability upon ground contact.

Overall, these results underscore the critical importance of consistent actuator performance for reliable and effective humanoid locomotion. Even small variations in torque production can destabilize learned behaviors and sharply reduce both motion quality and task success. Refer to Figure 6.16 for a comparison of jump distance under varying gear ratios.

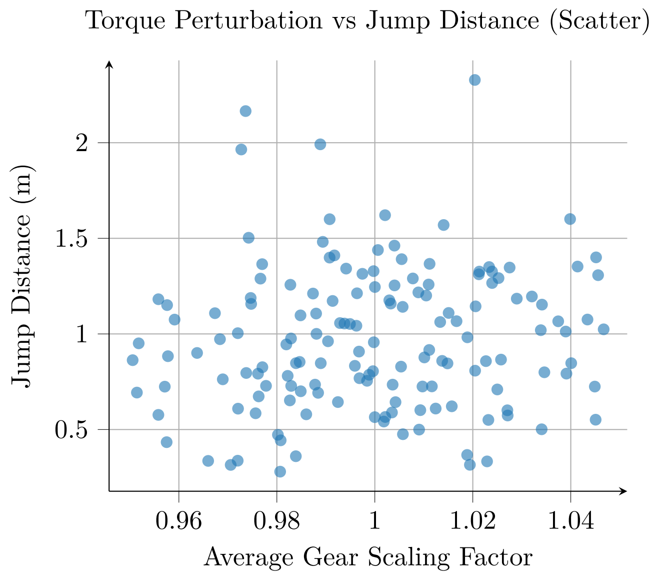


Figure 6.16.: Torque Fluctuation vs Distance covered by the humanoid

Discussion

The perturbation analysis highlights important insights into the robustness and limitations of the learned policy. Mass fluctuations revealed that the humanoid could exploit reductions in body weight to generate greater relative torque and achieve substantially longer jumps. However, this adaptation came at the cost of post-landing stability, with success rates dropping drastically. This mirrors biomechanical trade-offs observed in natural systems, where lighter bodies can often jump farther but face greater challenges in safely absorbing impact forces.

In contrast, torque perturbations disrupted performance more fundamentally. The randomized gear ratios degraded both takeoff and landing behaviors, reducing jump distance and sharply decreasing stability. Unlike mass fluctuations, which the policy could partially exploit, torque fluctuations directly interfered with the consistency of force generation, making it difficult for the agent to retain learned coordination strategies. This suggests that reliable actuator performance is essential for maintaining motion quality in RL based locomotion.

Taken together, these results underline the dual challenge of robustness: policies must not only adapt to beneficial perturbations but also withstand detrimental ones without collapsing performance. For future work, this points toward incorporating robustness objectives or domain randomization into training, enabling the policy to better generalize to variations in both morphology (mass) and actuation (torque), while maintaining biomechanical plausibility and stability.

6.3. Summary

In Summary, the humanoid policy successfully learned to imitate expert jumping motion while optimizing for forward displacement and stability. The combination of imitation learning and reward shaping enabled the agent to capture the essential kinematic patterns of the expert while adapting to the task-specific requirements of maximizing horizontal distance. The policy demonstrated robust performance across various perturbation scenarios, showing adaptability to changes in mass and torque while maintaining a stable landing strategy. The ablation study confirmed the importance of each reward component in shaping both performance and motion style, highlighting the necessity of a multi-faceted reward formulation for effective explosive motion imitation. The analysis of GRF profiles, energy efficiency, and landing precision further validated the biomechanical plausibility of the learned motion, indicating that the humanoid employs realistic force generation and impact absorption strategies. Overall, the results demonstrate that the humanoid has achieved a high level of proficiency in explosive jumping motion, balancing imitation accuracy with task-specific adaptation, and laying the groundwork for future advancements in humanoid robotics and locomotion control.

