Towards Robust and Generalized Quadruped Locomotion in the Physical World

Dikai Liu

College of Computing and Data Science

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

2026

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

28 July 2025
.....
Date

.....
Dikai Liu

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiargism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

28 July 2025

Date

Assoc Prof Tianwei Zhang

Authorship Attribution Statement

This thesis contains materials from 3 papers accepted at conferences and 1 technical report preprint, in which I am listed as an author.

Chapter 3 is published as: Dikai Liu, Tianwei Zhang, Jianxiong Yin, and Simon See. "Masked Sensory-Temporal Attention for Sensor Generalization in Quadruped Locomotion". In IEEE International Conference on Robotics and Automation (ICRA), 2025.

The contributions of the co-authors are as follows:

- I was the lead author. I wrote the manuscript draft and conducted all the experiments.
- Prof. Tianwei Zhang guided the initial research direction and revised the manuscript draft.
- The rest of the co-authors participated in the discussion.

Chapter 4 is published as: Dikai Liu, Jianxiong Yin, and Simon See. "Towards Fault-tolerant Quadruped Locomotion with Reinforcement Learning". In IEEE Conference on Artificial Intelligence (CAI), 2024.

An extended technical report is available as: Dikai Liu, Tianwei Zhang, Jianxiong Yin, and Simon See, "Saving the Limping: Fault-Tolerant Quadruped Locomotion via Reinforcement Learning", arXiv preprint arXiv:2210.00474.

The contributions of the co-authors are as follows:

- I was the lead author. I wrote the manuscript draft and conducted all the experiments.
- Prof. Tianwei Zhang guided the initial research direction and revised the manuscript draft.
- The rest of the co-authors participated in the discussion.

Chapter 5 is published as: Dikai Liu, Tianwei Zhang, Jianxiong Yin, and Simon See. "Unified Locomotion Transformer with Simultaneous Sim-to-Real Transfer for Quadrupeds". In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2025.

The contributions of the co-authors are as follows:

- I was the lead author. I wrote the manuscript draft and conducted all the experiments.
- Prof. Tianwei Zhang guided the initial research direction and revised the manuscript draft.

- The rest of the co-authors participated in the discussion.

28 July 2025

Date

Dikai Liu

Acknowledgements

As my journey in pursuing a Ph.D. comes to its end, I would like to convey my heartfelt gratitude to all the people who have been supporting and encouraging me throughout this experience.

First and foremost, I would like to express my deepest gratitude to my academic supervisor, Prof. Tianwei Zhang, for his invaluable guidance and intellectual mentorship throughout my doctoral studies. I am equally grateful to my industry supervisors, Dr. Simon See for his continuous advice and encouragement. Their expertise, patience, and dedication have been instrumental in shaping my research direction and fostering my growth as an independent researcher. I would thank the opportunity given by EDB-IPP to make this journey happen by bridging the gap between academic research and industry innovation.

I would like to extend my sincere appreciation to my NVIDIA colleague Mr. Jianxiong Yin, and manager Dr. Aik Beng Ng, for their exceptional support in balancing my work responsibilities and research commitments. Their understanding, flexibility, and encouragement have made it possible for me to pursue my doctoral studies while contributing to meaningful industry projects. The opportunity to work within such a supportive environment has been invaluable to my professional and academic development.

I would also like to extend my thanks to all the remarkable individuals I have encountered in the univeristy, workplace, and various venues during this journey. For my lab colleagues, thank you for your insightful discussions and creating our shared moments of discovery. To my workspace friends and collaborators, thank you for creating an environment of learning and innovation. To everyone who has engaged in meaningful conversations, shared insights, or simply provided moments of joy during this journey, if you are reading this and wondering if you are included, yes, you are. In the spirit of friendship and to stop the fighting, I create this wonderful Every Friend as First Friend following [1]:

[redacted]rence

Finally, I would like to express my profound gratitude to my parents and family for their unconditional love, endless support, and unwavering belief in my abilities. Their encouragement and understanding throughout the many years of my academic pursuit have been the foundation upon which this achievement stands and this work would not have been possible without their constant support and patience.

To everyone, thank you for being a part of this remarkable experience.

Dikai Liu

Better to run than curse the road.

Contents

Abstract

Physical AI systems represent a transformative paradigm for creating autonomous agents capable of perceiving, reasoning, and interacting within complex real-world environments. Among various robotic platforms, multi-legged systems, for example quadrupedal robots, have gained significant attention due to their exceptional capability to traverse rough and complex terrains with high agility, while simultaneously maintaining substantial payload carrying capacities for continuous operation. Locomotion is a fundamental skill in multi-legged systems that draws extensive research interest, as it enables the deployment of quadrupedal platforms in challenging real-world environments to provide a stable foundation for critical missions. However, the development of robust locomotion remains fundamentally constrained by challenges in environmental generalization, hardware reliability, and the persistent simulation-to-reality transfer gap. This thesis addresses these critical limitations through novel methodologies specifically designed for quadrupedal locomotion, advancing robust and generalizable quadruped locomotion agent through efficient simulation-based training approaches utilizing reinforcement learning (RL) and knowledge transfer.

The research confronts the inherent brittleness of simulation-first learning-based approaches, where the physical dynamics in unpredictable environments deviate significantly from the simulated training conditions. Traditional robotic control systems often exhibit catastrophic performance degradation when faced with hardware malfunctions, sensor failures, or environmental variations, severely limiting practical deployment in mission-critical applications. To overcome these fundamental challenges, this thesis presents three primary contributions that collectively advance robust and generalized quadrupedal locomotion through innovative simulation-based training methodologies.

First, we propose the Masked Sensory-Temporal Attention (`MSTA`) mechanism, a transformer-based architecture that addresses the sensor generalization challenges in quadrupedal locomotion. Unlike conventional approaches that rely on fixed sensor configurations, `MSTA` employs attention mechanisms to fuse available sensory

information while compensating for missing inputs. The framework tokenizes individual sensor modalities and channels with random masking during training, enabling robust performance across different sensor configurations and hardware platforms for generalized and fixable deployment.

Second, we introduce a comprehensive Fault-Tolerant Locomotion (`FTL`) framework that enables quadrupedal robots to maintain stable locomotion despite hardware failures, particularly joint locking scenarios. With the simulation strategies designed, dynamic failure situations can be added randomly to enhance the diversity of training scenarios. Through a teacher-student reinforcement learning paradigm with additional domain randomization, our approach demonstrates the capability to dynamically adapt locomotion patterns when joints face hardware failure, achieving zero-shot transfer from simulation to physical robots.

Third, we develop the Unified Locomotion Transformer (`ULT`) architecture, facilitating simultaneous optimization of teacher and student policies for effective knowledge transfer and simulation-to-reality transfer. Unlike traditional sequential training approaches that require separate phases of knowledge distillation, `ULT` integrates privileged information and proprioceptive observations within a unified transformer framework. The architecture employs action-mixing strategies, combining reinforcement learning loss with next state-action prediction and policy imitation objectives for a single-stage training.

Comprehensive experimental validation includes simulation studies using Isaac Gym and real-world validation on Unitree A1 quadrupedal platforms. The results demonstrate substantial improvements in robust fault-tolerance capabilities with successful locomotion across diverse failure scenarios, sensor generalization that maintains consistent performance across varying observation configurations, efficient training through unified frameworks that allow simultaneous development of teacher and student agents. Additionally, all the agents can be zero-shot sim-to-real transferred to physical robots without fine-tuning using real-world data.

The implications extend beyond quadrupedal locomotion, providing fundamental insight for developing robust physical AI systems in diverse robotic applications. By addressing core challenges in fault tolerance, sensor generalization, and knowledge transfer, this thesis contributes essential building blocks towards generalizable

physical AI systems capable of reliable operation in complex, dynamic real-world environments.

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

1.1.1 Legged Locomotion in the Era of Physical AI

Physical AI represents a transformative paradigm in artificial intelligence, fundamentally shifting the field from purely computational systems to embodied agents capable of perceiving, reasoning, and interacting within dynamic physical environments [5, 6]. This emerging discipline integrates advanced artificial intelligence with autonomous physical systems, creating intelligent agents that must seamlessly bridge the gap between digital- and physical-world, to advance toward artificial general intelligence (AGI). The development of physical AI systems typically begins with the mastery of fundamental capabilities such as locomotion, which serve as essential building blocks for more complex autonomous behaviors and real-world deployment capabilities [3, 7–10].

Multi-legged robots, especially quadrupedal systems, provide unmatched mobility over rugged, cluttered, and steep terrain that wheeled or tracked platforms suffer [8] and has become the iconic platform in the era of physical AI. Their stable gaits enable them to transport sizable sensing and manipulation payloads while withstanding external force from slips and pushes [3, 8]. In order to carry out critical missions, robust locomotion capability is an non-negotiable capability, which all higher-level behaviors are built upon, even for advanced vision-language-action (VLA) models [11, 12].

1.1.2 Simulation-first Learning-based Agents with Persistent Sim-to-real Gap

The evolution of autonomous AI systems reflects a fundamental paradigm shift from classical, model-based control methodologies to adaptive, learning-based approaches capable of handling complex, unstructured environments. Classical autonomous systems relied heavily on pre-programmed hand-crafted control strategies based on extensive human prior knowledge, such as Model Predictive Control (MPC) [13–15] integrated with Quadratic Programming (QP) solvers [16–18]. Although these classical approaches demonstrated remarkable precision and reliability in controlled industrial environments with well-defined dynamics, they increasingly exhibit fundamental limitations in dynamic, unstructured settings where model uncertainty and environmental variability challenge their assumptions, especially with significant breakthroughs in multi-legged robots like quadrupeds and humanoids and the deployment of intelligent physical systems in real-world applications [3, 8, 19]. This paradigm shift has necessitated the transition from rule-based control systems to learning-based approaches, such as reinforcement learning (RL), which can adapt and generalize to complex, previously unseen scenarios through experience and data-driven optimization, eliminating the need for extensive prior knowledge and manual parameter tuning [3, 4, 7–9, 20].

The development of advanced simulation technologies [2, 21, 22] capable of creating photorealistic and physically accurate virtual environments has fundamentally revolutionized physical AI development through a simulation-first paradigm that enables massively parallel training architectures, allowing thousands of virtual agents to execute simultaneously, reducing policy development from days or weeks to a mere hours [7, 8, 23, 24]. These modern simulation platforms provide safe, controlled, and computationally scalable environments that have made it economically and practically feasible to train increasingly complex robotic behaviors without expensive, time-consuming, and potentially dangerous real-world data collection, thus simulation environments have become the primary development platform for advanced embodied AI systems ranging from agile quadrupeds to complex humanoid systems [3, 7, 20].

Despite remarkable advances in simulation fidelity and physics modeling, inevitable discrepancies between virtual environments and physical world dynamics create

the fundamental simulation-to-reality (sim-to-real) gap, which remains a critical challenge limiting seamless transfer of simulation-trained agents to real-world deployment [25–28]. While various techniques have been developed to address this challenge—including domain randomization strategies that inject controlled randomness during training [7, 25, 26, 29], domain adaptation methods that bridge simulation-reality differences with real-world data [30, 31], and knowledge distillation frameworks leveraging privileged simulation information [3, 8, 9], the gap persists as a fundamental bottleneck, particularly under increasing environmental uncertainty with hardware degradation and extreme operating conditions where robotic agents encounter unexpected perturbations that challenge simulation-based training paradigms, especially for critical low-level capabilities like locomotion.

1.1.3 Robustness and Generalization in Locomotion

While the reliability and robustness of software-centric AI models have been extensively studied [32, 33], physical robots face compounded challenges when deployed in untrusted environments that go beyond algorithmic limitation. These systems must also address hardware-level vulnerabilities caused by unpredictable real-world conditions, including actuator wear and sensor failures due to human-robot interaction or environmental interference [34]. Physical AI agents must simultaneously maintain operational integrity, requiring resilience strategies for fault detection, adaptive control architectures, and robust sensor fusion, which are still underdeveloped in current RL-based paradigms for low-level locomotion control.

Effective deployment of physical AI systems requires addressing these fundamental robustness challenges before we can fully leverage their high-level capabilities. The recent surge in research on embodied AI has primarily emphasized task planning, reasoning, and interaction capabilities [5, 11, 35], often overlooking the critical importance of low-level system robustness. As cutting-edge foundation models enable increasingly complex high-level behaviors [36, 37], the gap between advanced cognitive capabilities and fundamental operational reliability widens. This imbalance creates systems that can reason about complex tasks, but may fail catastrophically when facing basic hardware or sensor issues, undermining their practical utility in real-world deployment.

Although learning-based agents have demonstrated remarkable capabilities in different domains, current approaches based on deep neural networks suffer from several critical limitations in generalization that hinder their widespread real-world deployment [38] for basic locomotion skills. With the growing diversity of robotic platforms on the market, each with different hardware designs, sensor suites, and capabilities, some early attempts have been made for cross-morphology [39] or cross-embodiment [40, 41]. However, these methods are not easy to scale out with fixed input and output dimensions defined in convolutional neural network (CNN) and multi-layer perceptron (MLP), making them incapable of adapting to changes in sensor configurations. This inflexibility becomes particularly problematic in real-world scenarios where sensor failures or new sensor modalities are needed.

1.1.4 The Rise of Transformer Paradigms

Transformer architectures have shown promising results to address these generalization challenges through their attention mechanisms and token-based processing [9, 42, 43] and has started a new era of unified frameworks in Natural Language Processing (NLP) [42, 44, 45], Computer Vision (CV) [46–48] and multimodal process [49–52], which is ideal to handle multi-sensor fusion in robotic applications [36, 43, 53, 54].

However, transformers introduce their own set of challenges when applied to real-time robotic control. Although transformer-based VLA models [11, 36, 37, 43, 54–57] have recently demonstrated remarkable progress in multimodal information processing for sensor fusing, high-level task reasoning and language grounding with human-robot interaction (HRI), their application to real-time low-level control remains fundamentally constrained by computational demands and latency requirements VLAs also remain fundamentally constrained by persistent challenges in action robustness under hardware failures, observation generalization across diverse sensor configurations, and learning efficiency for training adaptive agents capable of zero-shot transfer from simulation to reality.

Furthermore, most transformer-based RL agents are now trained with the teacher-student paradigm, which is promising for leveraging privileged information by first training the teacher with access to all needed information and then distilling this knowledge into student policies that operate with limited observations [3, 8, 9] and

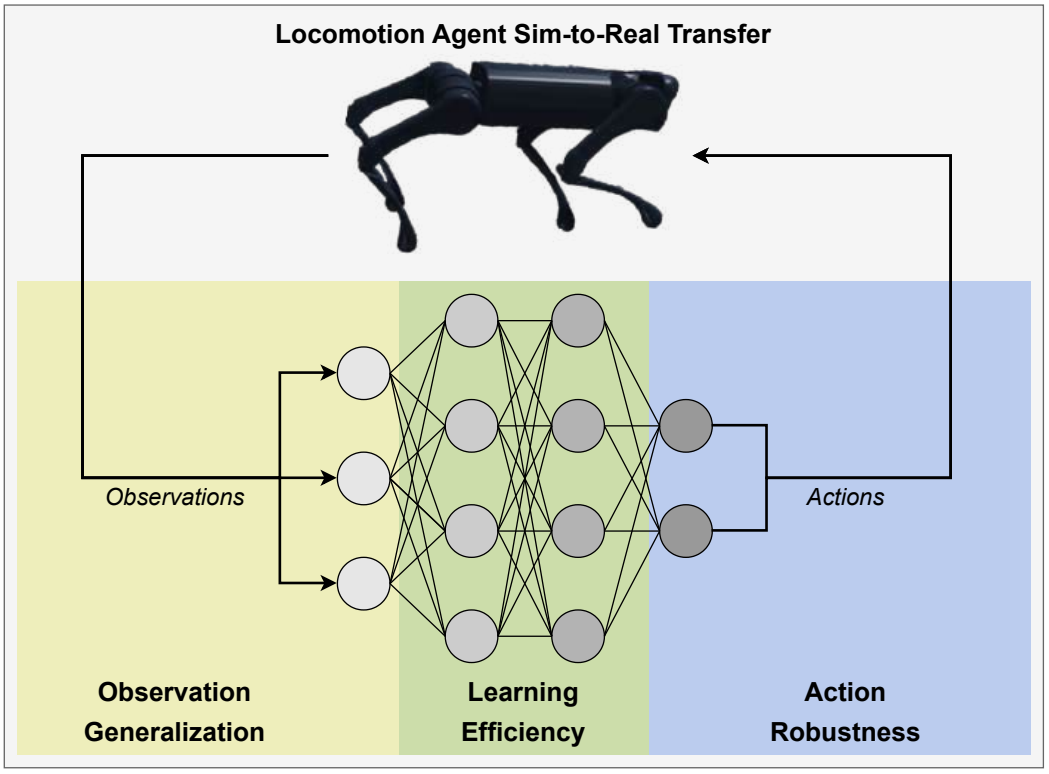


FIGURE 1.1: An overview of the research topics discussed in this thesis, while specifically applied to quadruped locomotion with sim-to-real transfer, are representative of broader challenges in physical AI across a variety of embodiments.

with new trajectories and associated action labels of the teacher created through algorithms such as Data Aggregator (DAgger) [58]. However, this requires multi stage of training and the student policy had limited self-exploration, which limits the overall training efficiency and performance, especially in corner cases where students cannot effectively imitate due to their limited observational capabilities and lack of self-exploration.

1.2 Motivation

The motivation of this thesis stems from the critical need to bridge the sim-to-real gap to achieve zero-shot deployment of simulation-first learning-based agents in physical environments and the the imbalance between research on high-level reasoning and fundamental low-level skills such as locomotion in the era of physical AI. For any learning-based agent, it generates desired actions based on sensor

input processed through a trained neural network, and these agents require action robustness and observation generalization while commonly being trained using simulation only, as illustrated in Figure 1.1. Although current robotic agents have demonstrated impressive capabilities across various tasks, they still face significant challenges when deployed in dynamic and uncertain environments, where robustness and adaptability are crucial for reliable operation. This thesis addresses three fundamental pillars that form the cornerstone of reliable embodied AI systems: observation generalization, action robustness, and learning efficiency.

Observation Generalization: Towards Sensor-Level Perception. Current locomotion controllers are fundamentally constrained by their architectures, which bind them to specific sensor configurations and input modalities. This limitation becomes particularly problematic when sensors fail during operation or when new sensor modalities need to be integrated into existing systems to enhance capabilities or address evolving task requirements.

The inability to handle dynamic sensor configurations represents a critical barrier to the practical deployment of robotic agents in real-world scenarios. Current approaches require complete retraining when sensor configurations change, which is impractical for deployed systems that must maintain operational capability with minimal downtime and flexibility to payload modifications. The traditional paradigm of training separate models for different sensor configurations leads to inefficient resource utilization, increased maintenance complexity, and reduced system robustness in the face of sensor failures or hardware modifications.

The emergence of transformer architectures presents a compelling opportunity to address these sensor-level generalization challenges through their flexibility in processing variable-length input sequences and their attention mechanisms for capturing complex inter-modal dependencies. However, realizing this potential requires novel approaches to sensor tokenization, training methodologies, and architectural design that can effectively leverage these capabilities while satisfying the stringent latency and reliability requirements of real-time robotic control systems.

Action Robustness: Resilience Under Hardware Degradation. Action robustness addresses the critical need for physical AI agents to maintain functional performance despite hardware failures and environmental uncertainties. Real-world environments present numerous sources of uncertainty and unpredictable hardware

malfunctions, including actuator degradation, sensor failures, and mechanical wear that can compromise system performance.

The growing imbalance between rapid advances in high-level AI capabilities and the relative neglect of low-level system robustness creates a critical vulnerability in physical AI systems. Although foundation models have enabled unprecedented progress in reasoning, planning, and human-robot interaction, these advanced capabilities often rest on fragile low-level control systems for locomotion that are unable to handle hardware failures and environmental uncertainties in real-world deployment. This discrepancy between cognitive sophistication and operational reliability represents a fundamental bottleneck in developing truly dependable embodied AI systems.

The development of fault-tolerant locomotion agents represents a crucial stepping stone toward creating robust autonomous systems that can be confidently deployed in remote and challenging terrains. Quadrupedal platforms serve as an ideal testbed for investigating robustness challenges due to their high degree-of-freedom (DoF) control complexity and their intended operation in unstructured outdoor environments, which expose them to various environmental challenges and hardware stress conditions. Understanding how physical AI agents can be trained to maintain functionality despite critical hardware failures is essential to build trustworthy autonomous systems capable of mission-critical operations.

Learning Efficiency: Beyond Sequential Knowledge Transfer. While the teacher-student framework has demonstrated promising utilization of privileged information during training, it suffers from limitations that constrain its effectiveness due to its multi-stage approach to knowledge distillation. Teachers trained with access to privileged information often converge to near-optimal policies with fewer training iterations. However, given the limited observational information available to students, supervised-based knowledge transfer struggles to cover comprehensive behavioral scenarios, resulting in suboptimal student policies that remain vulnerable to distribution shift during deployment.