7 Conclusion

This thesis has explored the application of RL to enhance explosive motions in simulated humanoid agents, with a focus on the standing long jump as a representative task. By integrating MoCap data for imitation guidance with a carefully designed reward function and PPO, we developed a framework that not only replicates human-like behaviors but also optimizes them for improved performance, such as greater jump distance and landing stability. This work addresses key challenges in humanoid robotics, including the need for agile, energy-efficient control strategies that extend beyond static imitation to adaptive, task-oriented motion synthesis.

7.1. Summary of Key Findings and Contributions

The approach builds on foundational RL techniques (Chapter 3) and related advancements in physics-based imitation and biomechanics (Chapter 2). The experimental setup in MuJoCo (Chapter 4) provided a robust platform for training a 28-DoF humanoid model, where the policy learned to execute jumps through a combination of imitation rewards (tracking pose, velocity, end-effectors, and center of mass) and enhancement rewards (promoting squat depth, leg extension, symmetry, takeoff velocity, and post-landing uprightness; Chapter 5).

Quantitative and qualitative evaluations (Chapter 6) demonstrated significant improvements over the baseline MoCap reference. The learned policy achieved an average jump distance of up to 2.5m under mass perturbations, surpassing the expert's 1.4m, while maintaining low tracking errors in joint velocities and accelerations. Biomechanical metrics, such as ground reaction force profiles and foot symmetry, confirmed that the motion remained plausible and human-like, with smoother landing transitions despite elevated jerk during takeoff. Robustness tests under mass and torque fluctuations revealed the policy's adaptability, though with trade-offs in landing success rates (e.g., dropping to 8% under mass variations and 3% under torque variations). Ablation studies validated the reward components' roles, showing that phase-specific terms (e.g., squat and takeoff velocity rewards) were critical for explosive performance, while symmetry and stability rewards ensured stylistic fidelity.

These results contribute to the field by demonstrating how RL can transcend mere imitation to foster emergent optimizations, such as exploiting reduced mass for longer jumps or prioritizing landing stability. The curriculum-based reward weighting scheme proved effective in transitioning from imitation-focused early training to performance-driven refinement, offering a scalable method for other dynamic tasks.

7.2. Limitations

Despite these advancements, several limitations warrant acknowledgment. The policy exhibited overtraining effects after 20 million episodes, leading to degraded performance in initial tasks, which suggests the need for more sophisticated regularization techniques. Energy efficiency remained suboptimal, with the humanoid incurring higher control costs than the expert, highlighting a trade-off between explosiveness and sustainability. Robustness under perturbations, while promising, was inconsistent, particularly for torque fluctuations—indicating that the current domain randomization may not fully capture real-world variability. Additionally, the framework was tested solely on a simulated standing long jump; generalization to diverse motions or real hardware was not evaluated, potentially limiting transferability due to the sim-to-real gap.

7.3. Future Work

Future research could extend this framework in several directions to address these limitations and broaden its impact. First, incorporating multi-task learning could enable the policy to handle a repertoire of explosive motions (e.g., high jumps, sprints, or directional changes), potentially using hierarchical RL to compose primitives from MoCap data. Second, advanced robustness techniques, such as adaptive domain randomization or metaRL, could improve resilience to morphological and environmental perturbations, facilitating sim-to-real transfer through techniques like system identification or hardware-in-the-loop training.

To enhance energy efficiency, reward functions could integrate explicit biomechanical priors, such as muscle activation models or metabolic cost penalties, drawing from human jumping studies. Diffusion-based models or adversarial imitation could further refine motion diversity, allowing stochastic variations for more naturalistic behaviors. Addition of toes for propulsion of motion can also be thought of as a viable addition to the project, keeping the biomechanical accuracy of the humanoid. This can potentially reduce the amount of torque required by the ankle joints of the humanoid, mitigating some effects to the toes. Finally, deploying the policy on physical humanoid platforms (e.g., Atlas or Digit) would validate its real-world applicability, with evaluations focusing on hardware constraints like torque limits and sensor noise.