The sequential nature of traditional knowledge distillation pipelines creates computational inefficiencies that limit their scalability to complex models and tasks. The rigid separation between teacher training and student learning phases prevents mutual adaptation and co-evolution, resulting in knowledge transfer processes that

are suboptimal and fail to leverage the full representational potential of both models. Furthermore, as model and task complexity increase, the computational cost and temporal requirements of these multi-stage approaches become expensive, creating significant barriers to practical deployment.

There exists a critical need for unified frameworks that can more effectively bridge the gap between privileged training information and deployment constraints, enabling superior utilization of available knowledge through a balanced integration of teacher guidance and student self-exploration while ensuring robustness to information limitations during real-world deployment. A unified training approach that seamlessly integrates policy optimization and knowledge transfer within a single computational process could potentially overcome these fundamental limitations, substantially reducing computational requirements while improving the effectiveness of knowledge transfer for complex embodied AI systems operating in dynamic environments.

1.3 Main Work

This thesis presents three complementary approaches that systematically address critical challenges in learning-based methods for quadrupedal locomotion, specifically targeting robustness under hardware failures, generalization across sensor configurations, and efficient knowledge transfer for sim-to-real deployment. Each contribution leverages advanced simulation strategies and transformer-based architectures to advance the state-of-the-art in physical AI systems through unified learning frameworks.

Masked Sensory-Temporal Attention for Sensor Generalization. We propose a Masked Sensory-Temporal Attention (`MSTA`) framework to address the challenge of deploying control policies across robots with varying proprioceptive sensor configurations. This transformer-based approach introduces direct sensor-level tokenization that treats each proprioceptive measurement as an independent token, enabling unified processing of diverse sensor types across temporal windows. The framework employs a straightforward random masking technique during training that forces robust state representation learning even with significant portions of sensory input missing or corrupted.

`MSTA`'s flexible architecture handles variable sensor configurations without retraining, enabling zero-shot deployment on robots with different proprioceptive sensor sets. Experiments demonstrate stable locomotion maintenance even with most of the sensory information masked out, establishing a foundation for universal quadruped policies that can adapt to diverse robotic platforms and sensor degradation scenarios.

Fault-Tolerant Quadruped Locomotion with Reinforcement Learning. We propose a Fault-Tolerant Locomotion (`FTL`) framework, one of the first comprehensive RL-based methodology for fault-tolerant quadruped locomotion, to address the inevitable reality of hardware failures in real-world deployments. The approach pioneers a novel joint failure simulation framework that accurately models both soft-lock and hard-lock joint failures, creating realistic training scenarios that match physical world conditions. Our joint optimization approach within the teacher-student RL paradigm enables zero-shot transfer of fault-tolerant policies to physical robots without fine-tuning. The general fault-tolerance strategy integrates seamlessly with existing RL-based controllers without requiring specialized reward design or pre-defined gait patterns.

Extensive validation on physical Unitree A1 robots demonstrates the controller's ability to maintain locomotion stability during joint failures, significantly outperforming baseline approaches. Notably, the system exhibits remarkable generalization beyond training scenarios, handling multiple simultaneous joint failures despite training only on single-joint failure cases. This work establishes the foundation for robust RL-based control that can operate reliably despite hardware failures.

Unified Training Pipeline for Knowledge Transfer. We propose a Unified Locomotion Transformer (`ULT`) framework to revolutionize sim-to-real knowledge transfer by integrating policy optimization and knowledge transfer in a single-phase training process. This approach addresses the computational inefficiency of traditional multi-stage teacher-student frameworks through a transformer-based architecture that simultaneously generates both teacher and student actions from a unified model. The innovative action mixer mechanism dynamically balances exploration between teacher and student policies during training, enhancing both policies while facilitating effective knowledge transfer.

ULT's joint optimization combines reinforcement learning, next state-action prediction, and action imitation in a single training stage, eliminating separate supervised learning phases and significantly reducing computational requirements. The framework achieves superior performance compared to sequential teacher-student approaches while dramatically simplifying the training pipeline, enabling direct sim-to-real deployment with zero-shot transfer across various terrains and movement directions.

1.4 Contribution of the Thesis

This thesis makes three primary contributions that advance learning-based controllers for quadrupedal robots through integrated simulation strategies and unified frameworks:

Sensor-Level Generalized Policy. A novel transformer-based framework enabling single policies to adapt across diverse sensor configurations through sensor-level tokenization and masking strategies. The approach addresses fundamental limitations in current locomotion controllers by enabling cross-sensor generalized deployment with a single policy, maintaining stable performance even with substantial sensor masking.

RL-Based Fault-Tolerant Control. The first comprehensive RL approach for hardware fault tolerance in quadrupedal locomotion, featuring novel joint failure simulation strategies that enable zero-shot transfer to physical systems. The methodology is generalized beyond training scenarios, handling multiple joint failures through innovative teacher-student optimization that integrates seamlessly with existing RL-based locomotion controllers.

Unified Knowledge Transfer Framework. A novel single-phase training architecture that eliminates traditional multi-stage teacher-student processes through unified transformer design and action mixing mechanisms. The framework significantly reduces computational complexity while maintaining superior performance, streamlining development pipelines for transformer-based robotic controllers.

These contributions collectively establish a comprehensive framework for robust, adaptable, and efficiently deployable quadrupedal robots that can operate reliably

in dynamic real-world environments despite hardware failures, sensor variations, and deployment constraints. By addressing the critical gap between high-level capabilities and low-level reliability, this work lays the foundation for physical AI systems that can successfully bridge the sim-to-real divide and maintain operational integrity in the face of real-world challenges.

1.5 List of Materials Related to the Thesis

The thesis mainly contains the materials from the following papers:

- **Dikai Liu**, Tianwei Zhang, Jianxiong Yin, and Simon See. Masked Sensory-Temporal Attention for Sensor Generalization in Quadruped Locomotion. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2025.
- **Dikai Liu**, Jianxiong Yin, and Simon See. Towards Fault-tolerant Quadruped Locomotion with Reinforcement Learning. In *IEEE Conference on Artificial Intelligence (CAI)*, 2024.
- **Dikai Liu**, Tianwei Zhang, Jianxiong Yin, and Simon See, Saving the Limping: Fault-Tolerant Quadruped Locomotion via Reinforcement Learning. *arXiv preprint arXiv:2210.00474.* (extended technical report of the above conference paper)
- **Dikai Liu**, Tianwei Zhang, Jianxiong Yin, and Simon See. Unified Locomotion Transformer with Simultaneous Sim-to-Real Transfer for Quadrupeds. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2025.

1.6 Outline of the Thesis

The remainder of this thesis is structured as follows:

- Chapter 2 presents a comprehensive literature review on various tools and techniques for learning-based robotic controller, such as essential knowledge for

robotic RL and simulation development. In particular, extensive approaches to closing the sim-to-real gap and achieving a generalized RL are reviewed. In addition, a brief introduction to transformer-based controllers, especially VLAs, is also provided.

- Chapter 3 focuses on sensor-level generalization for locomotion agent, introducing `MSTA`, a novel framework to enhance sensory-temporal understanding and improve the capabilities of a single agent to handle missing information.
- Chapter 4 focuses on handling real-world challenges for quadruped locomotion, introducing `FTL`, a pioneering comprehensive approach to train a fault-tolerant agent that can handle joint failure.
- Chapter 5 focuses on improving the utilization of privilege knowledge when training locomotion agents, introducing `ULT`, a unified framework to transfer teacher-student learning into a single phase to simplify the pipeline.
- Chapter 6 concludes the thesis, summarizes the findings, discusses the implications of all chapters, and offers insights for future research directions.

Chapter 2

Preliminary and Related Works

Robotics has been widely applied in industry and real-world applications to assist humans in various tasks, such as search & rescue [59, 60], patrol [61, 62], and delivery [63, 64]. Although developing continuous control algorithms in real-world environments remains challenging due to data acquisition costs and domain knowledge requirements, reinforcement learning (RL) has become a promising approach for training robots to perform complex tasks across diverse environments. This paradigm shift has been accelerated by advances in deep learning, simulation technologies [2, 21, 65] and robotic hardware [66–72]. The landscape of robotic control has undergone a fundamental transformation with the adoption of embodied artificial intelligence and Vision-Language-Action (VLA) models [11, 36, 37, 43, 54–57], which represents a significant departure from classical automation, where robots follow pre-programmed sequences, toward systems capable of generalized reasoning and adaptation in dynamic environments.

This chapter systematically reviews foundational concepts and recent advances in RL-based robotic control, focusing on key challenges, methodological innovations, and emerging trends that shape the current landscape of robotic learning.

2.1 Robot Learning for Real World Applications

2.1.1 Partially Observability in Robotics

Real-world robotic systems typically operate under partial observability due to multiple limitations that distinguish them from idealized simulated environments [73]. Unlike simulation environments where complete state information is readily accessible, physical robots face fundamental constraints that prevent direct access to the underlying state of the system [74].

Additionally, physical sensors introduce measurement uncertainty and noise that corrupt state observations. In robotics, observations from sensor measurements, such as motor encoders for joint states; inertial measurement unit (IMU) for body orientation; ultra-wideband (UWB) and Global Positioning System (GPS) for indoor and outdoor localization; and cameras for visual and depth information, can be derived from ground truth, as these sensor readings often contain noise due to interference from other onboard electronics and rapid motion [38]. These sensor uncertainty and noise propagate through the robot's perception pipeline, affecting environmental understanding and ultimately decision-making capabilities.

The unpredictable and dynamic real-world environment also introduces an increased difficulty in observability [74], unlike the simulation environment or even controlled laboratory settings. The hardware and computational constraints also mean that physical robots are operating under strict computational, power, and bandwidth limitations that require balance of achieving real-time process and maintaining accuracy for multi-sensor systems.

Since the underlying privileged ground truth states like root velocity, surrounding height maps, and friction coefficients are often expensive or impossible to retrieve in the real world, robotic systems often rely on temporal sequences of observations to infer hidden state information for effective control. Historical observations enable robots to maintain implicit state representations through temporal dependencies that capture system dynamics over time [75].

Classical control theory approaches have long recognized the importance of incorporating historical information to address partial observability in robotic systems. Model Predictive Control (MPC) frameworks utilize sliding window approaches

to process historical observation sequences, enabling robust state estimation and control synthesis under partial observability constraints [76]. Similarly, Optimal Control for Switched Systems (OCS2) employ receding horizon strategies that maintain temporal memory of past states and actions to optimize future control decisions [77].

Modern robotic systems leverage neural network architectures to process sequential observation data and reconstruct latent states from partial sensor information. Temporal Convolutional Networks (TCNs) [78] have become a powerful paradigm for handling time-series data in robotic applications, demonstrating superior performance in tasks such as action segmentation [79] and quadruped locomotion [8]. Recurrent Neural Networks (RNNs) [80], particularly Long Short-Term Memory (LSTM) networks [81], remain fundamental for sequential learning tasks due to their ability to maintain temporal memory and handle variable-length sequences and is applied in tasks such as 3D scene understanding [82]. Transformer architectures [42] represent the latest frontier in robotic sequential learning, bringing unprecedented capabilities, and become the foundation for VLA models [43].

For reinforcement learning applications, leveraging privileged information during training can significantly improve performance and yield better policies [83, 84]. By learning privileged knowledge representations from historical trajectories with underlying changeable dynamic information, the sim-to-real gap can be bridged through temporal information processing [85]. Although simulation environments provide access to complete state data, deployment policies must operate using only available sensor observations, making temporal state reconstruction capabilities essential for successful real-world performance.

Recent advances in sensor fusion and state estimation have shown promising results in addressing the partial observability challenge with deep learning approaches. Furthermore, the integration of multiple sensory modalities through learned representations has shown potential for robust state reconstruction under varying environmental conditions.

2.1.2 Partially Observable Markov Decision Process with Reinforcement Learning

The foundation of reinforcement learning lies in the Markov Decision Process (MDP) framework [86], which provides a mathematical formulation for sequential decision making under uncertainty. An MDP is formally defined as a 5-tuple:

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma) \tag{2.1}$$

where $\mathcal{S}$ represents the state space, $\mathcal{A}$ denotes the action space, $\mathcal{T}(s'|s,a)$ is the state transition probability from state s to state s' after taking action $a \in \mathcal{A}$, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function for each state-action pair, and $\gamma \in [0, 1)$ is the discount factor.

The MDP framework fundamentally assumes that states satisfy the Markov property, which asserts that the probability of transitioning to a future state depends solely on the current state and action taken, independent of the entire history of previous states and actions. Mathematically, this memoryless property of MDP can be expressed as:

$$\mathbb{P}(s_{t+1}|s_t, a_t) = \mathbb{P}(s_{t+1}|s_t, a_t, s_{t-1}, a_{t-1}, \ldots) \tag{2.2}$$

which significantly simplifies the computational complexity of decision making by eliminating the need to consider the complete trajectory history. In robotic applications, the agent is guided by a policy $\pi : \mathcal{S} \to \mathcal{A}$ that maps states to actions, allowing the robot to make optimal decisions based on current sensory observations. The optimal policy π^* maximizes the expected cumulative discounted reward, formally expressed through the Bellman optimality equation:

$$V^*(s) = \max_{a \in \mathcal{A}} \left[R(s,a) + \gamma \sum_{s' \in \mathcal{S}} P(s,a,s') V^*(s') \right] \tag{2.3}$$

However, most robotic applications operate under partial observability, necessitating the Partially Observable Markov Decision Process (POMDP) framework. POMDP

extends MDP by introducing an observation space Ω and an observation function $\mathcal{O} : \mathcal{S} \times \mathcal{A} \to \Omega$, creating the 7-tuple formulation:

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega, \mathcal{O}, \gamma) \tag{2.4}$$

where $\mathcal{O}(o|s', a)$ represents the probability of observing $o \in \Omega$ when transitioning to state s' after taking action a.

Proximal Policy Optimization (PPO) [87] has become the dominant policy gradient algorithm for RL-based robotic applications due to its balance in simplicity, stability, and performance. Compared to traditional policy gradient methods, which often suffer from high variance and instability, or more complex algorithms such as Trust Region Policy Optimization (TRPO) [87], the clipped surrogate objective in PPO prevents destructive policy updates while maintaining computational efficiency, while searching for the optimal policy $\pi^*(s, a)$:

$$\pi^*(s, a) := \arg\max_{\pi} \mathbb{E}_{s_{t+1} \sim T(\cdot|s_t, a_t)} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right] \tag{2.5}$$

The PPO algorithm operates by collecting trajectories $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \ldots)$ using the current policy π_{θ_k}, then optimizing a surrogate objective function. The probability ratio between new and old policies is defined as:

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)} \tag{2.6}$$

The clipped surrogate objective prevents large policy updates through:

$$\mathcal{L}^{CLIP}(\theta) = \mathbb{E}_t \left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon)\hat{A}_t) \right] \tag{2.7}$$

where $\hat{A}_t$ represents the advantage estimate and ϵ (typically 0.2) defines the clipping range. The advantage function measures how much better an action is compared to the average:

$$\hat{A}_t = \hat{R}_t - V_\phi(s_t) \tag{2.8}$$

where $\hat{R}_t = \sum_{l=0}^{T-t} \gamma^l r_{t+l}$ is the discounted return and $V_\phi(s_t)$ is the value function approximation.

PPO is effective in continuous control tasks, which dominate robotic applications, with the capability to handle high-dimensional action spaces and produce smooth, continuous control signals, making it particularly suitable for robotic manipulation [88] and locomotion [3, 4].

The complete PPO objective incorporates value function learning and entropy regularization:

$$\mathcal{L}(\theta, \phi) = \mathcal{L}^{CLIP}(\theta) - c_1 \mathcal{L}^{VF}(\phi) + c_2 S[\pi_\theta](s_t) \tag{2.9}$$

where $\mathcal{L}^{VF}(\phi) = \mathbb{E}_t[(V_\phi(s_t) - \hat{R}_t)^2]$ is the value function loss and $S[\pi_\theta](s_t) = -\sum_a \pi_\theta(a|s_t) \log \pi_\theta(a|s_t)$ represents the entropy bonus for exploration. The coefficients c_1 and c_2 control the relative importance of value function fitting and exploration, respectively.

2.1.3 Robot Learning with Simulation

Simulation represents the cornerstone of modern reinforcement learning in robotics, enabling scalable policy training while avoiding the high cost and safety constraints in physical experimentation [89]. The high-fidelity simulation for robotic RL provides controlled environments where millions of hours of experience can be accumulated within short wall time, which would be impossible to achieve in the real world. Without robust simulation frameworks, the sample complexity requirements of deep RL algorithms would make it infeasible [24].

Robotic simulators continue to evolve rapidly to meet the increasing demands for enhanced simulation accuracy, performance, and scalability in reinforcement learning contexts. GPU-accelerated physics engines like Isaac Gym deliver extraordinary performance improvements compared to legacy simulators with photorealistic rendering, physical-accurate modeling, and end-to-end acceleration for RL researchers [2]. For robotic RL training in virtual environments, computational scalability through massively parallel simulation represents a critical enabler for practical algorithm development. Modern simulation frameworks support parallel

training across thousands of independent simulation instances, allowing training iterations to be executed at high frequency within constrained timeframes [24].

The fragmentation of simulation environments has increased the difficulty in reproducibility and generalization across different platforms. A comprehensive survey by Collins et al. [89] systematically analyzes common simulators reveals a significant difference in their capabilities for robot learning. Many simulators like Gazebo [90], PyBullet [91], and MuJoCo [21] lack realistic rendering, while other platforms like RaiSim [92] excel in contact-rich scenarios with realistic rendering but sacrifice multi-physics flexibility. Isaac Sim achieves the optimal balance of comprehensive sensor modalities with photorealistic visual output and accurate physics, but is very computationally intensive [28]. To address the fundamental challenge of cross-simulation, RoboVerse [93] introduces a unified simulation framework that supports multiple simulators and robotic embodiments with unified APIs to launch environments, load assets and interact with physics engines. The challenge of transferring knowledge across different simulation environments has gained increasing attention with sim-to-sim transfer serving as an essential preliminary validation for eventual sim-to-real deployment by leveraging diverse simulation capabilities.

2.1.4 Closing Sim-to-Real Gap

Although utilizing simulators can significantly relax the requirement of domain knowledge, and accelerate the development process, a major challenge in this approach is the sim-to-real gap, which refers to the difficulty of transferring skills learned in simulation to the real world due to the persistent difference between two environments.

To bridge the reality gap, it is essential to improve generalization in reinforcement learning. A critical milestone in this pursuit is zero-shot transfer, which denotes an RL agent's capability to apply acquired skills and knowledge to novel, unseen scenarios without any fine-tuning, such as deploying to the physical world. Unlike generalization in well-established domains such as Computer Vision (CV) and Natural Language Processing (NLP), which benefit from shared foundational architectures, generalization in RL remains comparatively underexplored and underdeveloped [38]. Although recent work by Reed et al. [53] introduced a generalized

agent, it functions as a wrapper around a pre-trained RL agent rather than being trained end-to-end with RL.

Generalization in RL is typically examined across three environmental contexts: identical environments, independent and identically distributed (IID) environments, and out-of-distribution (OOD) environments. For identical environments, training and testing conditions are identical (e.g., classical Atari benchmarks). In IID, test environments are drawn from the same distribution as training data (e.g., procedurally generated levels in OpenAI's Procgen Benchmark [94]). Robotic applications frequently encounter OOD challenges, where testing conditions exhibit significant distributional shifts from training data, due to the sim-to-real gap [38]

Domain Randomization. As the most commonly used technique for closing the sim-to-real gap, Domain Randomization (DR) randomizes key environment factors (e.g. friction, motor strength, sensor noise) and even multiple types of rough terrains [3], enabling agents to experience diverse situations during training and improve generalization to noise, environment, and unseen or novel scenarios. DR can be extended to almost every parameter in the simulation, including environmental texture [95] and object status [96].

Since DR is applied at the simulation level, any robotic application can take advantage of it, as demonstrated in racing drones [95], manipulation tasks [96], and quadruped locomotion [7]. Sensor data, hardware specifications, and environmental factors are the parameters randomized most frequently [3, 7, 8]. With the recent focus on multi-terrain deployment, randomly generated terrains with curriculum strategies [3, 8, 24] are frequently applied to make policies robust across different outdoor environments.

However, there exists a trade-off between policy performance and training speed, as agents can become overly conservative, and training requires more sample steps with increased DR parameters and ranges [97]. The complexity of the sample increases exponentially with the number of randomization parameters, making the uniform sampling of the environment parameters suboptimal [98]. Recent research has explored more efficient approaches such as Active Domain Randomization (ADR) [99] and Automatic Domain Randomization [98] that address these limitations by learning parameter sampling strategies or automatically expanding

randomization ranges over time. Real-world information can be further introduced to tune randomization parameters [29] to reduce unnecessary complexity.

Knowledge Distillation. In order to utilize the privileged information available in simulators but inaccessible in the real world, knowledge distillation techniques—particularly the teacher-student framework—have been developed. This approach allows a student model to learn from a teacher model that has access to complete state information, effectively transferring knowledge while taking into account the real-world observation constraints [100].

In the teacher-student framework, the teacher model leverages privileged information including all relevant state information $\mathcal{S}$ to guide the student model's inference process based on limited observations $\mathcal{O}$ [3, 8]. The privileged information can include ground truth states, such as noise-free robot states, surrounding terrain information [8, 101, 102], and domain parameters [3, 4]. The student model learns to mimic the teacher model using perceptible but noisy sensor input, such as IMU readings and joint encoders [3, 4]. For high-dimensional input, point cloud data and camera imagery can also be used for imitation learning [101, 102].

In robotic knowledge distillation, Data Aggregation (DAgger) [58] is commonly used as a core mechanism to train the student policy in a supervised manner. DAgger addresses the distribution mismatch problem inherent in standard behavior cloning by iteratively collecting new trajectories and using optimal actions from the pre-trained teacher to guide the randomly initialized student policy [58]. This approach can operate in both offline and online modes, or combine both strategies [3, 8, 9]. In the offline setting, DAgger aggregates pre-collected expert demonstrations with student-generated trajectories for supervised training [8], which is efficient but contains only good trajectories. The online variant actively queries the expert with the rollout of the student policy, to guide the exploration of the student with a more comprehensive training dataset [3]. With the increased complexity from transformers, two-stage transfer by combining offline and online is used to better initialize the student policy for diverse and high quality trajectories [9]. Active exploration can also be added with RL loss [20].

The effectiveness of knowledge distillation in robotics has been demonstrated in various applications, from quadruped locomotion to manipulation tasks, showing

consistent improvements in sim-to-real transfer performance while maintaining computational efficiency during deployment [3, 4, 8].

2.2 Learning-based Legged Locomotion

Locomotion represents a fundamental capability in robotics, enabling legged systems to traverse through complex, unstructured environments that remain inaccessible to conventional wheeled or tracked platforms, which is critical for real-world applications across challenging terrains [60]. The inherent complexity of legged systems, characterized by high-dimensional and nonlinear dynamics, challenges classic model-based approaches, which struggle to address in a comprehensive way [27].

2.2.1 Legged Locomotion with Reinforcement Learning

The utility of RL for legged locomotion control has fundamentally shifted the paradigm from hand-crafted, rule-based controllers toward adaptive, data-driven policies capable of learning complex behaviors through environmental interaction. Pioneering work by Tan et al. [7] demonstrated the feasibility of sim-to-real transfer for quadrupedal locomotion, establishing domain randomization and robust policy learning as foundational techniques for bridging the reality gap. Subsequent advances in learning-based agile locomotion by Lee et al. [8] showcased the potential for RL policies to achieve dynamic gaits and rapid maneuvering capabilities over challenging terrains. The development of the Rapid Motor Adaptation (RMA) framework by Kumar et al. [3] marked a significant milestone by introducing the adaptation architecture that enables environmental adaptation through privileged learning and knowledge distillation. Margolis et al. [4] utilized curriculum learning approach to achieve unprecedented agility for quadrupedal robots, demonstrating world-record-level locomotion speed.

With the adoption of the transformer architecture, the Terrain Transformer (TERT) developed by Lai et al. [9] represents a paradigmatic change in locomotion control, demonstrating superior performance in varied terrain conditions. More recently, Radosavovic et al. [20] have extended these principles to humanoid locomotion,

employing causal transformer architectures trained through large-scale model-free reinforcement learning to achieve zero-shot real-world deployment capabilities.

The integration of visual perception has catalyzed a revolutionary advancement in legged locomotion, enabling extreme parkour capabilities. Vision-based end-to-end control architectures have enabled unprecedented athletic maneuvers in complex environments. With an end-to-end neural network, the agent can operate directly based on egocentric depth camera perception to climb over high obstacles and leap over large gaps [103–105]. The parkour capability is extended to humanoid robots for complete body control without any motion priors [106].

These developments collectively illustrate the evolution from reactive, environment-specific controllers toward generalizable, adaptable systems that embody the principles of physical AI for autonomous systems to gain capability of perceiving, reasoning, and interacting effectively across diverse operational contexts without extensive manual parameter tuning or environment-specific programming.

2.2.2 Robustness in Legged Locomotion

In physical AI, the trained agent is often closely dependent on robot specifications, sensor sets, and tasks due to the nature of neural network design and the training process. Thus, a policy is often difficult or even impossible to expand to another robot model with different sensing capabilities for a new task without re-training or even re-designing the entire network. However, there are diverse models of robots, and many are designed to take new payload for customization to enhance their capabilities, retraining for each configuration is not scalable. In addition, robots are often deployed in remote, unstructured areas where hardware degradation or even damage can occur. These introduce urgent requirements to enhance the robustness and generalization of physical AI.

Quadrupedal robots are recognized for their exceptional agility and mobility, thanks to their 12 degree-of-freedom (DoF) control and ability to carry diverse sensors and equipment. However, despite their widespread usage in various applications, the field of fault-tolerant control for quadrupedal robots remains relatively limited, with only a few studies addressing this critical research area [107].

Traditional control theory methods in fault-tolerant control for quadrupedal robots often face challenges with generalization. Most research focuses on addressing specific failures, such as joint locking, using control theory to develop fault-tolerant gaits and inverse kinematics solutions [108–110]. More recently, Cui et al. [34] proposed a whole-body control (WBC) method that optimizes robot posture to handle joint lock failures. However, these methods are often tailored to specific quadrupedal robot platforms and require extensive manual analysis and modeling for different failure scenarios.

In pursuit of designing intelligent fault-tolerant controllers, Koos et al. [111] developed an algorithm for a hexapod robot that enables the discovery of compensatory behaviors in unforeseen situations. However, this approach still relies on the gathering of real-world data under failure conditions and can be time consuming and dangerous, making it potentially unsuitable for modern quadrupedal robots deployed in critical missions.

2.2.3 Generalization in Legged Locomotion

The challenge of transferring policies between different robot configurations has gained significant attention in cross-morphology and cross-embodiment domains as the diversity of robotic platforms continues to grow.

Cross-morphology learning specifically addresses variations in physical structure while maintaining consistent core design. For example, transferring policies between quadrupedal robots with different limb lengths and joint configurations represents a cross-morphology challenge. GenLoco [39] exemplifies successful cross-morphology transfer through systematic morphology randomization during training. By procedurally generating quadrupedal robots with randomized body sizes, leg lengths, and mass distributions, GenLoco policies learn robust locomotion strategies that directly transfer to novel morphologies without platform-specific tuning. The framework demonstrates zero-shot transfer to real-world robots, including A1, Mini Cheetah, and Sirius, with performance often exceeding robot-specific policies.

In contrast, cross-embodiment transfer aims to enable a single policy to work effectively across different robots with varying morphology, sensors, or actuators [40, 41, 53, 112–115]. Recently, with transformers and robotic foundation models, Doshi

et al. [41] proposed CrossFormer, a cross-embodied transformer with action heads for different tasks and embodiments, including quadrupeds, single arm, bimanual manipulation and navigation. CrossFormer achieves state-of-the-art performance across six distinct embodiments without requiring action space alignment.

Recent developments have further advanced cross-morphology and cross-embodiment capabilities. The Unified Robot Morphology Architecture (URMA) [40] employs morphology-agnostic encoders and decoders that abstract robot-specific details into a shared latent space, enabling a single policy to control 16 different embodiments from three different legged robot morphologies with description vectors that uniquely characterize each joint through dynamics and kinematics properties, allowing the same network to adapt to arbitrary joint configurations without requiring predefined morphologies.

While significant progress has been made in morphological adaptation and cross-embodiment learning, sensor-level generalization remains a critical gap in current physical AI research. Most existing approaches assume fixed sensor configurations and modalities, requiring complete system redesign when sensing capabilities change due to mission requirements, hardware failures, or equipment upgrades.

2.3 Robotic Foundation Models

2.3.1 RFM with Transformer Architecture

The transformer architecture is a revolutionary breakthrough in machine learning with self-attention mechanisms that enable models to process sequential data effectively [42]. The mathematical foundation of this architecture lies in the scaled dot-product attention mechanism, which enables models to dynamically weigh the importance of different elements in input sequences, formulated as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.10}$$

where Q, K, and V represent query, key, and value matrices, respectively, and d_k is the dimension of the key vectors [42]. This mechanism allows models to

attend to different positions in input sequences simultaneously, capturing long-range dependencies more effectively than traditional sequence modeling approaches with Recurrent Neural Networks (RNNs) [80], such as Long Short-Term Memory (LSTM) [81] and Gated Recurrent Units (GRU) [80]. With this fundamental advantage, transformers have become the predominant architecture in domains such as Natural Language Processing (NLP) [42, 44, 45] and Computer Vision (CV) [46–48].

The evolution of transformers has been particularly profound in cross-modal processing through large language models (LLM), exemplified by systems such as ChatGPT [45, 52] and DeepSeek [116, 117]. These models demonstrate remarkable capabilities in processing language and vision information within unified architectures through vision-language models (VLMs) [118–121]. This multimodal capability has created the foundation for applications in robotic foundation models (RFM) [36, 122], where the integration of visual perception, natural language understanding, and action generation represents a paradigm shift toward more generalized robotic systems.

The adoption of RFMs represents a fundamental transformation toward general-purpose robotic systems capable of adapting across diverse platforms and tasks, by leveraging transfer learning from large-scale vision and language models and data, and subsequently fine-tuned on robotic datasets to acquire embodied intelligence [55, 123]. The theoretical foundation for this approach rests on the hypothesis that web-scale semantic knowledge, when properly adapted through robotics-specific training, can provide robust generalization capabilities across diverse task scenarios and environmental conditions.

2.3.2 Vision-Language-Action Models

Within the broader RFM framework, vision-language-action (VLA) models have become the predominant paradigm [11, 36, 37, 43, 54–57]. These models are typically implement with action heads attached to the VLM backbones, enabling end-to-end learning from visual observations and language instructions to motor commands. By leveraging behavior cloning with diffusion control policy [11, 37, 124, 125] on large-scale datasets comprising real-world, synthetic, and online data, and employing techniques such as Low-Rank Adaptation (LoRA) [126], VLAs

can be effectively fine-tuned from VLM backbones and further transferred to new embodiments.

RT-1 [43] pioneered this approach by demonstrating how transformer models can be adapted for real-world robotic control at scale, processing camera images, language instructions, and motor commands through compact token representations. The fundamental innovation lies in treating robotic control as a sequence modeling problem analogous to natural language processing, where all modalities are encoded into tokens and robot actions are de-tokenized from the output [127]. Building on this foundational work, RT-2 [54] demonstrated how vision-language models trained on web-scale data can be directly incorporated into end-to-end robotic control pipelines to enhance generalization and enable semantic reasoning capabilities. This approach leverages the rich semantic representations learned from web-scale vision-language data, allowing robots to exhibit behaviors that extend beyond their direct training experiences. The field has subsequently evolved with open-source initiatives including the Open X-Embodiment dataset, RT-X [36], and OpenVLA [55], which have demonstrated the potential for generalist robotic policies that inherit semantic understanding from large-scale real-world data with cross-embodiment compatibility.

2.3.3 Challenges in Real-Time Low-level Control

Although RFMs demonstrate superior performance in robotic control with high-level reasoning, they face a fundamental limitation that due to their large scale, inference operates at relatively low frequencies of only 5-10Hz [43, 54], which is not ideal for real-time control applications, particularly high-agility tasks such as locomotion and dexterous manipulation. To address this constraint, one of the approaches involves using RFMs for high-level planning actions such as gait pattern selection and velocity commands, which are subsequently passed to external controllers for execution [128, 129]. For an end-to-end controller, Physical Intelligence introduced π_0 [37], which uses a novel conditional flow matching architecture for continuous action chunk generation with high precision and multimodal modeling capability, making it particularly suitable for high-frequency dexterous tasks. However, these methods limits the capabilities for direct joint control and timely reaction to unpredicted situations, which is critical to high-agile tasks like locomotion.

The progression from single, unified models to dual-system designs represents another exploration in the architecture of the RFMs [11, 12, 130]. This paradigm addresses the increased tension between high-level reasoning capabilities and real-time control requirements in robotic systems by decoupling these functionalities with inspiration drawn from human cognitive processing. Both Figure AI's Helix [12] and NVIDIA's GR00T N1 [11] feature a multi-billion-parameter VLM (System 2) for semantic understanding and scene interpretation and are coupled with a much smaller transformer system (System 1) for high-frequency motor control. This separation enables real-time responsiveness while maintaining reasoning capabilities.

The key advantage of these dual-system approaches lies in their ability to optimize each component for specific computational requirements. System 2 can leverage large-scale pre-trained models for complex reasoning and semantic understanding, while System 1 can be designed for efficient, precise control execution. This architectural evolution represents a maturation of the field, moving beyond monolithic models toward more complex, cognitively-inspired designs that better balance reasoning capabilities with real-time performance demands.

Chapter 3

Masked Sensory-Temporal Attention for Sensor Generalization in Quadruped Locomotion

With the rising focus on quadrupeds, a generalized policy capable of handling different robot models and sensor inputs becomes highly beneficial. Although several methods have been proposed to address different morphologies, it remains a challenge for learning-based policies to manage various combinations of proprioceptive information. This chapter[1] presents Masked Sensory-Temporal Attention (`MSTA`), a novel transformer-based mechanism with masking for quadruped locomotion. It employs direct sensor-level attention to enhance the sensory-temporal understanding and handle different combinations of sensor data, serving as a foundation for incorporating unseen information. `MSTA` can effectively understand its states even with a large portion of missing information, and is flexible enough to be deployed on physical systems despite the long input sequence. Supplementary video available at https://johnliudk.github.io/msta/.

[1]The content of this chapter is published in [131].

TABLE 3.1: Default sensor set for common commercial quadrupedal robots out of the box

Robot Model	Joint Encoder	IMU	Foot Contact	RGB-D Camera	LiDAR
A1[66]	✓	✓	✓	✓	✗
Mini Cheetah[67]	✓	✓	✗	✗	✗
ANYmal C[68]	✓	✓	✓	✓	✓
Spot[69]	✓	✓	✓	✓	✗
Minitaurs[135]	✓	✓	✗	✗	✗

3.1 Introduction

Benefiting from the rapid advancements of deep reinforcement learning (RL) technology [3, 7, 8, 27], quadrupedal robots have showcased their capability to navigate in diverse complex terrains. With the increasing availability of affordable quadruped robots on the market, such as Unitree A1 [66], MIT Mini Cheetah [67] and ANYbotics ANYmal C [68], which are equipped with different sensor sets out of the box as summarized in Table 3.1, there is a growing interest in developing general-purpose locomotion policies that can fit all types of quadrupedal devices. Unfortunately, existing learning-based locomotion policies are trained for specific models, observation spaces, and tasks, making it challenging to transfer or generalize to other unseen robots or scenarios.

Recently, researchers have developed some generalized policies for quadruped locomotion, such as GenLoco [39], ManyQuadrupeds [132] and BoT [133], which have the ability to adapt to diverse morphologies and degrees of freedom (DoF). However, these methods still depend on a fixed observation space input for generating latent space representations. They become ineffective when facing the following situations: (1) deployment on quadrupeds with a different sensor set; (2) unreliable sensor data due to wear and tear, (3) adapting to a new task with new input. Since sensory feedback is interrelated and each sensor plays a critical role at different stages of the locomotion [134], a policy with a deep understanding of proprioceptive information to handle flexible inputs is desired, to enhance the generalization, flexibility, and extensibility.

FIGURE 3.1: Commonly seen low-level sensors on a quadrupedal robot. However, actual sensor set is still different across models, and sensor degradation can cause part of sensor data to be unreliable or even unavailable. With `MSTA`, we create a generalized model to enhance the understanding of sensor information to handle variable sensor input for quadruped locomotion.

One promising solution is self-attention-based transformers [42], which have demonstrated exceptional capabilities in understanding complex sequential information of arbitrary lengths. Transformers have become the preferred choice in domains such as Natural Language Processing (NLP) [42, 44, 45] and Computer Vision (CV) [46–48]. They can efficiently capture sequential information with arbitrary input length and outperform traditional sequence modeling methods such as Long Short-Term Memory (LSTM) [81] and Gated Recurrent Units (GRU) [80]. They have been widely used in robotics to enhance various tasks with multimodal processes [36, 43, 53, 54]. However, due to the complex model structures and vast parameters, robots driven by transformers often run at very low frequency [43, 54], or depend on external high-power computing platforms [136]. with Vision Language Models (VLM) [43] and Vision Language Action Models (VLA) [36, 54] to achieve embodied intelligence. However, since most of these models are based on multi-million-parameter Large Language Models (LLM) such as GPT-4 [52], they often run at only 5 Hz [43, 54] or depend on an external high-power computing platform [136]. To alleviate these issues, two common strategies are proposed to achieve smooth and stable control for transformer-based locomotion tasks. The first strategy involves outputting high-level commands as an interface, such as gait pattern [128] and velocity commands [129]. An external joint controller, which requires additional design and training, is then used to convert these high-level commands into joint commands. The second strategy, targeting the vanilla locomotion tasks, applies linear projections to observation-action data for tokenization [9, 10, 20, 137].

This method is straightforward and efficient for producing joint commands in an end-to-end manner and can be used to merge multiple specialist policies into a single policy [137]. However, it limits the transformer's direct access to sensory information and still relies on fixed sensor input, thereby constraining its in-context understanding capability and multimodal nature of the data. For locomotion tasks, the observation-action data are commonly encoded at the timestep level [9, 10, 20, 137], which is straightforward and efficient for producing joint commands in an end-to-end manner. However, it limits the transformer's direct access to sensory information and still relies on fixed sensor input as they are hidden behind linear projection, thereby constraining its in-context understanding capability and multimodal nature of the data.

To address the above limitations, we propose Masked Sensory-Temporal Attention (`MSTA`), a novel transformer-based model for end-to-end quadruped locomotion control. It achieves sensor input generalization with its multimodal nature, while still being directly deployable on physical systems. Specifically, in `MSTA`, all sensory data are discretized and tokenized to form a long proprioceptive information sequence. Inspired by the work [138] on learning spatiotemporal information in video understanding, a random mask is applied to remove a portion of the observation during training. This significantly enhances the model's sensory-temporal understanding, to better handle different combinations of sensor data and serve as a foundation for incorporating unseen data. Additionally, it aids in identifying the most essential sensory information, thereby reducing the computational power required for physical deployment.

We conduct extensive experiments in the simulation and physical world. Evaluation results demonstrate that `MSTA` can efficiently handle incomplete sensory information, even with half of the data missing. It is also robust against unseen data, making it a solid foundation for further extensions. With direct sensory-temporal attention, the model is flexible enough to mix-and-match desired information for finetuning, meeting the requirement for different end-to-end quadruped locomotion control in the physical world.

3.2 Preliminary

We adopt the two-stage teacher-student transfer approach from TERT [9] as the basis, which utilizes a well-trained teacher policy through RL with privileged information.

3.2.1 Simulation Environment

We implement the simulation environment based on Isaac Gym and its open-source library IsaacGymEnvs [2] to enable massive parallel training.

Terrain and Curriculum. We adopt the terrain curriculum from [24] with five terrain types (smooth slope, rough slope, stairs up, stairs down, discrete obstacle) and difficulty curriculum. A total of 20 terrains are spawned with proportions of $[0.1, 0.1, 0.35, 0.25, 0.2]$ respectively. Each terrain has 10 levels with increasing difficulty. for larger slope angle, higher stairs step and larger obstacle size. During training, the agent progresses to the next level when the tracked linear reward reaches 80% of the maximum achievable value and regresses if it fails to reach 25%. If any agent finishes the highest level, it will be sent to a random level to continue the exploration. The agent progresses and regresses the level based on the episode cumulative tracked linear reward.

Domain Randomization.

Table 3.2 lists the key parameters of domain randomization used in the simulation to enhance the robustness of the policy following [24, 139]. To simulate user commands, we frequently resample linear commands in longitudinal and lateral directions, as well as the angular heading, which are then converted to angular velocity commands capped at [-1.0, 1.0] rad/s. To enhance the robustness and generalization of our policy, key robot control factors such as stiffness, damping, and motor strength are randomized. External interferences, such as payload variations and push forces, are also randomized. To accommodate the processing time of the transformer, each action can be delayed by up to 15 ms. The same randomization techniques are applied to both the teacher training session and the knowledge transfer session.

TABLE 3.2: Key simulation parameters of domain randomization.

Parameters	Range	Unit
Linear Command	[-1, 1]	m/s
Angular Heading	[-3.14, 3.14]	rad
K_p Scale	[0.9, 1.1]	-
K_d Scale	[0.9, 1.1]	-
Friction Scale	[0.7, 1.3]	-
Motor Strength Scale	[0.9. 1.1]	-
Payload	[0, 5]	kg
Payload CoM Offset	[-0.1, 0.1]	m
External Push	[-1, 1]	m/s
Gravity	[9,41, 10.21]	m/s^2
System Delay	[0, 0.015]	s

TABLE 3.3: Observation Noise Distribution

Obs	Distribution Scale
q	0.01
$\dot{q}$	1.5
ω	0.2
g	0.05
cmd	0.02

To enhance generalizability, in our simulation environment, following [24], the noise was generated from a uniform distribution with the scale specified in Table 3.3 and added to each element of the observation.