In conclusion, this thesis advances the synthesis of explosive humanoid motions by blending imitation learning with RL-driven enhancement, paving the way for more capable and adaptable robotic systems. By pushing beyond reference trajectories, the approach underscores the potential of learning-based methods to unlock novel behaviors, ultimately contributing to agile robotics in dynamic, unstructured environments.

Bibliography

Ashby, B. M. and J. H. Heegaard (2002): "Role of arm motion in the standing long jump". In: *Journal of Biomechanics* 35.12, pp. 1631–1637. URL: `https://www.sciencedirect.com/science/article/pii/S0021929002002397`.

Baldwin, M. A., C. Clary, L. P. Maletsky, and P. J. Rullkoetter (2009): "Verification of predicted specimen-specific natural and implanted patellofemoral kinematics during simulated deep knee bend". In: *Journal of Biomechanics* 42.14, pp. 2341–2348. URL: `https://www.sciencedirect.com/science/article/pii/S0021929009003625`.

Al-Hafez, F., G. Zhao, J. Peters, and D. Tateo (2023): *LocoMuJoCo: A Comprehensive Imitation Learning Benchmark for Locomotion.* arXiv: `2311.02496 [cs.LG]`. URL: `https://arxiv.org/abs/2311.02496`.

Heess, N., D. TB, S. Sriram, J. Lemmon, J. Merel, G. Wayne, Y. Tassa, T. Erez, Z. Wang, S. M. A. Eslami, M. A. Riedmiller, and D. Silver (2017): "Emergence of Locomotion Behaviours in Rich Environments". In: *CoRR* abs/1707.02286. arXiv: `1707.02286`. URL: `http://arxiv.org/abs/1707.02286`.

Kingma, D. P. and J. Ba (2017): *Adam: A Method for Stochastic Optimization.* arXiv: `1412.6980 [cs.LG]`. URL: `https://arxiv.org/abs/1412.6980`.

Liu, J., Z. Li, M. Yu, Z. Dong, S. Calinon, D. Caldwell, and F. Chen (2024): *Human-Humanoid Robots Cross-Embodiment Behavior-Skill Transfer Using Decomposed Adversarial Learning from Demonstration.* arXiv: `2412.15166 [cs.RO]`. URL: `https://arxiv.org/abs/2412.15166`.

Merel, J., Y. Tassa, D. TB, S. Srinivasan, J. Lemmon, Z. Wang, G. Wayne, and N. Heess (2017): "Learning human behaviors from motion capture by adversarial imitation". In: *CoRR* abs/1707.02201. arXiv: `1707.02201`. URL: `http://arxiv.org/abs/1707.02201`.

Miller, A., S. Fahmi, M. Chignoli, and S. Kim (2023): *Reinforcement Learning for Legged Robots: Motion Imitation from Model-Based Optimal Control.* arXiv: `2305.10989 [cs.RO]`. URL: `https://arxiv.org/abs/2305.10989`.

Pearce, T., T. Rashid, A. Kanervisto, D. Bignell, M. Sun, R. Georgescu, S. V. Macua, S. Z. Tan, I. Momennejad, K. Hofmann, and S. Devlin (2023): *Imitating Human Behaviour with Diffusion Models.* arXiv: `2301.10677 [cs.AI]`. URL: `https://arxiv.org/abs/2301.10677`.

Peng, X. B., P. Abbeel, S. Levine, and M. van de Panne (2018): "DeepMimic: Example-Guided Deep Reinforcement Learning of Physics-Based Character Skills". In:

CoRR abs/1804.02717. arXiv: 1804.02717. URL: http://arxiv.org/abs/1804.02717.

Peng, X. B., Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa (2021): "AMP: Adversarial Motion Priors for Stylized Physics-Based Character Control". In: *CoRR* abs/2104.02180. arXiv: 2104.02180. URL: https://arxiv.org/abs/2104.02180.

Pintrest (n.d.): *Standing Long Jump Phases.* Available under: https://i.pinimg.com/564x/b1/fe/45/b1fe450c66c82cecda2bcf7926993eaa.jpg.

Schulman, J., F. Wolski, P. Dhariwal, A. Radford, and O. Klimov (2017): "Proximal Policy Optimization Algorithms". In: *CoRR* abs/1707.06347. arXiv: 1707.06347. URL: http://arxiv.org/abs/1707.06347.

Sutton, R. S. and A. G. Barto (2014): *Introduction to Reinforcement Learning.* MIT Press. URL: https://www.andrew.cmu.edu/course/10-703/textbook/BartoSutton.pdf.

Tang, A., T. Hiraoka, N. Hiraoka, F. Shi, K. Kawaharazuka, K. Kojima, K. Okada, and M. Inaba (May 2024): "HumanMimic: Learning Natural Locomotion and Transitions for Humanoid Robot via Wasserstein Adversarial Imitation". In: *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 13107–13114. URL: http://dx.doi.org/10.1109/ICRA57147.2024.10610449.

Wen-lanwu, Jia-hroungwu, Hwai-tinglin, and Gwo-jawwang (Apr. 2012): "BIOMECHANICAL ANALYSIS OF THE STANDING LONG JUMP". In: *Biomedical Engineering: Applications, Basis and Communications* 15. URL: https://www.worldscientific.com/doi/abs/10.4015/S1016237203000286.

Yan, Y., E. V. Mascaro, T. Egle, and D. Lee (2025): *I-CTRL: Imitation to Control Humanoid Robots Through Constrained Reinforcement Learning.* arXiv: 2405.08726 [cs.RO]. URL: https://arxiv.org/abs/2405.08726.

Yang, C., K. Yuan, W. Merkt, T. Komura, S. Vijayakumar, and Z. Li (2020): "Learning Whole-body Motor Skills for Humanoids". In: *CoRR* abs/2002.02991. arXiv: 2002.02991. URL: https://arxiv.org/abs/2002.02991.

A Appendix

Below are the hyperparameters used in the network. Refer to Section 4.6 for more details.

Table A.1.: Hyperparameters

Symbol	Value
T	2048
n_{steps}	3,000,000
γ	0.95
λ	0.95
ϵ	0.2
n_{epochs}	8
w_{net}	512
w_{hid}	256
α_{π}	5×10^{-5}
α_{V}	3×10^{-4}
β	0.001
n_{batch}	256
n_{mini}	10
$\log \sigma$	-1.0
c_1	0.001
ζ	0.99

Below are the specifications of the humanoid model used in the simulation. Refer to Section 4.4 for more details.

Table A.2.: Humanoid Joint Configuration

Joint	Degrees of Freedom	Range
Hips	3	[200, 400, 200]
Knees	1	400
Ankles	3	[90, 200, 90]
Shoulders	3	[100, 100, 100]
Elbows	1	60
Chest	3	[200, 200, 200]
Neck	3	[50, 50, 50]

Table A.3.: Joint Limits

Joint	Limit (rad)
Hip_x	$[-1.2,\ 1.2]$
Hip_y	$[-2.57,\ 1.57]$
Hip_z	$[-1.0,\ 1.0]$
Knee	$[-2.7,\ 0]$
Ankle_x	$[-1.0,\ 1.0]$
Ankle_y	$[-1.0,\ 1.57]$
Ankle_z	$[-1.0,\ 1.0]$
Shoulder_x	$[-0.5,\ 3.14]$
Shoulder_y	$[-3.14,\ 0.7]$
Shoulder_z	$[-1.5,\ 1.5]$
Elbow	$[0,\ 2.8]$
Chest_x	$[-1.2,\ 1.2]$
Chest_y	$[-1.2,\ 1.2]$
Chest_z	$[-1.2,\ 1.2]$
Neck_x	$[-1.0,\ 1.0]$
Neck_y	$[-1.0,\ 1.0]$
Neck_z	$[-1.0,\ 1.0]$