Observations and Actions. The privilege observation e_t for teacher training contains ground-truth data gathered from simulation, including base linear and angular velocity, orientation, surrounding height map and randomized parameters as described above. For proprioceptive information, we use three commonly seen low-level sensors from quadrupeds. i.e., joint encoders, IMU and foot contact sensors. These sensors can provide five sensory data, including joint position $q \in \mathbb{R}^{12}$, joint velocity $\dot{q} \in \mathbb{R}^{12}$, angular velocity $\omega \in \mathbb{R}^3$, gravity vector $g \in \mathbb{R}^3$ and binary foot contact $c \in \mathbb{R}^4$. Furthermore, the randomly sampled user command target $cmd = [v_x, v_y, \omega_z]$ and actions from previous step $a_{t-1} \in \mathbb{R}^{12}$ are added, resulting in an observation of $o_t \in \mathbb{R}^{49}$ for each step. To gather the temporal information, a list of historical proprioceptive information $[o_0, o_1, \cdots, o_T]$ from past $T = 15$ steps is stored. Thus, full observation is in the $\mathbb{R}^{49\times15}$ space. Both the teacher and student output the desired joint position a_t, which is further processed by a PD controller

TABLE 3.4: Reward terms for reinforcement learning

Reward	Definition	Scale
Linear Velocity Tracking	$\exp\left[-4(v_{xy}^{cmd} - v_{xy})\right]$	1.0
Angular Velocity Tracking	$\exp\left[-4(\omega_{z}^{cmd} - \omega_{z})\right]$	0.5
Body Z Velocity	$\|v_z\|^2$	-2.0
Body Rotation	$\|\omega_z\|^2$	-0.05
Joint Acceleration	$\|\ddot{\theta}\|^2$	-2.5e-7
Output Work	$\int \|\tau \cdot \dot{\theta}\|$	-2e-5
Power Distribution	$\text{var}(\|\tau \cdot \dot{\theta}\|)$	-1e-5
Action Rate	$(a_t - a_{t-1})^2$	-0.1
Action Smoothness	$(a_t - 2a_{t-1} + a_{t-2})^2$	-0.1
Feet Slip	$\|g_t \cdot v_{xy}^{feet}\|$	-0.04
Collision	$\mathbb{1}_{collision}$	-1

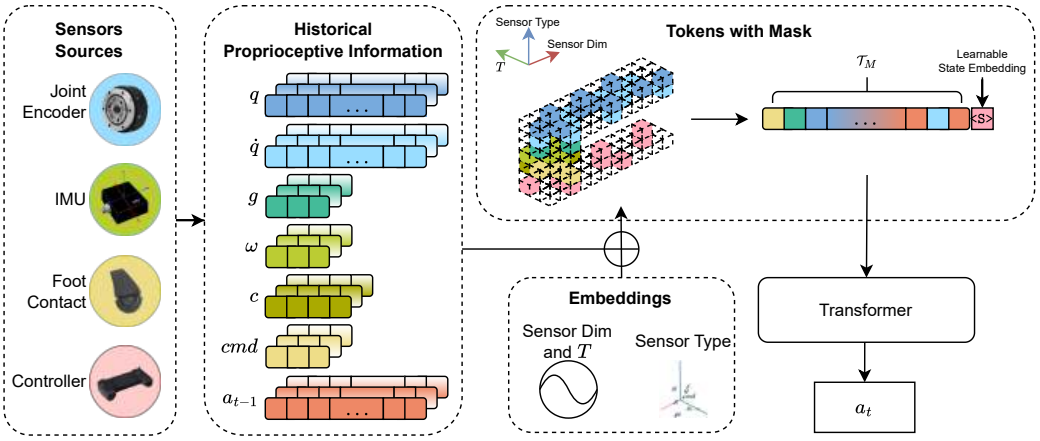


FIGURE 3.2: Overview of our `MSTA`. We gather proprioceptive information from commonly seen low-level sensors for discretization and tokenization. Similar to video understanding, we add additional embedding in three dimensions: sensor type, sensor dim and time. Before being passed to the transformer, a random mask is applied to partially remove the information and a learnable state embedding `<S>` is used to consolidate the information for action prediction. The target joint position output is passed to the PD controller for direct joint control.

for the output torque $\tau = K_p(\hat{q} - q) + K_d(\hat{\dot{q}} - \dot{q})$, with base stiffness and damping set to 30 and 0.7 respectively and the target joint velocity $\hat{\dot{q}}$ set to 0.

Reward Function for RL. The reward functions are designed to encourage the agent to follow the commanded velocity. Following [3, 24, 139], we primarily penalize the linear and angular movement along other axes, large joint acceleration and excessive power consumption. Details of each reward term can be found in Table 3.4

TABLE 3.5: Hyper-Parameters for different training session

Parameters	PPO	Supervised Transfer
Number of GPUs	2	2
Actors per GPU	4096	2048
Episode Length	20s	20s
Horizon Length	24	24
Number of Epochs	5	5
Minibatch Size	16384	3072
Learning Rate	3e-4	5e-4
Scheduler	adaptive	cosine
Optimizer	Adam	AdamW
Clip range	0.2	-
Entropy coefficient	0.001	-
Reward Discount	0.99	-
GAE Discount	0.95	-
Desired KL-divergence	0.008	-
Weight Decay	-	0.1

3.2.2 Teacher Policy and Training

We implement a teacher policy following [3]. The teacher first encodes the privilege information e_t, with a factor encoder μ into a latent space l_t, which is then combined with the latest observation-action pair o_t for the teacher policy $\hat{\pi}$ to output the desired joint position $\hat{a}_t$:

$$\hat{l_t} = \mu(e_t) \tag{3.1}$$

$$\hat{a}_t = \hat{\pi}(\hat{l_t}, o_t) \tag{3.2}$$

The μ and π networks are implemented as MLP with hidden layers of $[512, 256, 128]$ and $[256, 128]$, respectively. The teacher policy is trained with PPO [140] directly to maximize the reward return and is shared across all student transfers at later stages for a fair comparison.

In Table 3.5, present the detailed training hyper-parameters for PPO training of the teacher policy and the knowledge transfer for transformer-based student policies. The two stages in the knowledge transfer process share the same hyper-parameters.

3.3 Methodology

We present `MSTA`, a novel transformer-based model to generate a generalized understanding of low-level proprioceptive information for quadruped locomotion in complex terrains to handle different sensor set equipped on various robot models or when the sensors are damaged and not available. Unlike previous works [9, 10, 20], where each observation-action pair is processed at the timestep level, we treat each sensor modality individually so that the transformer can learn at the lowest level possible. With this foundational understanding, our model is capable of handling different combinations of sensor inputs, enabling better generalization and flexibility. It can potentially be extended to incorporate high-dimensional sensors for more complex tasks. Figure 3.2 shows the overview of `MSTA`.

3.3.1 Tokenization and Embeddings

Sensory-Action Data Tokenize. To learn in-context information at the lowest sensor level, each modality is encoded individually. Instead of the linear projection used in previous transformer-based locomotion controllers [9, 10, 20], where all sensory observations are merged, individual continuous sensor and control data are mapped to tokens directly. Following previous works [43, 53, 54], we pass the normalized data through an encoder to discretizes the value into 256 bins, which are further mapped into a learnable embedding space with $d = 128$ dimensions. Compared to timestep level encoding, in this way, the most information is preserved for in-context understanding by transformer.

Positional Embedding and Sensor Type Embedding. We view the encoded information in a three-dimensional way, sensor type, sensor channel and timestep. To distinguish proprioceptive information from different sources with temporal relations, two additional embeddings are added. The first one is a fixed 2D sin-cos position embedding e_p [141] applied on the channel dimension and time axis of each sensor. For instance, $e_P^{i,t}$ means the embedding added to the i-th channel at timestep t This allows the model to handle sensors with varying lengths of dimensions and historical time windows directly and be easily extendable. To accommodate the multimodal nature of the sensory data, another learnable embedding e_S is add to indicate each sensor type. This enables easy mix and match of information from

different sensors without concerns about the order or placeholders. When new sensors are added, a new sensor embedding can be trained and added in. Thus, for the embedding of i-th channel of a sensor at timestep t, with original encoded token embedding $e_t^{a,i}$, the finial value in the sequence $\mathcal{T}$ is:

$$\mathcal{T}_t^{a,i} = e_t^{a,i} + e_P^{i,t} + e_S^a \tag{3.3}$$

3.3.2 Attention with Missing Information

Random Masking. Inspired by the use of masking in image and video understanding [48, 138] with autoencoders to improve vision understanding, we create a binary mask M based on the target ratio α to randomly mask out portions of the collected sensory, which are directly removed from the original sequence:

$$M_i = \begin{cases} 0, \text{if } m_i < \alpha \\ 1, \text{otherwise} \end{cases} \quad \text{where } m_i \sim \mathcal{U}(0,1), i = 1, \cdots, 49T \tag{3.4}$$

$$\mathcal{T}_M = \{e_i \in \mathcal{T} : M_i = 1\} \tag{3.5}$$

Since only part of the sensory data are visible to the network, the model is required to infer and reconstruct the missing information from them, thereby enhancing its understanding of the relationships between different sensory inputs. Furthermore, random masking significantly reduces the training time and computational resources required. With sensor level tokens, the input sequence length grows from T to $49T$ for observation, and as the complexity of self-attention is necessarily quadratic in the input length [142], the added overhead is enormous, and masking makes it more feasible to run during massive parallel training.

Transformer Model. We implement a vanilla transformer model to process the generated tokens. The model consists of multi-head self-attention blocks with an MLP ratio of 2.0. An additional learnable state embedding `<S>` is added to the end of the masked sequence $\mathcal{T}_M$ to consolidate the processed information [46], which is subsequently projected into the action space with an MLP network π:

$$l_t = \texttt{MSTA}([\mathcal{T}_M, \texttt{<S>}]) \tag{3.6}$$

$$a_t = \pi(l_t) \tag{3.7}$$

Teacher-Student Transfer. Following TERT [9], we train `MSTA` with a two-stage transfer strategy. In the first offline pretraining stage, trajectory is gathered by unrolling the well-trained teacher policy while the student will predict the next actions. This is to ensure that the student can produce reasonable actions during the second online correction stage to overcome the gap of distribution shift by training on its own trajectory. We minimize the loss for action prediction:

$$\mathcal{L} = \|a_t - \hat{a}_t\|^2 \tag{3.8}$$

3.4 Experiments and Results

We design and conduct various simulation experiments to evaluate the effectiveness of the proposed `MSTA`, and its generalization ability for different sensor data. We mainly adopt three metrics: linear velocity tracking return per step, angular velocity tracking return per step, and total final reward return. They indicate how the agent can conduct the task following users' commands and the overall performance. All reported results are averaged over 5000 trails with five terrain types and different levels. They are normalized on the basis of respect teacher data for easy comparison.

3.4.1 Impact of Mask Ratio

First, we investigate the maximum portion of missing data that `MSTA` can handle to reconstruct robot states. During each the transfer stages, we set the masking ratio to 0%, 25%, 50% and 75% independently. Figure 3.3 shows the resultant heatmap matrix. When trained without masking, despite the model having very good performance with all the information available, it suffers from missing data and cannot efficiently reconstruct the status. We can also see that the performance is more dependent on the masking ratio in the second stage than that in the first stage. This is because in the second transfer stage, the student is interacting with

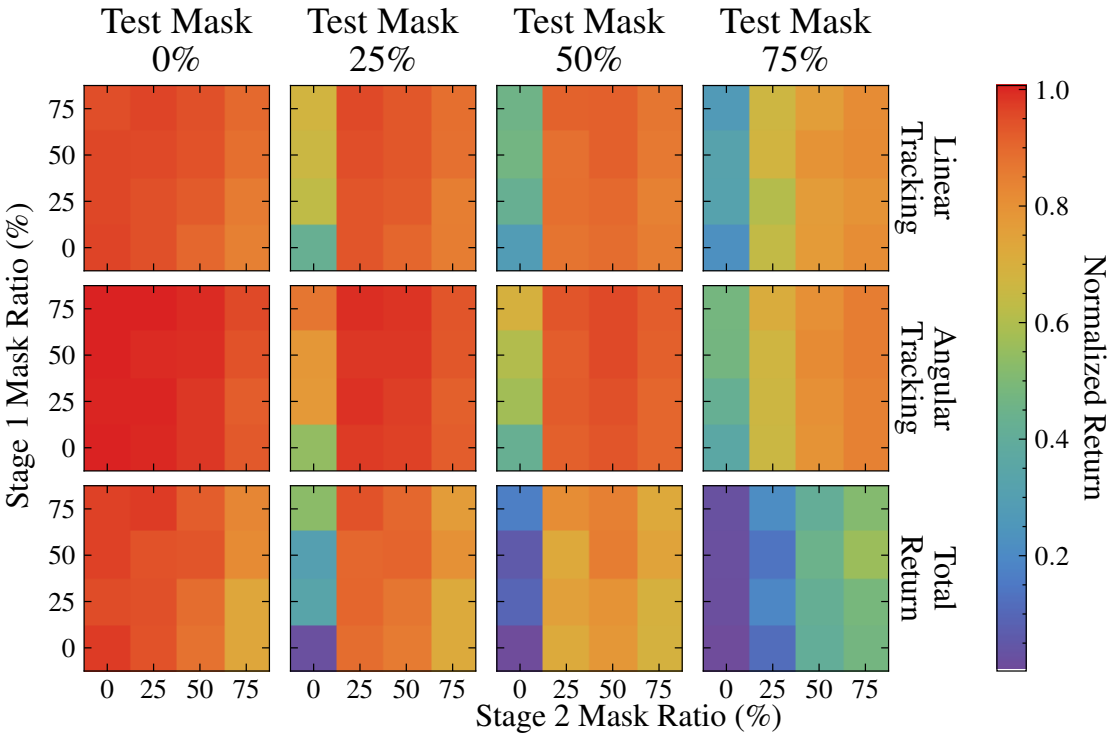


FIGURE 3.3: Heatmap matrix for the performance of models that are trained with different combinations of mask ratios. The three rows from top to bottom represent the linear velocity tracking, angular velocity tracking and total reward return respectively. The four columns denote different masking ratios applied during testing. For each sub-figure, the y-axis is the masking ratio applied during the offline pretraining stage and the x-axis is the masking ratio applied during the online correction stage.

the environment to reduce the gap caused by missing information and observation shift. In contrast, the mission of the first stage is to generate a usable policy that outputs reasonable actions so the agent does not fail dramatically and has the chance in the second stage to generate high-quality trajectories for optimization, which is achievable even with a masking ratio of 75%. This also demonstrates the importance of using two-stage transfer.

Comparing the performance of these models, we choose the one trained with the masking ratio of 75% in the first stage and 50% in the second stage, which can well balance the resource requirement and agent performance.

3.4.2 Comparison with Baselines

We compare `MSTA` with two baselines. The first is RMA [3], which is implemented with TCN [78] to capture temporal information. The second is TERT [9], a transformer-based framework with linear projection for observations and actions in two flavors: concatenated single token and separate tokens for states and action, resulting in T and $2T$ tokens respectively [10]. To evaluate the masking mechanism in our method, we replace the selected observation in `MSTA` with a learnable representation instead of removing them. To further evaluate the importance and capability of the transformer structure, we replace it with a GRU [80] model. We expand the missing information testing to TERT. However, since the observations and actions in TERT are encoded through linear projection before passing to the transformer, it is impossible to directly remove any input. Thus, the the same learnable masks method is applied to TERT.

All variations of `MSTA`, TERT and other baselines are trained with the same two-stage transfer, sharing a common well-trained teacher network, and we apply a testing mask of up to 50%, as identified in Chapter 3.4.1.

Table 3.6 shows the comparison results. When fully optimized with teacher-student transfer, the performance of all fully trained vanilla policies with complete observations is very close, often within just 2% difference. When faced with incomplete information, transformer-based `MSTA` can have a better understanding of the data and reconstruct the robot state more accurately than the GRU-based network, even with only half of the information. When using a learnable representation mask, with `MSTA` or TERT, the agent underperforms to the vanilla removing mask, especially with full observation, showing that a direct removing mask has an advance in both better performance and less resource required. Although we can hack the linear projection in the TERT network to take in missing information, it is not comparable to direct sensor level tokenization and attention for sensory information understanding.

3.4.3 Generalization, Robustness and Flexibility

While achieving state-of-the-art performance, `MSTA` offers additional benefits of generalization and flexibility to customize the model after training or even on the

TABLE 3.6: Comparison results on different terrain types in terms of linear velocity tracking, angular velocity tracking and total reward return for all the variations of trained models.

Terrain	**Metric**	**Ours**			**RMA**	**TERT**		**GRU**			**TERT w/ Mask**			**Ours w/ Learnable**		
		0%	**25%**	**50%**		**Concat**	**Separate**	**0%**	**25%**	**50%**	**0%**	**25%**	**50%**	**0%**	**25%**	**50%**
Smooth Slope	Linear Tracking	1.00	0.99	0.99	1.00	1.01	1.00	1.00	0.96	0.78	0.95	0.98	0.99	0.21	0.93	0.99
	Angular Tracking	1.02	1.02	1.00	1.02	1.02	1.02	1.02	1.01	0.97	0.99	1.00	0.99	0.35	0.97	0.99
	Total Reward	1.01	1.01	0.99	1.02	1.02	1.02	1.01	0.99	0.84	0.95	0.99	0.98	0.25	0.92	0.99
Rough Slope	Linear Tracking	0.97	0.94	0.95	0.99	1.00	0.98	0.96	0.90	0.74	0.91	0.95	0.93	0.21	0.87	0.95
	Angular Tracking	1.01	1.00	0.98	1.02	1.02	1.01	1.01	1.00	0.95	0.96	0.97	0.94	0.33	0.94	0.96
	Total Reward	0.98	0.95	0.95	1.00	1.01	0.99	0.97	0.92	0.79	0.90	0.94	0.89	0.19	0.80	0.91
Stairs Up	Linear Tracking	0.93	0.91	0.91	0.94	0.95	0.95	0.90	0.91	0.69	0.82	0.86	0.85	0.26	0.81	0.85
	Angular Tracking	1.00	0.98	0.97	0.99	1.00	1.01	0.99	0.98	0.94	0.95	0.95	0.92	0.71	0.94	0.93
	Total Reward	0.95	0.91	0.87	0.95	0.99	1.00	0.90	0.87	0.72	0.80	0.81	0.73	0.54	0.74	0.72
Stairs Down	Linear Tracking	0.94	0.93	0.93	0.96	0.97	0.95	0.92	0.93	0.74	0.86	0.91	0.92	0.71	0.94	0.93
	Angular Tracking	1.00	0.99	0.99	1.00	1.01	1.01	0.99	0.98	0.95	0.95	0.96	0.92	0.70	0.93	0.93
	Total Reward	0.94	0.91	0.91	0.95	1.00	0.98	0.89	0.90	0.77	0.83	0.86	0.77	0.49	0.75	0.74
Discrete	Linear Tracking	0.96	0.96	0.94	0.97	0.99	0.98	0.93	0.88	0.75	0.87	0.91	0.89	0.25	0.87	0.85
	Angular Tracking	1.01	1.01	0.99	1.01	1.02	1.02	1.01	0.99	0.95	0.96	0.97	0.94	0.37	0.94	0.94
	Total Reward	1.00	0.97	0.96	0.99	1.04	1.02	0.94	0.89	0.83	0.88	0.91	0.80	0.03	0.80	0.76
Average	Linear Tracking	0.96	0.95	0.94	0.97	0.98	0.97	0.94	0.92	0.74	0.88	0.92	0.92	0.23	0.87	0.91
	Angular Tracking	1.01	1.00	0.99	1.01	1.01	1.01	1.00	0.99	0.95	0.96	0.97	0.94	0.49	0.95	0.95
	Total Reward	0.97	0.95	0.94	0.98	1.01	1.00	0.94	0.92	0.79	0.87	0.90	0.83	0.30	0.80	0.82

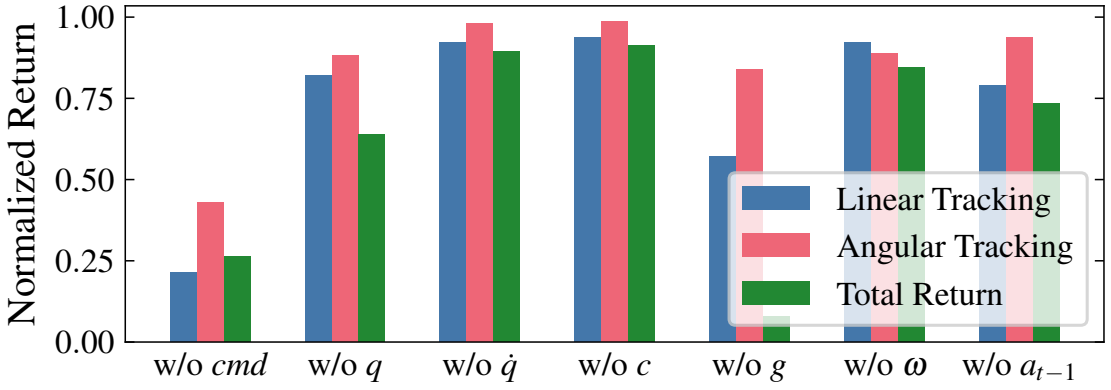
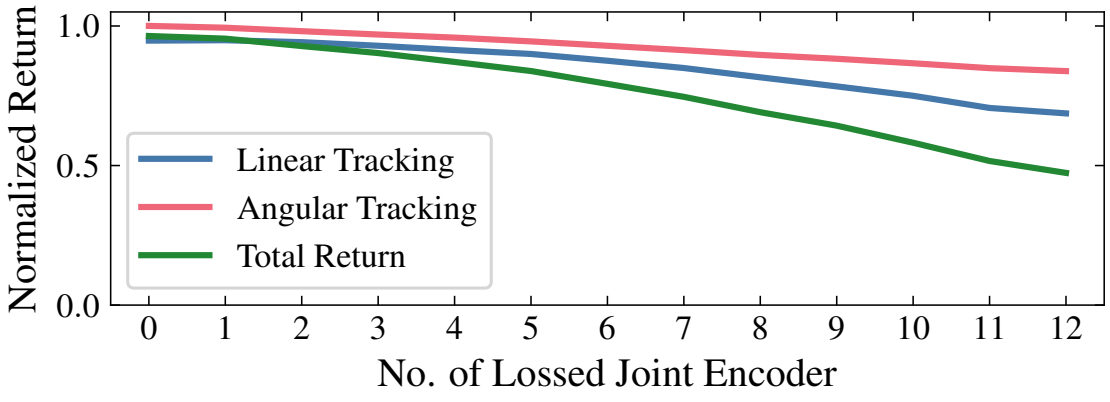


FIGURE 3.4: Performance with certain sensory feedback completely removed.

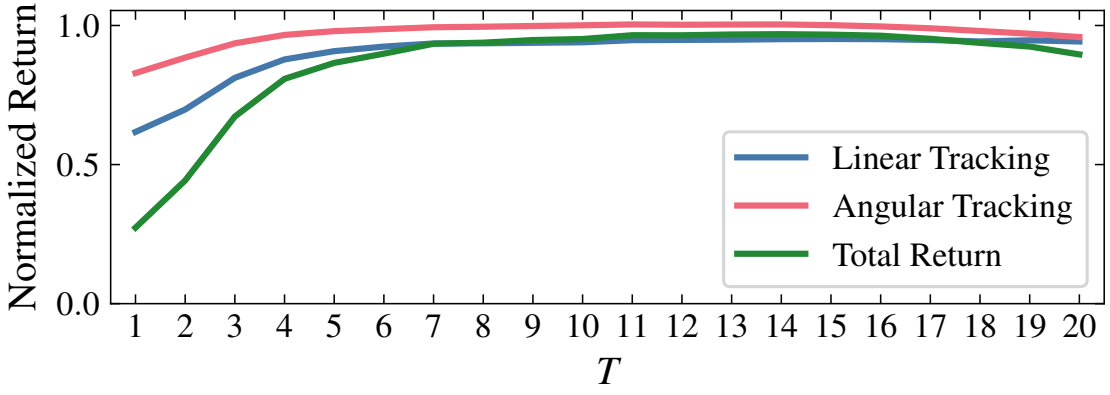
fly to fit the deployment requirement. Quadrupeds are equipped with different sensor sets, and sensor damage can cause certain channels or the entire sensor to be unavailable during deployment, which required the robustness against missing information to handle. Furthermore, we can balance the performance and required computation power by using a shorter sequence based on the insights from in-context sensory information understanding.

Important Sensory Feedback. To understand the importance of each sensory feedback, we further investigate the impact of removing each sensor completely from the observation and the results are shown in Figure 3.4. It is clear that certain feedback like $\dot{q}$, c, ω and even a_{t-1} are quite redundant and a well trained transformer-based `MSTA` can easily compensate the missing information from other sources, while the other sensor data are more critical.

Missing of Sensor Dimension. Some proprioceptive information has multiple channels, such as joint encoders and force senors. This means that these sensors can also be damaged independently due to wear and tear from daily operations and it is not easy to have a redundant sensor. Among these sensors, joint encoders are the source of both q and $\dot{q}$ for the observation. From previous analysis, missing of joint information can be crucial. We investigate the scenario where only a few encoders are dead or the data are compromised and need to be excluded. We conduct the test by masking certain numbers of joint encoders and for each masked joint, the related q and $\dot{q}$ are removed completely from the observation. The results are shown in Figure 3.5A. The loss of the joint encoder can have a great impact on the performance as the related information is very essential for quadruped locomotion.



(A)



(B)

FIGURE 3.5: Performance with various setups: A certain numbers of joint encoders are masked out; B different history time window T is applied.

However, our transformer model can still handle multiple missing encoders before large performance degradation.

Time Window. Another special masking is to completely remove some timesteps, the default window, $T = 15$, is equivalent to past 0.3s. We check whether such a long sequence of information is necessary by applying different time windows without masking. The results are shown in Figure 3.5B. It is clear that `MSTA` can efficiently extract and reconstruct the robot state for actions even with only 7 steps of past information. Interestingly, given a longer timeframe like $T = 20$, which the transformer has never seen during training, `MSTA` is still robust and not affected by such unknown information.

Minimized Observation and Fine-tuning. It is not feasible to infer the transformer with full observation space and long windows with the limited computation power onboard. From the previous analysis, we have identified the important

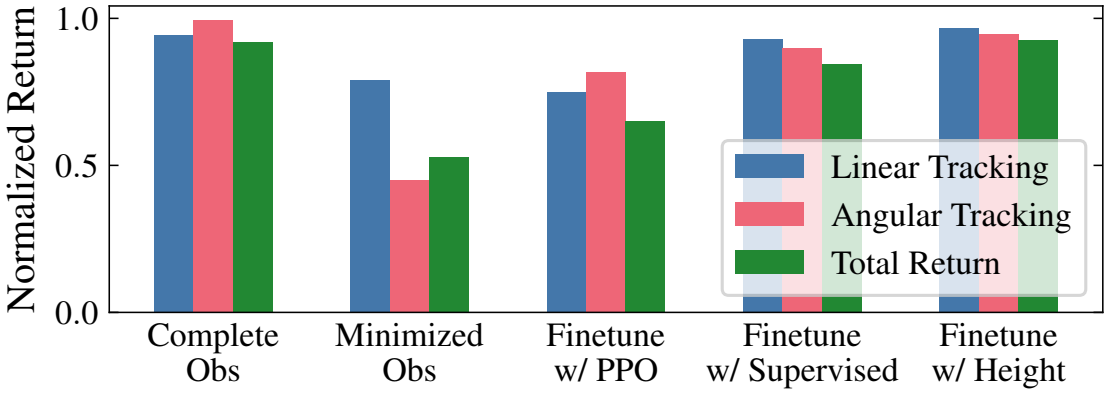
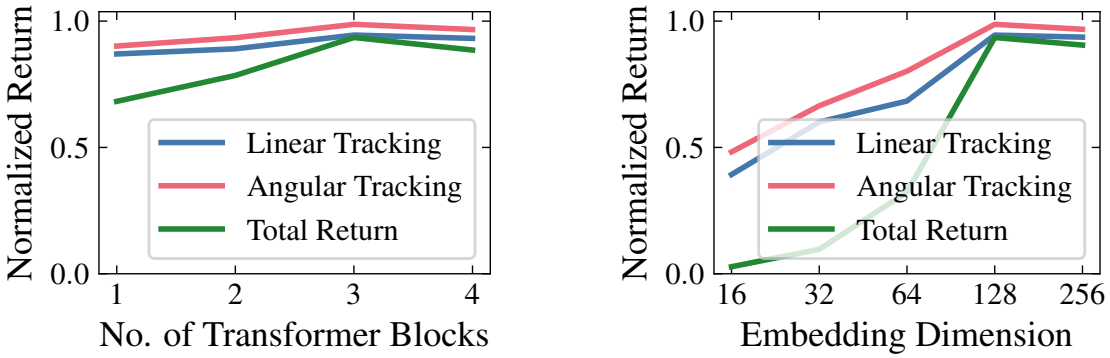


FIGURE 3.6: Performance using minimized observations with finetuning and extension of height map.

sensors and the minimal history length required. We further explore the feasibility of creating a minimized observation policy based on the information. Using an observation with only cmd, q, g and a_{t-1} with a window of $T = 7$, we are essentially removing 71% of the tokens from the complete training observation space.

When directly deployed with such mask, the policy cannot perform well due to all the missing information. To restore the performance, we freeze the transformer for fast fine-tuning of the projection layers and test both the vanilla PPO [140] and supervised learning with online correction [9]. The performance of the policies is shown in Figure 3.6. While both algorithms can help improve the performance of the policy with only minimized observations, supervised learning gives larger boost. Training with the teacher has been identified as one major approach to achieve quadruped locomotion on challenging terrains [3, 9]. Although our foundation with the transformer can provide a solid start point of student policy, additional work is still needed for pure RL-based fine-tuning to reduce the dependence on privilege information.

Extension with New Information. In previous analysis, `MSTA` is robustness against new timestep information. When extending the capability for quadrupeds, additional sensors such as cameras and LiDAR are often needed. We assess the model's capability of handling previously unseen information, which can be appended into $\mathcal{T}$ as new tokens. For instance, we tokenize the height map information using a vanilla MLP encoder and directly extend it to our minimized observation agent for fine-tuning. The performance of the extended agent is shown in Figure 3.6. Height information significantly aids in navigating challenging terrains, such as



(A) (B)

FIGURE 3.7: Performance with various network size with a random mask of 50%: A using different numbers of self-attention blocks tested with; B using different sizes of the embedding dimension.

staircases, and improves the overall locomotion performance even with minimal observations and new encoder needed to be trained. To take the test to an extreme, we added 256 randomly generated dummy tokens, equivalent to a camera frame with ViT [46] before processing, and the agent can still produce explore, which is crucial for two-stage knowledge transfer. This demonstrates that the model can be used as a solid foundation for further extension with high-dimensional information by direct deployment in virtual environments to gather new trajectories. Please refer to the supplementary video for more information.

Model Size. We adjust our model by stacking $[1, 2, 3, 4]$ attention blocks, resulting in a model with 0.2M, 0.33M, 0.46M and 0.59M parameters respectively. Figure 3.7A shows that the model performance increases with the model size, reaching the optimal results with three stacked layers.

We also evaluate the impact of the embedding dimension size, ranging between $[16, 32, 64, 128, 256]$. Figure 3.7B shows that the transformer needs an embedding size of at least 128 for good performance but does not scale beyond that.

3.4.4 Physical Deployment

We successfully deploy the trained policy, exported with JIT, directly on a Unitree A1 robot equipped with a Jetson AGX Orin Developer Kit that serves as both the main processor and a payload. No further model optimization is required for a zero-shot transfer. Although `MSTA` introduces sensor-level tokenization that increases

FIGURE 3.8: Deployment in the physical world on Unitree A1 with minimized observations with zero-shot transfer.

the input sequence length from T to $49T$, the quadratic complexity of self-attention is effectively managed through the masking mechanism and minimized observation strategy. On the Jetson AGX Orin, `MSTA` with the full observation space runs at approximately 75Hz, 100Hz when masking out half of the tokens, and 150Hz with the minimized observation. In comparison, TCN-based controllers [8] and timestep-level transformer tokenization [9, 20] achieve around 300Hz on the same hardware due to their simpler architecture and shorter input sequences. While `MSTA` is slower, 150Hz is sufficient for deployment with a 50Hz control loop, and the system delay randomization during training (Table 3.2) helps the model adapt to such inference latency. For more resource-constrained edge devices, model quantization techniques could further reduce the inference cost while preserving the generalization benefits of sensor-level attention. We notice that the JIT model has slight differences in output compared to the original model, indicating additional work is needed for better portability. Figure 3.8 shows some snapshots from the deployment test. Please refer to the supplementary video for more information.

3.5 Conclusion

This chapter introduces `MSTA`, a transformer-based model for quadruped locomotion. It leverages the masking technique and direct sensor-level attention to enhance the understanding and generation of sensory information input. We evaluate the robustness of `MSTA` with different combinations of proprioceptive information and demonstrate its capability to compensate for missing data and handle unseen information. Finally, we show that `MSTA` is efficient to be deployed on a physical robot without any additional optimization.

Chapter 4

Saving the Limping: Fault-tolerant Quadruped Locomotion via Reinforcement Learning

Modern quadrupeds are skillful in traversing or even sprinting on uneven terrains in a remote uncontrolled environment. However, survival in the wild requires not only maneuverability, but also the ability to handle potential critical hardware failures. How to grant such ability to quadrupeds is rarely investigated. In this chapter[1], we propose a novel methodology Fault-Tolerant Locomotion (FTL) to train and test hardware fault-tolerant controllers for quadruped locomotion, both in the simulation and physical world. The teacher-student reinforcement learning framework is adopted to train the controller with close-to-reality joint-locking failure in the simulation, which can be zero-shot transferred to the physical robot without any fine-tuning. Extensive experiments show that our fault-tolerant controller can efficiently lead a quadruped stably when it faces joint failures during locomotion. Supplementary video available at https://johnliudk.github.io/saving-the-limping/.

4.1 Introduction

Quadrupedal robots are becoming more intelligent in solving various tasks. They demonstrate high flexibility and versatility in complex contexts, and are expected

[1]The content of this chapter is published in [143], with an extended technical report in [144].

to tackle many critical real-world missions, such as search & rescue [60], patrol [61] and delivery [63]. Quadrupeds are normally deployed in remote uncontrolled environments [102], where accidents could happen at any time to cause potential critical hardware failures to the physical device, e.g., joint locking, free swinging, broken brackets. These failures could bring significant harm to the robots and humans, increase the down time, and shorten the service life of the robots. The nature of quadruped instability makes it more susceptible to failures compared to other robotic platforms [145] and fault tolerance is a crucial aspect in the design of quadruped controllers. Therefore, it is important for the onboard controller to be robust against the hardware failures and bring the quadrupeds back home safely.

Unfortunately, existing commercial quadrupeds employ limited hardware failure detection (e.g., motor overheating, sensor signal loss) or protection functions (e.g., shutting down the system), which are not sufficient to operate the system safely and effectively in dynamic and unpredictable outdoor environments. In the research community, previous studies have introduced solutions to achieve various locomotion tasks under *normal conditions*, such as traveling through rough terrains [3, 8, 102], jumping & falling recovery [146], running at high speeds [4] and dexterous manipulation [147]. How to cope with hardware failures at runtime without disruption to the system is still an unsolved problem.

It is challenging to grant the quadrupeds the capability of handling hardware failures automatically. Traditional control theory methods, such as model predictive control (MPC) and whole body control (WBC), require manual tuning of model parameters with in-depth domain knowledge [14, 148]. Using such a predetermined motion and trajectory planner is significantly restricted in reality, especially when facing unknown environmental conditions and failures: even different situations of the same failure type (e.g., joint locking) need separate modeling [34], making them more difficult to generalize and scale to different failure types and hardware configurations.

The study of fault tolerance is expensive in the real world, and the reality gap makes it difficult to deploy a virtual agent to a physical robot. Recent works of RL-based quadruped fault tolerance control have been tested only in simulation and the lack of physical experiment makes it still unclear how such a controller performs in the real world.

The limitation of the simulation engine causing inaccurate physical computation and visual rendering. (2) Real-world environment can be unpredictable and considerably different from the virtual world. (3) Robot model used in simulation is usually simplified and can be significantly different from the real robot condition.

A more promising strategy is to apply reinforcement learning (RL) algorithms to train the policy in a simulator, and then transfer it to the physical world [3, 4, 8]. This can remarkably relax the requirement of domain knowledge. To improve the model performance in the real world, a number of simulators like Isaac Gym [2], have been developed with photo-realistic rendering and physical-accurate modeling. Meanwhile, many techniques are proposed to reduce the sim-to-real gap [25–27]. However, such gap still exists especially in the context of hardware failures for two reasons. First, it is impossible to simulate every possible environment state where hardware failures could occur, even with the domain randomization technique [25]. The consideration of too many environment states can significantly slow down the training process, or even cause convergence failures. Second, the physical robot model used in the simulator is usually simplified, and cannot reflect the real robot conditions (e.g., with hardware faults). Recent works proposed several RL-based controllers to achieve fault-tolerance [149, 150], which are only tested in the simulation environment. Due to the huge sim-to-real gap, it is unclear how these methods will perform in the physical world.

Motivated by these limitations, we design a novel framework to achieve robust fault-tolerance quadruped locomotion in the physical world. We make the following contributions. (1) We design a simple yet efficient way to add, validate, and test fault tolerance against joint locking failure to realistically simulate joint locking failures, and a locking mechanism for real-world testing. The randomized failure in simulation helps to train a generalized agent that can handle various joint locking scenarios in the real world, instead of certain pre-defined cases without dedicate reward design as guidance. (2) We adopt the teacher-student reinforcement learning paradigm to achieve jointly single-phase training and zero-shot transfer. The student model can efficiently extract information from the onboard sensors. When deployed in a physical quadruped, the policy can provide real-time locomotion control against possible hardware failures in uncontrolled environments. Extensive experiments are conducted in both simulation and a physical Unitree AI robot (Figure 4.1).

FIGURE 4.1: Physical robot and its simulated counterpart. Unitree A1 is equipped with our joint locking mechanism. Its official URDF model is used in the Isaac Gym simulator [2] with body links of the locked joint showing in red.

Evaluations show that our method can significantly improve the robustness and hardware fault tolerance of RL-based control policies.

4.2 Methodology

Our goal is to train a control policy π to guide the stable locomotion of the quadruped even when it faces critical hardware failures (e.g., joint locking). This policy takes as input the latent representation $\hat{z}_t$ encoded from the perceivable sensor data and outputs the desired joint position. It is designed to be capable of zero-shot deployment in the real world. Figure 4.2 presents the overview of our methodology.

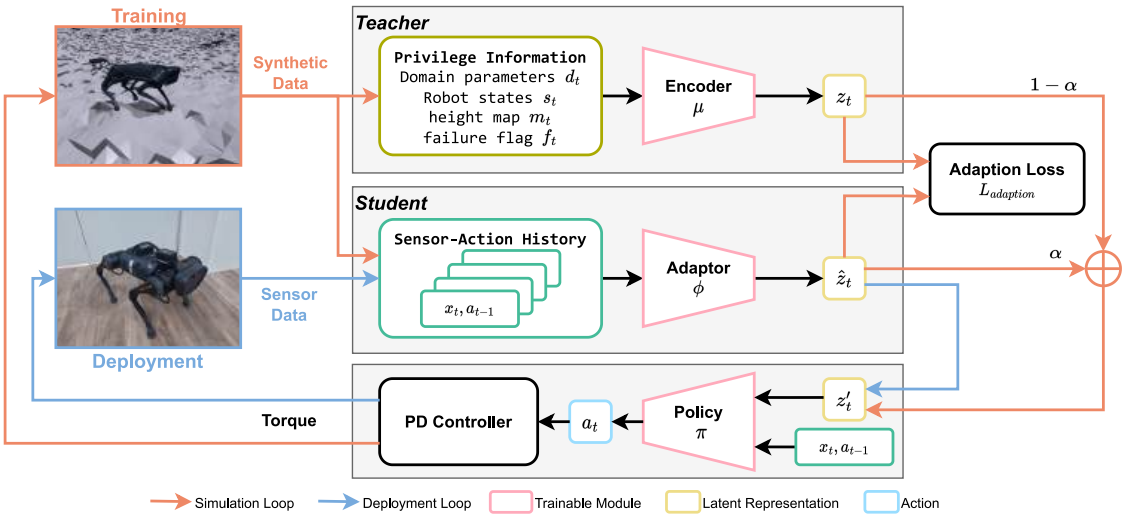


FIGURE 4.2: Methodology overview. We adopt the reinforcement learning architecture with the teacher-student framework from [3, 4] to train the policy. The architecture consists of a teacher network μ, a student network ϕ, and a policy network π. During training, synthetic data from the simulator are used to compute the latent representation z_t and $\hat{z}_t$ of the teacher and the student, respectively. By fusing the latent information, all three networks are trained jointly for fast convergence in the early stage and then an optimized student policy in the end. The policy and student model will be directly deployed on the physical robot without any further offline training or fine-tuning. During deployment, policy network takes only $\hat{z}_t$ from student network as the latent representation.