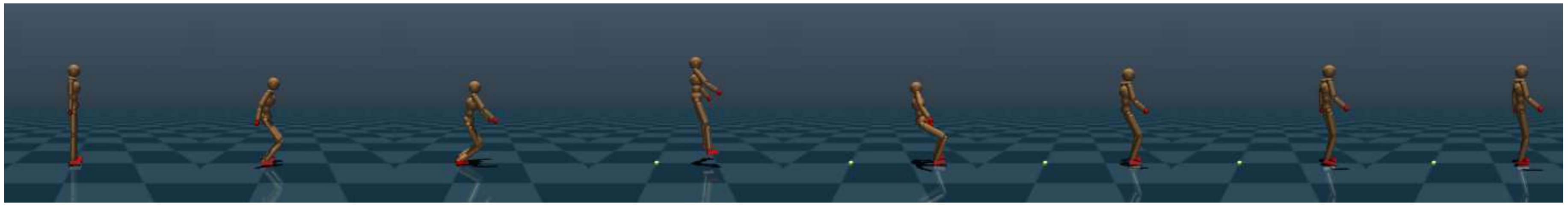


Figure A.1.: Above is the full motion trail for the successful humanoid performing a standing long jump in the simulation. The performance of the successful policy including stability post jump. Refer to Section 6 for more details.

Usage of generative AI - Affidavit

☐ not at all

☒ for correcting, optimizing, or restructuring the entire work (This eliminates the need for explicit marking of individual passages or sections, as this type of usage refers to the entire written work. Explicit marking in the text is not necessary, as this serves as the global indication.)

☐ Code optimization: Optimization or restructuring of software function

☐ Code generation: Creating entire software functions from a detailed functional description.

☐ Substance generation in code: Generating entire software source code

☐ Media optimization: Correction, optimization, or restructuring of entire passages

☐ Media generation: Creating entire passages from given content.

☐ Substance generation in media: Generating entire sections

☐ More, namely:

I assure that I have provided all usages completely. Missing or incorrect information may be considered an attempt to deceive.

Dortmund, 01.07.2025
Place, Date

[illegible]
Shubhankar Kulkarni

Fakultät für
Elektrotechnik und
Informationstechnik

Lehrstuhl für
Regelungssystemtechnik

Master Thesis (A&R): Enhancing Explosive Motion in Humanoid Robotics with Imitation and Reinforcement Learning
Shubhankar Kulkarni (243453)

Background

Explosive motion—characterized by rapid, high-energy movements such as sprinting, jumping, and sudden directional changes—is essential for human performance in sports, search and rescue operations, and dynamic physical tasks. These movements rely on efficient force production and precise coordination, making their replication in humanoid robots a crucial step toward improving agility and responsiveness in real-world applications.

This thesis aims to develop and optimize strategies for explosive locomotion in humanoids using Reinforcement Learning (RL) and Imitation Learning (IL). By leveraging MuJoCo as a simulation environment, the study will explore how humanoid robots can learn and execute high-energy movement patterns, with potential future extensions to manipulation tasks.

The thesis addresses two main research questions:

How can reinforcement learning and imitation learning be leveraged to develop effective strategies for explosive motion in humanoid robots?

What methods can be used to stimulate and optimize high-energy motion patterns in simulated humanoid environments?

Methodology and Approach

This study will employ a combination of reinforcement learning and imitation learning techniques to train humanoid models in MuJoCo. The key aspects of the methodology include:

1. Imitation Learning Techniques
   Using behavior cloning and generative adversarial imitation learning (GAIL) to train models from expert demonstrations.
2. Reward Function Design:
   Developing and optimizing reward functions that promote explosive motion and maximize high-energy movement efficiency.
3. Large-Scale Training and Testing
   Evaluating humanoid motion under diverse environmental conditions to enhance adaptability and robustness.
4. Performance Comparison

Assessing the effectiveness of learned policies against baseline locomotion techniques to measure efficiency and versatility.

5. Advanced Motion Strategies (optional)
   Exploring additional factors such as toe-based propulsion and biomechanically inspired designs to improve real-world applicability.