4.2.1 Joint Locking in Quadruped Locomotion

Quadrupeds deployed in remote uncontrolled environments face the challenges of unpredictable joint failure, which can immobilize or even damage the robot [34]. Joint locking and free-swinging are the most common faulty situations. A locked joint cannot be controlled freely and has only a limited range of motion. However, the actuator can still support the body as torques are still applied. A free swing joint cannot be controlled and no actions are made. It moves easily by an external force, and thus cannot support the body. In this chapter, we mainly focus on the single joint locking failure, which is also the target of recent related works [34, 149, 150].

Joint Failure in Simulation. To safely develop a fault-tolerant controller, Isaac Gym [2] is used to simulate failure situations in the locomotion task with domain randomization. An RMA-like vanilla environment `BaseEnv` is first created, where no failure occurs.

We model joint locking failure by restricting the movement of the joint with a limited range $\theta_{allowed}$, controlled by a central position $\bar{\theta}$ and symmetric tolerance θ_{tol}.

$$\theta_{allowed} = [\bar{\theta} - \theta_t, \bar{\theta} + \theta_t] \tag{4.1}$$

We use a failure flag $f_t \in [0, 12]$ to track the failure status. Then for each virtual agent, we randomly sample the failure time T_f and failure joint J_f, and and the failure tolerance θ_{tol}:

$$T_f \sim \mathcal{U}(T^f_{min}, T^f_{max}) \tag{4.2}$$

$$J_f \sim \mathcal{U}\{1, \ldots, 12\} \tag{4.3}$$

$$\theta_{tol} \sim \mathcal{N}(0, {\theta_{max}}^2) \tag{4.4}$$

Initially and after every reset, f_t is cleared as 0 to indicate a normal state. In the episode, when the agent progresses to T_f, the failure occurs, and f_t is updated to reflect the joint failure $f_t = J_f$. The current position of the selected joint J_t is used as the central failure angle $\bar{\theta} = q_{J_t}$. Joint locking failure is modeled by restricting the joint movement with a limited range $\theta_{allowed}$, controlled by the central position $\bar{\theta}$ and symmetric tolerance θ_{tol}, which are used to directly overwrite the joint's limit with Isaac Gym's API.

We refer to the failure environment as `FailureEnv`. Unlike previous methods [149, 150], where joint failures are predefined and fixed, we use domain randomization to generate versatile and unpredictable situations. Since joint locking directly affects joint control, and the robot status and surroundings at the failure moment can greatly alter the result, online randomization can help to train a robust and generalized policy against various joint locking accidents.

Joint Failure in the Real World. Quadrupedal robots are complex machines, and modifying the hardware can be dangerous without the support from the manufacturer, especially when joint failure is intentionally added to the system. To safely evaluate the fault-tolerant controller on the physical robot, both hardware and software methods are used to simulate the joint locking situation in the real world.

(A) Real world Installation (B) 3D model

FIGURE 4.3: The 3D-printed joint locking mechanism assembled in the physical device A, containing two mounts for thigh link and calf link B for rod connection to form a locking situation.

For *hardlock*, we design and 3D print an external locking mechanism (Figure 4.3) to directly limit the motion of the joint. Although such locking is closer to the real situation, it cannot be easily used on every joint due to the hardware design of the quadruped. For most quadruped, such as Unitree A1, only the calf joint of each leg is exposed in the open space and the mechanism can be attached before the experiment, which could limit the diversity of test cases.

To address this limitation, *softlock* is also used to simulate joint locking. Similar to the simulator, the joint position after failure occurs is tracked. The desired joint position from the controller is clipped in the range of $\theta_{allowed}$ before being passed to the onboard PD controller for torque output. With *softlock*, joint failure can occur at any time and at any joint during the experiment. However, with a modified command, the real-world simulation can be different from the actual situation. Both methods are tested and evaluated in Chapter 4.3.4

4.2.2 Reinforcement Learning Architecture

The fault-tolerant controller is trained with reinforcement learning, which takes data from the common onboard sensor as observations and outputs the optimal actions as joint position.

Observation. Data from the equipped low-level sensor are collected to provide observations. At any time t, joint encoders, IMU and foot encoders provide noisy

TABLE 4.1: Reward terms in our RL training methodology

Category	Term	Formula	Scaling
Movement	Forward	$min(v_t^x, 0.5)$	1.0
	Lateral movement	$\|v_t^y\|^2$	-1.5
	Yawing	$\|\omega_t^z\|^2$	-0.5
Stability	Base bumping	$\|v_t^z\|^2$	-2.0
	Base rolling	$\|\omega_t^{x,y}\|^2$	-0.02
	Base orientation	$\|g_t^{x,y}\|^2$	-1.0
	Joint off limit	$\mathbb{1}_{q>q_{max}\|q<q_{min}}$	-10.0
	Collision	$\mathbb{1}_{collision}$	-1.0
Smoothness	Joint acceleration	$\|\dot{q}_t - \dot{q}_{t-1}\|^2$	-1.0e-8
	Power consumption	$\int_{t-1}^{t} \|\tau \cdot \dot{q}\| dt$	-5.0e-3
	Feet air time	$\sum(t_{air} - 0.5)$	1.0

sensor data $x_t \in \mathbb{R}^{30}$, which consists of joint position $q \in \mathbb{R}^{12}$, joint velocity $\dot{q} \in \mathbb{R}^{12}$, gravity vector $g \in \mathbb{R}^2$ and binary foot contact $c \in \mathbb{R}^4$. The previous actions $a_{t-1} \in \mathbb{R}^{12}$ are further added to form the observation $o_t = [x_t, a_{t-1}] \in \mathbb{R}^{42}$. Following recent works [3, 4, 8], a historical observations of length $H = 50$ ($[o_{t-H+1}, ..., o_t]$) is used to capture the temporal information.

Action. The control policy π predicts the target joint position $\hat{q} = a_t \in \mathbb{R}^{12}$, which is consequently processed by a PD controller for the desired torque τ of each actuator.

$$\tau = K_p(\hat{q} - q) + K_d(\hat{\dot{q}} - \dot{q}) \tag{4.5}$$

where K_p and K_d are the stiffness and damping gain controlled by DR. The target joint velocity $\hat{\dot{q}}$ is set to 0.

Reward Function. Closely following [3, 4], the reward functions encourage the agent to move forward stably and smoothly with a target speed of 0.5 m/s. Penalization is given mainly for movement in other axes, such as lateral movement and yawing, large joint acceleration, power consumption, and collision with robot body. No special reward to guide the quadruped to handle the locked joint is designed in addition. Details of each reward term can be found in Table 4.1

Terrain Generator and Curriculum: We use the terrain generator from [24] to simulate uneven terrains. Three types of terrains are included in our simulation: 1. rough sloped terrain; 2. smooth sloped terrain; 3. flat terrain with discrete

obstacles. The terrains are generated with a curriculum strategy. Specifically, each type of terrain has 10 levels, with increased difficulty (larger slope, higher roughness, and higher obstacles). We track the assigned level of each agent and update the level based on the total travel distance during its lifespan. Despite the task of moving forward, the total distance is used because it indicates the basic maneuverability of the agent. If any agent has solved all the terrain levels, it will be looped back to level 1 for further exploration.

Domain Randomization. Besides the failure simulation described in Chapter 4.2.1, ground friction, PD controller settings, payload, and motor strength are also randomized to add robustness for various situations. Table 4.2 summarizes the range of different randomized factors during training.

TABLE 4.2: Domain Randomization parameters

Factor	Range	Unit
Friction	$[0.05, 4.5]$	-
K_p	$[50, 60]$	-
K_d	$[0.4, 0.8]$	-
Payload Mass	$[0, 6]$	kg
Payload CoM	$[-0.1, 0.1]$	m
Motor Strength	$[90, 110]$	%
Failure Joint	$\{0, \ldots, 12\}$	-

4.2.3 Joint Teacher-Student Framework

Our goal is to obtain a zero-shot policy, which is trained completely in the simulator and transferred to the real world without fine-tuning. For RL-based policies, the privileged underlying states of the robot and environment can produce better performance in a smaller number of training iterations [83, 84].The teacher-student learning paradigm [3, 4, 8] is adopted to achieve this goal. It enables implicit identification of the hidden dynamics of the environment and robots e_t for different behaviors. It also learns dynamics from perceivable data, making the policy deployable in the real world.

Specifically, a teacher model μ encodes the environmental factor e_t into the latent space representation z_t with length D:

$$z_t = \mu(e_t) \in \mathbb{R}^D \tag{4.6}$$

To better capture the dynamics, e_t contains necessary underlying ground-truth synthetic data that are accessible from the simulator including the DR parameters d_t, clean robot states $s_t = [v_t, \omega_t]$ and height map m_t of the surrounding terrain.

A student model ϕ is also introduced to learn from historical observations to mimic the encoding from μ by performing system identification:

$$\hat{z}_t = \phi(o_{t-H:t}) \in \mathbb{R}^D \tag{4.7}$$

To optimize ϕ, previous works [3, 4, 8] focus on imitating μ's behaviors only by using supervised learning inspired by DAgger [58]. With the trajectory generated by trained μ or randomly initialized ϕ in the online or offline fashion, $\mathcal{L}_{adaption} = \|z_t - \hat{z}_t\|^2$ is minimized and used with previously trained π. However, it is almost impossible to get an exact latent representation such that $\hat{z}_t = z_t$. The difference in the latent representation can cause unpredictable behaviours, and performance degradation. To address this issue and minimize uncertainty, we propose to fuse the output of μ and ϕ with an adaptive ratio α to jointly optimize the policy network with the student network:

$$z'_t = \alpha\hat{z}_t + (1-\alpha)z_t \tag{4.8}$$

$$a_t = \pi(z'_t, o_t) \tag{4.9}$$

During training, α is updated with the progression of policy optimization. In the early stage, we set $\alpha = 0$ to train π by leveraging the privilege information encoded by μ. With the training going on, we gradually increase α, until only $\hat{z}_t$ is used for policy making in the late stage. Thus, even if we cannot get a perfect replica of z_t, π can still have the opportunity to learn to adapt to such a difference in a single phase to maximize the reward.

We choose PPO [140] for RL optimization as it is commonly used in recent works [4, 9, 151] and demonstrate stable performance in robotics control, which can provide

a general indication of the performance of the proposed method. To train the adaption jointly, we append the PPO loss $\mathcal{L}_{RL}$ with $\mathcal{L}_{adaption}$, which is similar to [20, 152]:

$$\mathcal{L} = \mathcal{L}_{RL} + \beta \mathcal{L}_{adaption} \tag{4.10}$$

The adaptive ratio β is negatively correlated with α. As the proportion of the student output increases for policy making, the focus can be shifted more on the reward benefits rather than mimicking the teacher's behavior. The details of PPO parameters can be found at Table 4.3.

TABLE 4.3: PPO Hyperparameter

Hyperparameter	**Value**
Discount factor	0.99
GAE parameter	0.95
Clip range	0.2
Value coefficient	1.0
Entropy coefficients	0.001
Learning rate	adaptive [24]
KL threshold	0.008
Horizon length	24
Minibatches	6
Mini epochs	5
Optimizer	Adam
Total Frames	300M

4.3 Evaluation

4.3.1 Implementation and Experimental Setup

Module Implementation. Both the teacher model μ and control policy ϕ are implemented in MLP with hidden layers of $[512, 256, 128]$ and $[256, 128]$, respectively, and ELU activation. μ outputs the latent representation with length $D = 8$. Two different student models are implemented to capture temporal information, one with vanilla 1D CNNs following [3] and another with TCN [78] following [8].

Simulation. Isaac Gym and its open source library IsaacGymEnvs [2] are used to simulate massive parallel environments with rough terrains, including rough sloped

terrain, smooth sloped terrain and discrete obstacles [24]. The simulation runs on two NVIDIA A6000 GPUs, each handling 4096 environments at 200Hz, which can provide more than 0.1M FPS for simulation. The controller runs at 50Hz for command.

Hardware. Unitree A1 is used as the test platform. It is a low-cost quadruped driven by 12 direct-drive actuators, equipped with IMU and foot-end force sensors used as observations. We use an external NVIDIA Jetson Xavier NX for GPU acceleration to process the exported JIT model.

Baselines. To evaluate the impact of joint locking, different variants of the following [**T**]eacher networks are trained:

- **BaseEnv**[**T**]: The teacher network trained in vanilla `BaseEnv` with privilege information.
- **FailureEnv**[**T**]: The teacher network trained in `FailureEnv` with privilege information as a baseline of fault-tolerant performance.

Each teacher model is trained from scratch with the same environment and PPO configuration.

To evaluate the efficiency of knowledge transfer and performance in terms of reward, forward velocity, and survival time for fault-tolerant control, various [**S**]tudent policies are trained in both `BaseEnv` and `FailureEnv` with the proposed joint training ([**JT**]) and separate supervised ([**SS**]) from RMA [3]:

4.3.2 Teacher-Student Transfer

The teacher agent cannot be directly deployed on the physical robot due to the usage of privilege information, which is either unavailable or expensive to acquire in real world. Therefore, knowledge transfer is needed to train student policies. To better demonstrate the efficiency of our proposed joint training, the policy with both CNN and TCN student network are transferred under different frame configuration:

- [**T**]**:** this is trained in two configuration: **600M**, which is the default frame count; and **300M** with half of the simulated frame for usage in limited frame supervised transfer.

TABLE 4.4: Normalized Reward return for different Student policy.

Environment	[T]		[JT]		[SS]			
	600M	300M	600M		300M+300M		600M+600M	
			CNN	TCN	CNN	TCN	CNN	TCN
BaseEnv	1	0.92	0.89	0.92	0.15	0.77	0.85	**0.93**
FailureEnv	1	0.85	**0.97**	0.85	0.20	0.73	0.81	0.84

- **[JT]:** this is is trained with standard 600M total frames.
- **[SS]:** this is fully trained with **600M+600M** frames to achieve optimal performance. To compare the performance under the same total frame as **[JT]**, an additional configuration of **300M+300M** is used with teacher and supervised training stages equally divided.

For supervised transfer, we follow RMA [3] to unroll the teacher policy to generate trajectories and ground truth for the student. The trained policies are then deployed in the simulation and the average return is shown in Table 4.4. To better show the difference in performance, the result are normalized based on the corresponding teacher performance for `BaseEnv` and `FailureEnv`.

Despite supervised transfer showing great performance while fully trained, especially with TCN, it struggles when the total simulation step is limited, and there is a signification return drop. For joint training, it can achieve the same level or even outperform **[SS]** transfer with only half of the simulation step required in both `BaseEnv` and `FailureEnv` indicating a superior efficiency in knowledge transfer.

4.3.3 Virtual Deployment

Overall Performance. Both `BaseEnv` and `FailureEnv` student policies are deployed into the same test environment where robots are spawned across different terrains and levels evenly with joint locking failure occurs randomly. Each virtual robot can run a maximum of 20 seconds after joint failure occurs. The forward velocity both before and after joint locking are tracked and the survival time of each agent is measured on average, 25% percentile (P25) and 50% percentile (P50)

TABLE 4.5: Agent performance with Joint Failure in Simulation.

Agent	Terrain	Avg. Forward Velocity (m/s)		Survival Time (%)		
		Before	After	Average	P25	P50
BaseEnv	Smooth Slope	0.56	0.45	51.4	6.7	36.5
	Rough Slope	0.55	0.41	44.4	4.7	20.7
	Discrete	0.54	0.41	40.8	4.4	17.45
	All	0.55	0.42	44.7	5.0	21.35
FailureEnv	Smooth Slope	0.59	0.52	68.3	20.1	100.0
	Rough Slope	0.57	0.47	59.1	11.7	81.0
	Discrete	0.55	0.44	45.8	6.6	31.0
	All	0.57	0.47	56.5	10.8	59.0

so that we can see how each agent handles joint locking in the worst scenarios. The result averaged over 1500 instances per terrain is shown in Table 4.5.

Before joint failure, both agents can drive the robot forward close to the target velocity of 0.5 m/s. After failure occurs, the velocity drops in both agents, but the fault-tolerant `FailureEnv` agent maintain the velocity slightly better. During deployment, the critical failure mostly kills the robot within seconds after joint locking, and the surviving instance can normally remain until the end, thus increasing the average survival time. Despite `BaseEnv` agent can still walk with a reasonable velocity after joint locking, it is more vulnerable to joint locking and fails within 5s for half of the instances. In contrast, the fault-tolerant `FailureEnv` agent can survival much longer with a locked joint. In smooth slope and rough slope terrains, even with joint failure, most of the robot can survive to the end of the journey. Robots in discrete obstacle terrain have significantly worse performance. Due to the small physical size of the A1 robot, it is too difficult for it to step up and down even under normal hardware conditions [9, 151].

Gait Pattern Analysis. To understand how `FailureEnv` agent handles joint failure, the gait pattern of foot contact is captured during deployment around the failure moment, with F/R denoting front/rear and L/R denoting left/right. The patterns of two instances are shown in Figure 4.4. With an unexpected joint locking failure, the agent can quickly adapt based on the actual situation and react accordingly. While the top instance can keep the previous gait, if the joint is locked in an extremely location, the agent will adjust the motion to even drag the

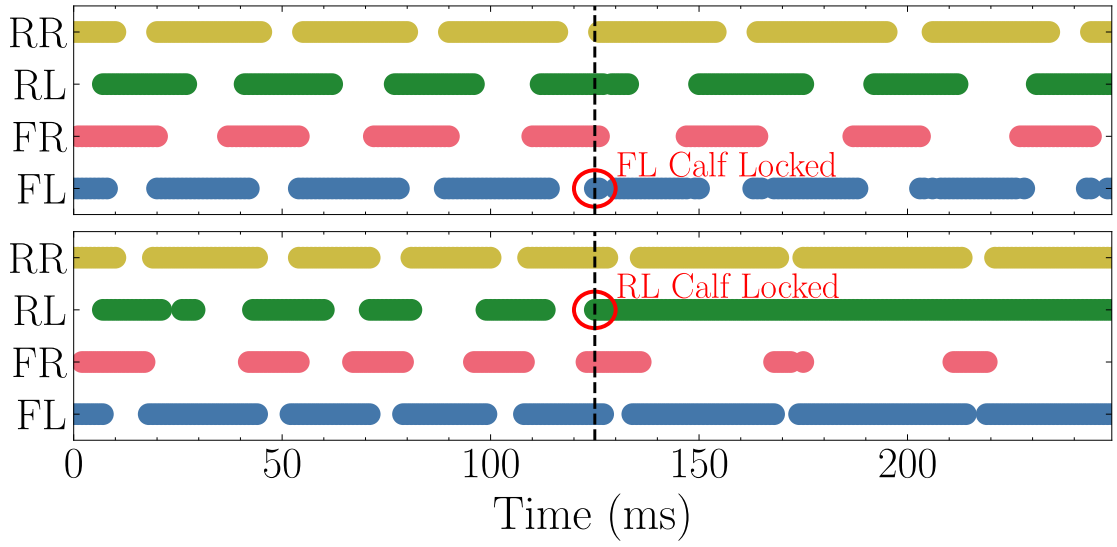


FIGURE 4.4: Gait pattern of two instances during virtual deployment before and after joint locking.

failure leg forward, as shown in the bottom instance. The dynamic adjustment for different situations demonstrates the generalization ability of `FailureEnv` agent in the handling of failures without relaying on some predefined pattern.

Failure Case Study. To further understand how joint locking affects control and leads to failure, the most vulnerable instances of virtual deployment that fail within seconds for both agents are identified. Figure 4.5 shows the distribution of the failure joint in the worst scenarios. The thigh and calf are the most vulnerable joint for `FailureEnv` and `BaseEnv` agents respectively. Both joints have larger movements compared to the hip joint, making them more sensitive to joint locking. The distribution shift reflects that while `FailureEnv` agent learns to overcome the locking of the calf joint, the thigh joint is still not fully handled.

We then track the joint status of these failure cases in Figure 4.6. Our proposed simulation strategy can effectively limit joint movement as desired in both position and velocity. With a locked joint, when the desired position is not in the range of $\theta_{allowed}$, large torque and fast joint jitter are observed, which are two major factors of critical failure.

Performance with *softlock*. To better evaluate generalization performance, a *softlock* virtual environment is implemented following Section 4.2.1. Although such an environment is never seen during training, the `FailureEnv` agent can easily handle the new type of failure with 5% greater average reward return and 8% longer

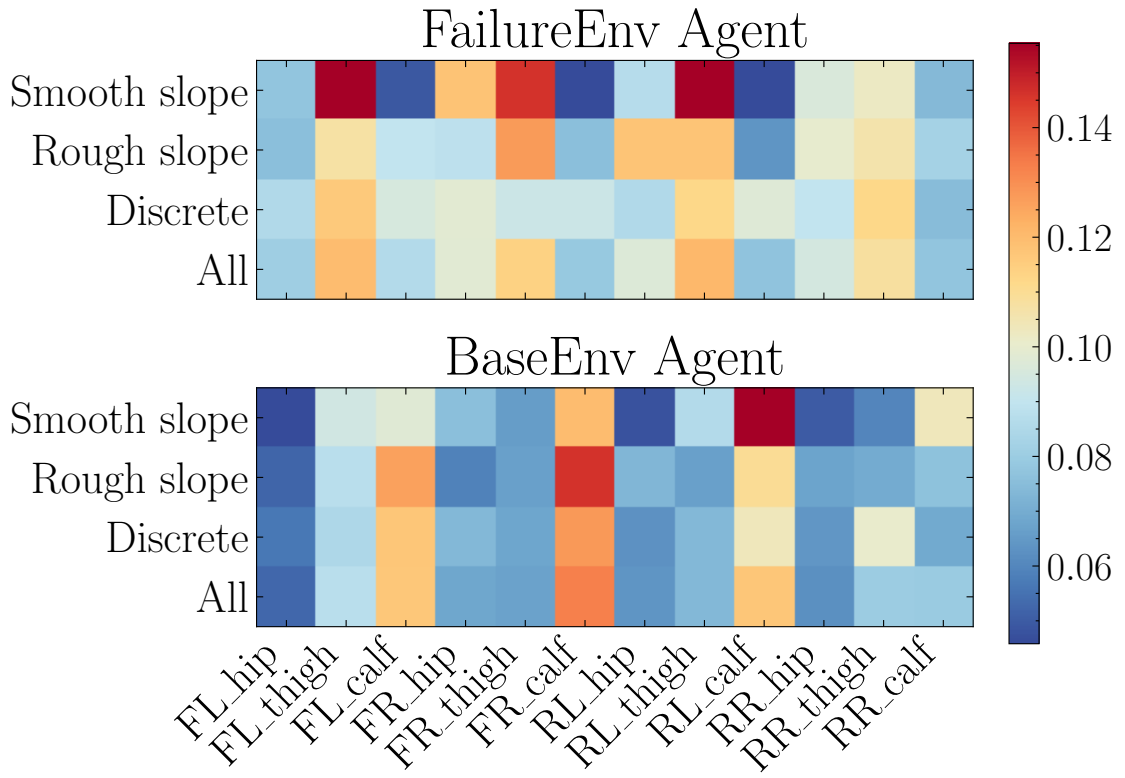


FIGURE 4.5: Joint distribution of failure joint in the worst cases of `FailureEnv` agent and `BaseEnv` agent

average survival time comparing with *hardlock*. With the clipping of policy output, the resulted torque and velocity is smaller and smoother. However, the limited motion of joint is still a major factor of critical failure.

Multiple Joint Failure. We further evaluated the ability of `FailureEnv` agent to handle multiple joint locking failures, which is never seen during training. The survival time and the average forward velocity after failure are tracked for both random failure and whole leg failure in Figure 4.7.

Although the `FailureEnv` agent can easily handle two joint locks simultaneously, it becomes more difficult to maintain the heading. With more joints locked, the agent begins to struggle, especially when the failure joints are distributed across multiple legs. When all failures occur on the same leg, even with all three joints locked, resulting in the loss of a whole leg, the agent can compensate for the loss with other legs. The difference in performance between legs indicates that the current agent may not be perfectly symmetrical. The uniform randomization treats all failure cases equally, overlooking the fact that some scenarios are more dangerous and critical. As the policy attempts to cover all possible failure cases, it risks converging

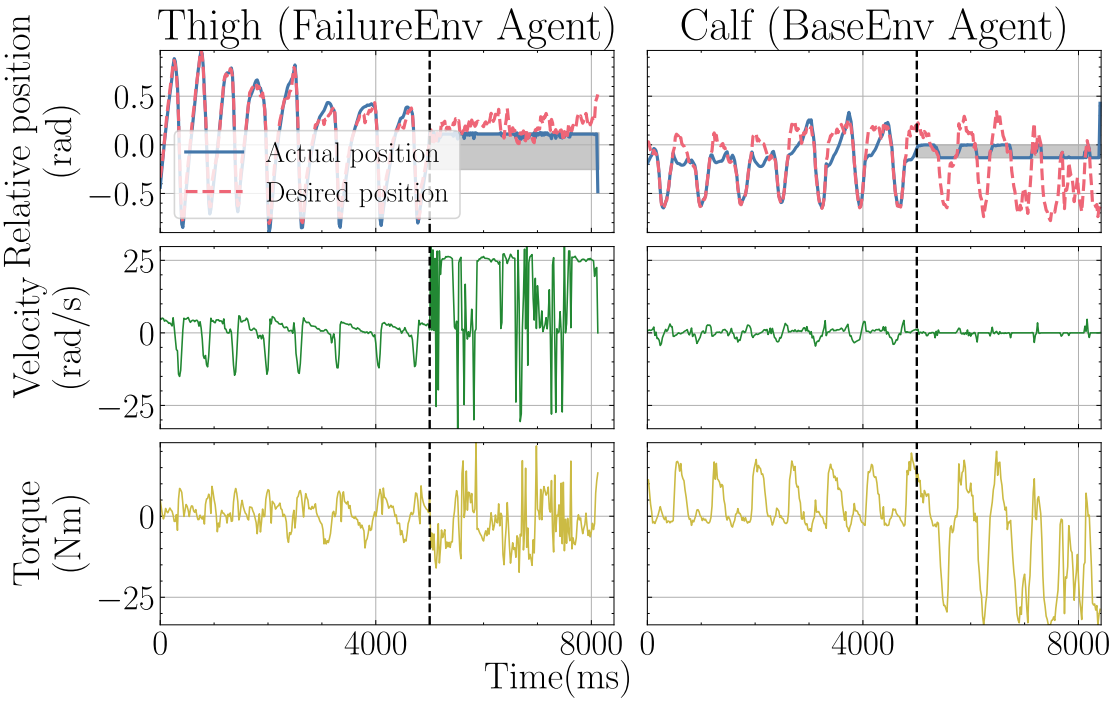
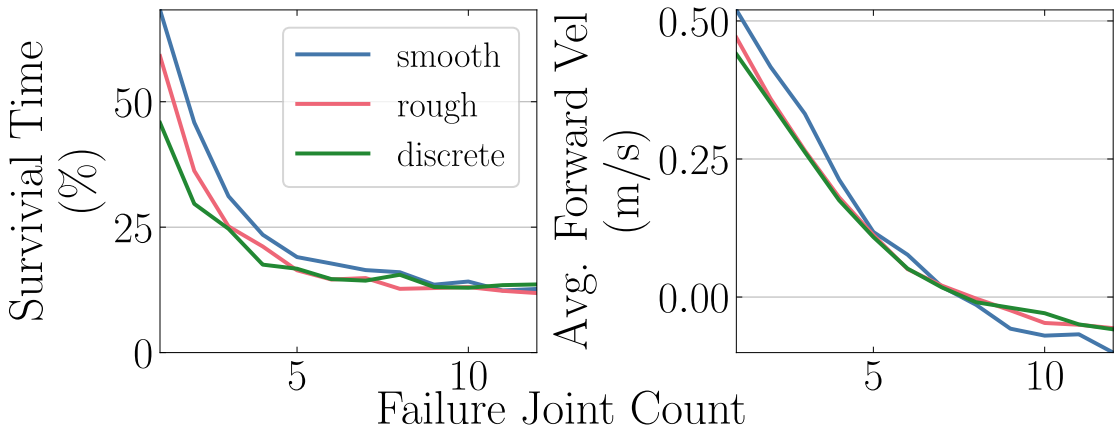


FIGURE 4.6: Motion of the most vulnerable joint for `FailureEnv` and `BaseEnv` agent identified in Figure 4.5. The timeline is intercepted from 5 seconds before joint locking to critical failure, where the agent needs to be reset. The grey box shows the limited motion range $\theta_{allowed}$. The joint position is relative to the default position for standing.

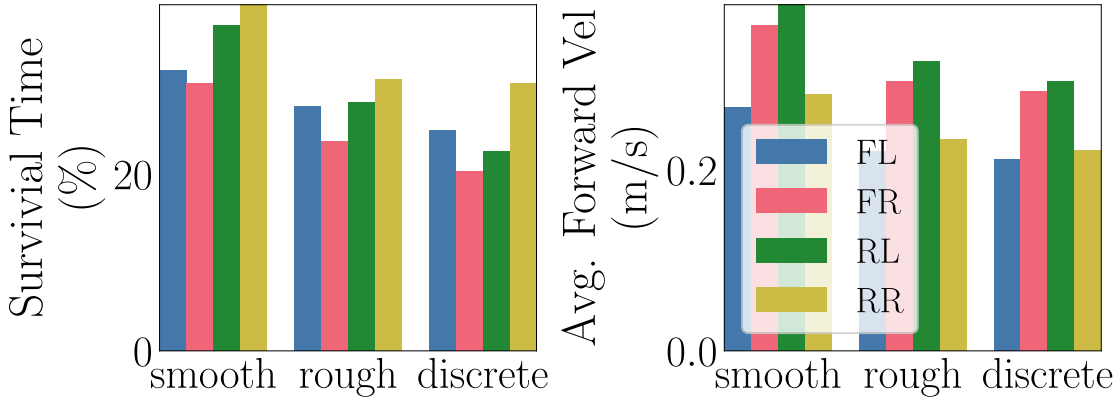
to a suboptimal equilibrium that compromises overall gait quality. Future work should investigate more structured curriculum strategies that prioritize critical failure scenarios, along with gait-aware optimization objectives, to achieve more balanced and effective fault-tolerant performance.

4.3.4 Physical Validation

RL-Based Controller. To compare the trained RL-based controllers, the trained model is converted to JIT and deploy it directly on the physical Unitree A1 for zero-shot transfer without any fine-tuning. We compare the proposed fault-tolerant `FailureEnv` agent with the baseline model trained in `BaseEnv` and the built-in A1 controller. Following Section 4.2.1, both *hardlock* and *softlock* are used during the deployment for comprehensive validation. Figure 4.8 shows the snapshots of the trials.



(A) Random Multiple Joint Failure



(B) Whole Leg Joint Failure

FIGURE 4.7: Performance with multiple joint failure for A random joint failure and B whole leg failure.

Similar to virtual deployment, the joint motion is tracked during deployment. Figure 4.9 shows the motion of trails run by `FailureEnv` agent for both *softlock* and *hardlock* on all eligible joints. Both locking methods can limit the joint motion and show a similar pattern as the simulation results in Figure 4.6. Due to the manipulation of target joint position, *softlock* tends to have a much lower torque and, sometimes, the joint will overshot the locking range, making it less ideal and dangerous. But it is still sufficient for real-world validation on joints where *hardlock* cannot be applied as discussed in Chapter 4.3.3. For *hardlock* on calf joint, the motion is limited to a range of around 0.15 rad and the large torque pattern similar to the virtual deployment is observed, demonstrating that the designed locking mechanism is efficient for real-world testing.

We further measure the survival time in the real world with a maximum lifetime

(A) Running with joint locking by the `FailureEnv` agent

(B) Running with joint locking by the `BaseEnv` agent

(C) Running with joint locking by built-in controller

FIGURE 4.8: Deployment snapshots on the physical robot run by A fault-tolerant `FailureEnv` agent, B baseline `BaseEnv` agent and C A1's built-in controller. The safety rope is only used to prevent hardware damage and does not affect running. Refer to the supplementary video for more information.

TABLE 4.6: Average Survival time in physical tests under different joint locking

Agent	Softlock			Hardlock
	Hip	Thigh	Calf	
FailureEnv	100%	100%	100%	100%
BaseEnv	100%	20%	5%	35%
Built-in	-	-	-	0%

of 20 seconds in Table 4.6. The fault-tolerant `FailureEnv` agent can handle all the test seniors while the vanilla `BaseEnv` agent struggles on thigh and calf joint, which is in line with the observations of the virtual deployment in Chapter 4.3.3, and the robot stalls or falls directly to the ground. We further lock two joints for `FailureEnv` policy and the quadruped can still move safely, even if this situation is never seen during training.

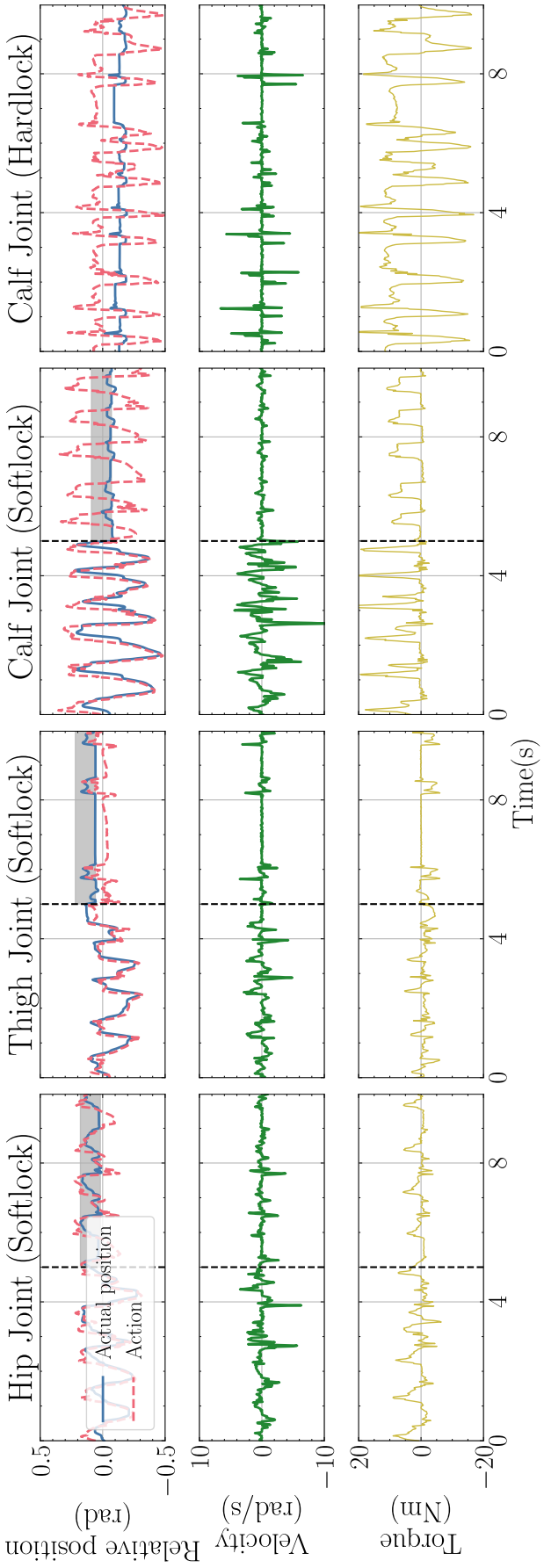


FIGURE 4.9: Joint motion of the locked joint under both *softlock* and *hardlock* run by `FailureEnv` agent. For *softlock*, 10 seconds around the joint locking timestamp are intercepted, with the allowed movement range $\theta_{allowed}$ showing as a gray box. For *hardlock*, the joint is locked at the beginning of the run. The joint position is relative to the default position for standing.

MPC-Based Controller. Further evaluation is conducted with two MPC based controller: (1) Unitree A1's built-in proprietary controller. (2) an OCS2 [77] based controller. Both controller is tested with **hardlock** while it is commanded to move forward with a velocity of 0.5 m/s.

The built-in controller cannot handle the locked joint and fall directly to the ground. The OCS2 based controller can efficiently handle the joint failure most of the time while still falling at the beginning from time to time and have large lateral movement. MPC-based controllers required manually tuning and configuration, which requires prior knowledge, while the RL-based controller can be trained in simulation within hours, greatly reducing the development challenging to achieve fault-tolerant quadruped controller.

4.4 Conclusion

In this chapter, we propose a novel methodology to train and test hardware fault-tolerant RL-based controllers for quadruped locomotion, both in the simulation and real world. We design a novel simulation strategy for joint locking failures and a joint training pipeline to efficiently train a fault-tolerant quadruped locomotion controller with the teacher-student framework. We use commonly equipped low-level sensors available on quadrupedal robots as observations to generate robust actions. We demonstrate that even with one joint locked, our controller can still drive the quadruped without losing too much heading or speed.

Fault-tolerant control is a complex topic that depends heavily on robot specifications and deployment conditions. Domain randomization has become an important technique in robot learning to improve robustness against various anomalies including sensor drift [24] and system delays [139]. Following our work, recent studies have further extended this idea to other scenarios, such as weakened motors and actuator degradation [153]. However, such approaches require training new teachers whenever novel failure scenarios arise, limiting their scalability. More broadly, over-randomization introduces a well-known trade-off between robustness and performance [97] that excessively diverse failure scenarios can reduce learning efficiency and lead to suboptimal policies that sacrifice nominal performance in exchange for broader fault coverage. Meta-learning is also a promising direction, where

task-specific teacher policies are trained for different failure conditions and distilled into a single student policy [149, 154]. How to effectively balance the scope of failure randomization with training efficiency and final policy quality remains an open question that warrants further study.

Chapter 5

Unified Locomotion Transformer with Simultaneous Sim-to-Real Transfer for Quadrupeds

Quadrupeds have gained rapid advancement in their capability of traversing across complex terrains. The adoption of deep Reinforcement Learning (RL), transformers and various knowledge transfer techniques can greatly reduce the sim-to-real gap. However, the classical teacher-student framework commonly used in existing locomotion policies requires a pre-trained teacher and leverages the privilege information to guide the student policy. With the implementation of large-scale models in robotics controllers, especially transformers-based ones, this knowledge distillation technique starts to show its weakness in efficiency, due to the requirement of multiple supervised stages. In this chapter[1], we propose Unified Locomotion Transformer (`ULT`), a new transformer-based framework to unify the processes of knowledge transfer and policy optimization in a single network while still taking advantage of privilege information. The policies are optimized with reinforcement learning, next state-action prediction, and action imitation, all in just one training stage, to achieve zero-shot deployment. Evaluation results demonstrate that with `ULT`, optimal teacher and student policies can be obtained at the same time, greatly easing the difficulty in knowledge transfer, even with complex transformer-based models. Supplementary video available at https://johnliudk.github.io/ult/.

[1]The content of this chapter is published in [155].

5.1 Introduction

Driven by the advancement of deep reinforcement learning (RL), quadruped robots have drawn great attention, due to their capability of traversing across complex terrains [3, 7, 8, 27]. Traditional robotics controllers rely on dedicated models and heuristics, which require extensive prior knowledge and can struggle to adapt to dynamic environments and unpredictable situations [3, 8]. Recently, a new paradigm is introduced to push quadrupeds' limit to handle complex tasks in challenging environments [59–61, 63]: a controller is learned from simulations that contain various environmental and physical factors, and then transferred to the physical robot through RL and various knowledge transfer techniques [3, 8, 9, 20, 156] to overcome the sim-to-real gap using privilege information.

One of the commonly used knowledge distillation methods is the teacher-student framework [8]. A teacher policy is first trained through RL, and privilege information about the environment is provided for learning the optimal locomotion strategies efficiently. As additional information, including ground-truth dynamics and terrain profiles, is often inaccessible in the real world, a student policy is subsequently trained to make the model deployable. This is typically achieved through supervised knowledge distillation, either in an offline [8] or online [3] fashion by creating a data set with new trajectories and associated action labels of the teacher created through algorithms such as Data Aggregator (DAgger) [58].

Sequential training with policy imitation for locomotion control is generally not data-efficient [157]. The performance of the student is limited by the teacher policy and the robustness depends on the diversity of the data set. Generally, they will deteriorate if the situation deviates from the trajectories in real-world deployment. To address this, DreamWaQ [139] was introduced with the asymmetric actor-critic architecture [158] and context estimation, including next observation, base velocity and latent space. By concurrently training with proprioceptive observation for exploration and privileged information as the critic, it allows the agent to explore with indirect guidance.

Recently, self-attention-based transformers [42] have been widely introduced into robotics in both lower control (e.g., legged locomotion directly [9, 10, 20, 131] or with a command interface [128]) and high-level decision making with multimodal

processes [36, 43, 53, 54] with their capability to handle variable context lengths, sensor combinations [131] and even robot embodiments [133]. In direct locomotion controllers, transformers have demonstrated superior capability in temporal information capture compared to recurring neural networks (RNN) and temporal convolutional networks (TCN). However, training transformers is data-hungry in general. For example, when combing the vanilla transformer with legged locomotion, Lai et al. [9] generated an additional 40M timesteps for 400K updates using a two-stage supervised training approach. Similarly, Radosavovic et al. [20] doubled the number of timesteps for joint supervised transfer compared to the teacher policy training stage. These approaches are not only time-consuming, but also necessitating additional setup to handle multiple models and the large volumes of generated data.

Motivated by the above limitation, the objective of this chapter is to simplify the knowledge distillation process with a unified architecture to keep teacher and student policies in a single network for simultaneous optimization. Our inspiration comes from the exceptional capabilities of transformers in multimodal modeling of temporal and sensory information and context understanding [10], which can well support the policy optimization [139]. Thus, we can introduce the privilege information into the observation as another modality to form a unified framework for single-phase optimization of teacher and student policies, achieving zero-shot sim-real transfer.