Expected Outcomes

1. A humanoid model capable of executing explosive locomotion, with possibilities for expansion into manipulation tasks.
2. A detailed evaluation of different reward function designs and their impact on reinforcement learning algorithms.
3. A comparative analysis of imitation learning and reinforcement learning approaches in developing effective control policies.
4. A comprehensive thesis documenting the methodology, experimental results, insights, and future directions in humanoid explosive motion research.

References

Issue date: 01.03.2025
Submission date: 01.09.2025

[illegible]

Univ.-Prof. Dr.-Ing. Prof. h.c. Dr. h.c. Torsten Bertram

[illegible]

apl. Prof. Dr. rer. nat. Frank Hoffmann

[illegible]

Shubhankar Kulkarni

Eidesstattliche Versicherung

(Affidavit)

Kulkarni, Shubhankar
Name, Vorname
(surname, first name)

243453
Matrikelnummer
(student ID number)

☐ Bachelorarbeit
(Bachelor's thesis)

☒ Masterarbeit
(Master's thesis)

Titel
(Title)

Enhancing Explosive Motion in Humanoid Robotics with Imitation and Reinforcement Learning

Ich versichere hiermit an Eides statt, dass ich die vorliegende Abschlussarbeit mit dem oben genannten Titel selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie wörtliche und sinngemäße Zitate kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

I declare in lieu of oath that I have completed the present thesis with the above-mentioned title independently and without any unauthorized assistance. I have not used any other sources or aids than the ones listed and have documented quotations and paraphrases as such. The thesis in its current or similar version has not been submitted to an auditing institution before.

Dortmund, 01.09.2025
Ort, Datum
(place, date)

Unterschrift [illegible]
(signature)

Belehrung:
Wer vorsätzlich gegen eine die Täuschung über Prüfungsleistungen betreffende Regelung einer Hochschulprüfungsordnung verstößt, handelt ordnungswidrig. Die Ordnungswidrigkeit kann mit einer Geldbuße von bis zu 50.000,00 € geahndet werden. Zuständige Verwaltungsbehörde für die Verfolgung und Ahndung von Ordnungswidrigkeiten ist der Kanzler/die Kanzlerin der Technischen Universität Dortmund. Im Falle eines mehrfachen oder sonstigen schwerwiegenden Täuschungsversuches kann der Prüfling zudem exmatrikuliert werden. (§ 63 Abs. 5 Hochschulgesetz - HG -).

Die Abgabe einer falschen Versicherung an Eides statt wird mit Freiheitsstrafe bis zu 3 Jahren oder mit Geldstrafe bestraft.

Die Technische Universität Dortmund wird ggf. elektronische Vergleichswerkzeuge (wie z.B. die Software „turnitin") zur Überprüfung von Ordnungswidrigkeiten in Prüfungsverfahren nutzen.

Die oben stehende Belehrung habe ich zur Kenntnis genommen:

Official notification:
Any person who intentionally breaches any regulation of university examination regulations relating to deception in examination performance is acting improperly. This offense can be punished with a fine of up to EUR 50,000.00. The competent administrative authority for the pursuit and prosecution of offenses of this type is the Chancellor of TU Dortmund University. In the case of multiple or other serious attempts at deception, the examinee can also be unenrolled, Section 63 (5) North Rhine-Westphalia Higher Education Act (*Hochschulgesetz, HG*).

The submission of a false affidavit will be punished with a prison sentence of up to three years or a fine.

As may be necessary, TU Dortmund University will make use of electronic plagiarism-prevention tools (e.g. the "turnitin" service) in order to monitor violations during the examination procedures.

I have taken note of the above official notification:*

Dortmund, 01.09.2025
Ort, Datum
(place, date)

Unterschrift [illegible]
(signature)

***Please be aware that solely the German version of the affidavit ("Eidesstattliche Versicherung") for the Bachelor's/ Master's thesis is the official and legally binding version.**