To this end, we propose Unified Locomotion Transformer (`ULT`), a new unified framework for end-to-end quadruped locomotion. It is based on the standard transformer architecture with casual masking to pack teacher and student policies in a single network. These policies are optimized jointly with reinforcement learning, next state-action prediction, and action imitation, all in just one training phase, to overcome the sim-to-real gap and achieve zero-shot deployment. In this way, we eliminate the need of dedicated design and training of a teacher network, while the privilege information can still efficiently guide the student policy with proprioception observation during the exploration to generate more diverse trajectories to improve the overall generalization and robustness.

We extensively evaluate our framework in simulation and compare it with state-of-the-art knowledge transfer baselines. We also deploy it directly in the real world for practicality validation. Evaluation results demonstrate that `ULT` exhibits

better performance with less trajectory information needed, indicating its higher efficiency with the help from next state-action predication, action imitation and mixed exploration. With unified training in one single phase for simultaneous teacher and student policy optimization, we ease the pipeline of knowledge distillation for zero-shot sim-to-real transfer.

5.2 Simulation Environment

In this chapter, we implement the simulation environment in Isaac Gym [2] and IsaacGymEnvs [2] to train locomotion agents in large scale parallelism. All baselines and variants of our method are trained with the exact same simulation setups to ensure fair comparison.

5.2.1 Terrain and Curriculum

To ensure that the policy is robust against different indoor and outdoor environments, we adopt the terrain curriculum from [24] with smooth slope, rough slope, stairs up, stairs down and discrete obstacle terrains. Each type of terrain has 10 levels with incremental difficulty and an overall proportion of $[0.1, 0.1, 0.35, 0.25, 0.2]$, respectively. The linear velocity return is tracked across each trajectory's life cycle. The level is considered solved when an agent reaches 80% of the maximum achievable tracking reward and progresses to the next level. If any agent fails to reach 25% of the maximum reward, it will regress to a lower level for more exploration.

5.2.2 Domain Randomization

Following [24, 139], we apply Domain Randomization (DR) on key dynamics parameters to enhance the robustness of the policy. To simulate the actual user commands, we sample the linear commands in longitudinal and lateral direction separately with a uniform distribution in $[-0.5, 0.5]$ m/s. For the angular command, we first sample the desired heading of the robot and cap the resulted angular velocity command at 0.5 rad/s. The commands are resampled every 10 seconds. Table 5.1 lists the key parameters of DR used in the simulation.

TABLE 5.1: Simulation parameters of domain randomization.

Parameters	**Range**	**Unit**
Linear Command	[-0.5, 0.5]	m/s
Angular Heading	[-3.14, 3.14]	rad
K_p Scale	[0.9, 1.1]	-
K_d Scale	[0.9, 1.1]	-
Friction Scale	[0.7, 1.3]	-
Motor Strength Scale	[0.9. 1.1]	-
Payload	[0, 5]	kg
Payload CoM Offset	[-0.1, 0.1]	m
External Push	[-1, 1]	m/s
Gravity	[9,41, 10.21]	m/s^2
System Delay	[0, 0.015]	s

5.2.3 Observations and Actions

Privilege Information. To achieve an optimal locomotion policy with the hidden information about the environment, related privilege data is extracted from simulation to form the privilege observation e_t for the teacher policy to utilize. The privilege information contains randomized dynamics parameters d_t sampled from Chapter 5.2.2, ground truth robot states s_t including base velocity, orientation, and precise surrounding height map m_t. Although such information is often inaccessible in the real world, it helps the teacher policy reconstruct the states and improve the learning efficiency [3, 8].

Proprioceptive Observation. In order to conduct knowledge transfer for a deployable agent, the student policy only relies on onboard sensors to provide observations. Typically, quadrupedal robots are equipped with joint encoders, IMU, and foot contact sensors, which can provide information on joint position $q \in \mathbb{R}^{12}$, joint velocity $\dot{q} \in \mathbb{R}^{12}$, angular velocity $\omega \in \mathbb{R}^3$, gravity vector $g \in \mathbb{R}^3$ and binary foot contact $c \in \mathbb{R}^4$. In order to follow the user command, the agent also needs access to the randomly sampled cmd $= [v_x, v_y, \omega_z] \in \mathbb{R}^3$ to form the observation of each step $o_t = [q, \dot{q}, \omega, g, c, \text{cmd}] \in \mathbb{R}^{37}$. To provide the state transition and temporal information, the actions of the previous step $a_{t-1} \in \mathbb{R}^{12}$ are added with a list of historical information $\mathcal{T} = [a_0, o_1, a_1, o_2, \cdots, a_{t-1}, o_t]$. We use a rolling window of $t = 15$, resulting in a full observation in the space of $\mathbb{R}^{49\times15}$.

Actions. As typical RL locomotion policies perform inference at the frequency of 50-100 Hz, both the teacher and student policies output the desired joint position

TABLE 5.2: Reward terms for reinforcement learning

Reward	Definition	Scale
Linear Velocity Tracking	$\exp\left(-5\|v_{xy}^{\text{cmd}} - v_{xy}\|^2\right)$	1.0
Angular Velocity Tracking	$\exp\left(-5(\omega_z^{\text{cmd}} - \omega_z)^2\right)$	0.5
Body Z Velocity	$\|v_z\|^2$	-2.0
Body Rotation	$\|\omega_z\|^2$	-0.05
Joint Acceleration	$\|\ddot{\theta}\|^2$	-2.5e-7
Output Work	$\int \|\tau \cdot \dot{\theta}\|$	-2.e-5
Action Rate	$(a_t - a_{t-1})^2$	-0.05
Feet Slip	$\|g_t \cdot v_{xy}^{feet}\|$	-0.1
Collision	$\mathbb{1}_{collision}$	-1

a_t, which is passed to a PD controller running at a much higher frequency for a smooth torque output:

$$\tau = K_p(\hat{q} - q) + K_d(\hat{\dot{q}} - \dot{q}) \tag{5.1}$$

with base stiffness K_p and damping K_d set to 30 and 0.7, respectively, and additional DR is added on. The target joint velocity $\hat{\dot{q}}$ is set to 0.

5.2.4 Reward Function

We follow the classic reward function design for omni-direction locomotion [3, 24, 139] to encourage the agent to follow the commanded velocity and primarily penalize the linear and angular movement along other axes, large joint acceleration and excessive power consumption. The complete reward structure is detailed in Table 5.2.

5.3 Methodology

We present `ULT`, a transformer-based framework to unify the process of knowledge transfer and policy optimization in a single network leveraging the privilege information. Compared to the classic teacher-student transfer solution that requires a pre-trained teacher policy [3, 8, 9], `ULT` optimizes both policies jointly in a single phase to simplify the sim-to-real pipeline and reduce the total number of generated trajectories. Figure 5.1 shows an overview of `ULT`.

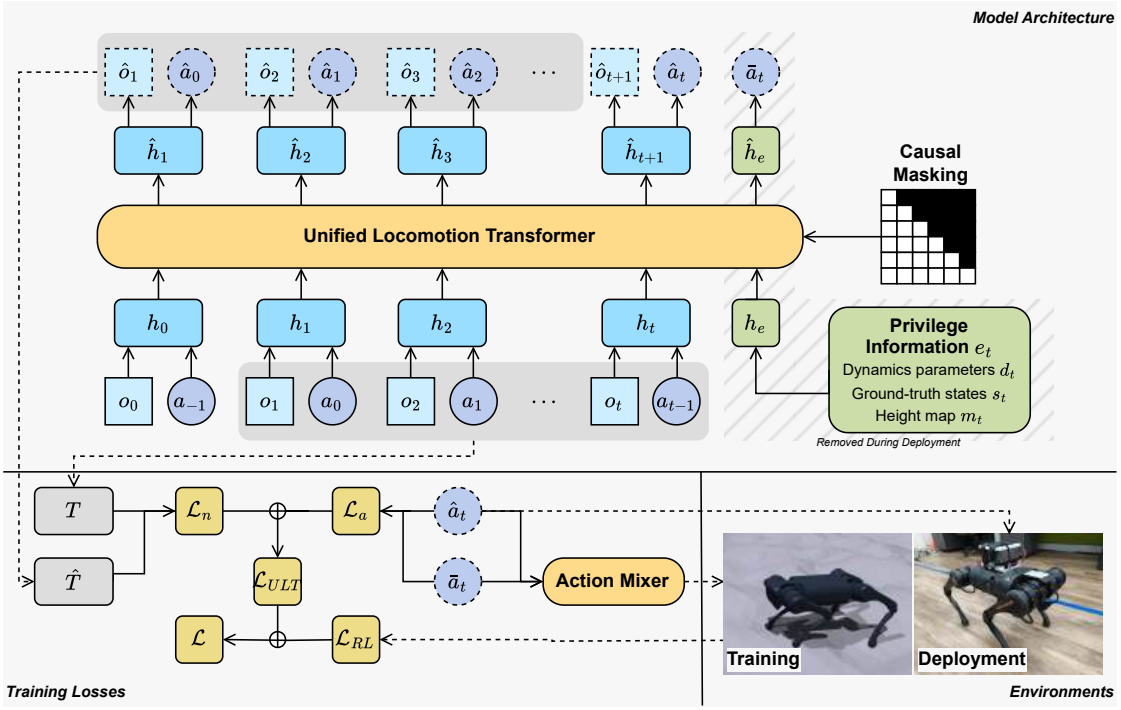


FIGURE 5.1: Illustration of the Unified Locomotion Transformer (ULT) framework. ULT is a vanilla transformer-based architecture to unify the optimization of locomotion policy and knowledge transfer. With state-action trajectories and privilege information in a single framework, both teacher and student actions can be generated simultaneously. The optimization is conducted jointly through PPO by combining the RL loss and transformer loss, which contains the next state-action prediction for future trajectories, and action imitation between student and teacher policies. During training in simulation, an action mixer is used to ensure both policies are played to enhance exploration. During the physical deployment, only proprioceptive observation is used for student actions to achieve zero-shot sim-to-real transfer.

RL-based quadrupedal locomotion is often formulated as a Partially Observable Markov Decision Process (POMDP), defined by a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, T, \mathcal{R}, \Omega, \mathcal{O}, \gamma)$, where $\mathcal{S}, \mathcal{A}, \mathcal{R}, \Omega$ are the state, action, reward and observation spaces, respectively. The ground truth states $s_t \in \mathcal{S}$ give the most important information about the environment and can be accessed by the teacher policy in the simulation to search for an optimal control policy $\pi^*(a_{t+1}|s_t)$ by maximizing the sum of discounted future rewards:

$$\pi^*(s,a) := \arg\max_{\pi} \mathbb{E}_{s_{t+1}\sim T(\cdot|s_t,a_t)} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right] \tag{5.2}$$

However, such information is often inaccessible in the real world, and the student policy can only rely on the noisy proprioceptive observation $o_t \in \Omega$ during deployment. Essentially, knowledge transfer aims to find the observation probabilities

$\mathcal{O} : \mathcal{S} \to \Omega$ such that the student policy can estimate its current status for decision making, either with latent space estimation [3] or direct action imitation [8, 9, 20].

Although many existing quadruped locomotion solutions have used the historical information $\mathcal{T}$ to capture the temporal information [3, 8, 9], they only treat it as a whole when predicting the next action $a_{t+1} = \pi(\mathcal{T})$ while missing the transition information $T(s_{t+1}|s_t, a_t)$ hidden in the sequence itself. With the transformer architecture and its next token prediction capability [10], we can extract and utilize information about the state-action transition from the history $\mathcal{T}$. By pairing it with policy imitation from the privilege information and mixed exploration, we can greatly improve the sample efficiency to discovery the state transition and observation probabilities, thus achieving the optimal teacher and student policies simultaneously in just one training phase.

5.3.1 Model Architecture

The foundational part of `ULT` is a vanilla transformer [42]. It contains multiple stacked multi-head attention blocks with causal masking, so the tokens can only attend to themselves and the past tokens while the proprioceptive tokens will not access the information from the privilege tokens.

Similarly to [10], we first tokenize the input trajectory $\mathcal{T}$ with a concatenated states action pair by a shared linear projection layer $W \in \mathbb{R}^{d\times(m+n)}$, where $m = 37$ and $n = 12$ are the dimensions of observation o and a at each step t. We choose $d = 128$ as the size of the token embedding:

$$z_t = \text{concat}(o_t, a_{t-1}) \tag{5.3}$$

$$h_t = W z_t \tag{5.4}$$

For privilege information $e_t = [d_t, s_t, m_t]$, we use an environmental factor encoder μ with a three-layer MLP similar to [3] to project it into the same embedding space:

$$h_e = \mu(e) \tag{5.5}$$

The transformer module takes the entire sequence of $H = [h_0, h_1, \cdots, h_t, h_e]$ with privilege information at the end to ensure that the information will not be leaked

to the proprioceptive observation. The sequence is then processed through all the attention layers:

$$\begin{aligned} \hat{H} &= \texttt{ULT}(H) \\ &= [\hat{h}_0, \hat{h}_1, \cdots, \hat{h}_t, \hat{h}_e] \end{aligned} \tag{5.6}$$

Next State-Action Prediction. To perform an aligned prediction, we extract the first $(t-1)$ tokens from the output $\hat{H}_{0:(t-1)} = [\hat{h}_0, \hat{h}_1, \cdots, \hat{h}_{t-1}]$, and decode them through another shared linear project $\hat{W} \in \mathbb{R}^{(m+n)\times d}$ to predict the future state action trajectory for each step:

$$\hat{z}_{t+1} = \hat{W}\hat{h}_t \tag{5.7}$$

$$\hat{T} = [\hat{z}_1, \hat{z}_2, \cdots, \hat{z}_t] \tag{5.8}$$

We can compare the next state-action pairs between the predicted trajectory $\hat{T}$ and actual trajectory $T = [z_1, z_2, \cdots, z_t]$:

$$\mathcal{L}_n = \frac{1}{t}\sum_{1}^{t} \|z_t - \hat{z}_t\|^2 \tag{5.9}$$

By optimizing $\mathcal{L}_n$, the transformer can always learn the transition relation of the robot state and action, regardless of the actions taken or the quality of the trajectories.

Action Output. `ULT` can simultaneously output actions from the teacher and student. For the student, with the next state-action prediction, we already implement a decoder layer to extract the predicted next state-action trajectory and it can already output the next action at each time step. Thus, we directly reuse the information in Eq. 5.7 to form the last concatenated state-action input token:

$$\hat{a}_t = (\hat{z}_{t+1})_{m:(m+n)} \tag{5.10}$$

For the teacher, with the process through all the attention layers, the resulted $\hat{h}_e$ (Eq. 5.6) have already gathered all the information from the state-action trajectory due to casual masking. In order to generate actions from $\hat{h}_e$, we implement a policy

TABLE 5.3: Hyperparameters for PPO

Parameters	Value
Number of GPUs	2
Actors per GPU	4096
Episode Length	20s
Horizon Length	24
Mini Epochs	5
Minibatch Size	16384
Learning Rate	3e-3
Scheduler	cosine
Optimizer	AdamW
Clip range	0.2
Entropy coefficient	0.005
Reward Discount	0.99
GAE Discount	0.95
Desired KL-divergence	0.008
Weight Decay	0.01

π with an MLP network similar to [3]:

$$\bar{a}_t = \pi(\hat{h}_e) \tag{5.11}$$

Thus, we can combine the imitation loss of the action and the next state-action prediction to get the overall performance with a weighting factor β:

$$\mathcal{L}_a = \|\bar{a}_t - \hat{a}_t\|^2 \tag{5.12}$$

$$\mathcal{L}_{\texttt{ULT}} = \mathcal{L}_n + \beta\mathcal{L}_a \tag{5.13}$$

5.3.2 Action Mixer and Unified Training

In order to achieve optimized performance of $\bar{a}_t$ and $\hat{a}_t$ simultaneously without a pre-trained policy, online trajectories generated by both actions are needed. In a massive parallelism training environment with X agents, an agent mask M is created with a threshold α as the mix ratio such that:

$$a_i = \begin{cases} \bar{a}_i, \text{if } M_i < \alpha \\ \hat{a}_i, \text{otherwise} \end{cases} \quad \text{where } M_i \sim \mathcal{U}(0,1), i = 1, \cdots, X \tag{5.14}$$

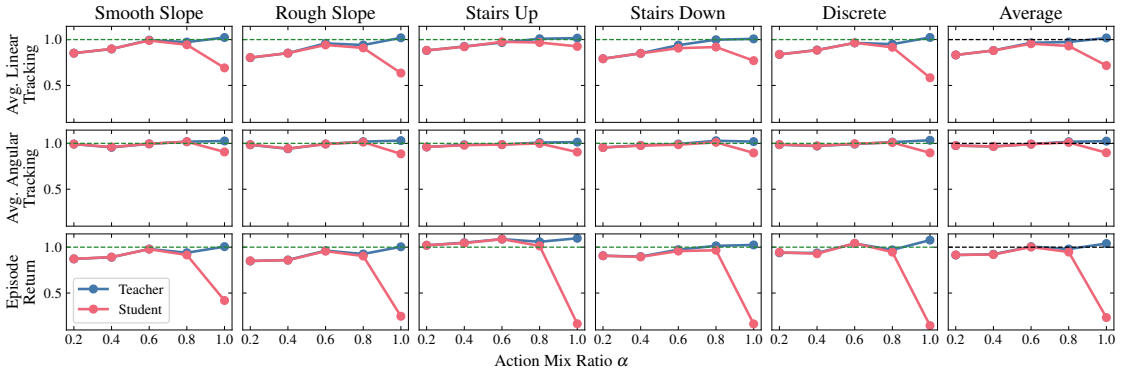


FIGURE 5.2: Performance of `ULT` with different values of Action Mixer ratio α on five terrains and the overall performance across all trails.

Thus, a higher mix ratio α means more involvement of the teacher. The agent mask M is frequently resampled to ensure exploration for both policies. With the trajectories generated by the resulted a, we use PPO [140] to jointly optimize the policy and action imitation by appending the PPO RL loss and transformer loss with the hyperparameters in Table 5.3:

$$\mathcal{L} = \mathcal{L}_{\text{RL}} + \lambda \mathcal{L}_{\texttt{ULT}} \tag{5.15}$$

Optimization of $\mathcal{L}$ can simultaneously leverage the state transition, action imitation, and guidance from the teacher policy based on the privilege information to achieve a unified training in a single phase.

5.3.3 Direct Sim-to-Real Deployment

The transformer architecture gives the flexibility of variable input length. During training, causal masking ensures that future information will not be seen by previous tokens and the privilege information will never be leaked to proprioceptive observation. Thus, when deployed in the real world, we can directly remove the last privileged environmental token h_e safely from the input sequence to generate the student action $\hat{a}_i$ directly with only the historical information $\mathcal{T}$.

TABLE 5.4: Normalized Performance in key metrics with different baselines on five terrain types and average across all trails.

Terrain	Metric	ULT (Ours)		Supervised Transfer			Joint Transfer	CENet	PPO
		Teacher	Student	Offline-Only	Online-Only	Two-Stages			
Smooth Slope	Avg. Linear Tracking	0.994	0.990	0.691	0.994	1.003	0.907	1.005	0.956
	Avg. Angular Tracking	0.995	0.995	0.851	1.006	1.002	0.958	1.020	0.977
	Total Episode Return	0.980	0.978	0.412	0.999	0.993	0.880	1.001	0.916
Rough Slope	Avg. Linear Tracking	0.959	0.941	0.660	0.985	0.990	0.882	0.950	0.926
	Avg. Angular Tracking	0.993	0.992	0.844	1.007	0.997	0.957	1.015	0.987
	Total Episode Return	0.962	0.957	0.432	0.977	0.967	0.876	0.965	0.911
Stairs Up	Avg. Linear Tracking	0.967	0.965	0.837	0.962	0.995	0.943	0.966	0.933
	Avg. Angular Tracking	0.987	0.985	0.844	1.000	0.973	0.965	0.982	0.982
	Total Episode Return	1.088	1.086	0.138	0.829	0.824	0.890	1.074	0.933
Stairs Down	Avg. Linear Tracking	0.939	0.906	0.705	0.948	0.953	0.884	0.872	0.890
	Avg. Angular Tracking	0.993	0.986	0.837	0.993	0.979	0.956	0.961	0.967
	Total Episode Return	0.973	0.957	0.125	0.836	0.879	0.923	0.929	0.877
Discrete	Avg. Linear Tracking	0.964	0.964	0.613	1.000	0.996	0.848	0.915	0.857
	Avg. Angular Tracking	0.992	0.997	0.805	1.012	1.003	0.944	0.983	0.957
	Total Episode Return	1.039	1.043	0.107	1.009	0.990	0.889	0.926	0.905
Average	Avg. Linear Tracking	0.965	0.953	0.698	0.978	0.987	0.892	0.942	0.912
	Avg. Angular Tracking	0.992	0.991	0.836	1.004	0.991	0.956	0.992	0.974
	Total Episode Return	1.006	1.001	0.249	0.932	0.933	0.892	0.978	0.908

5.4 Experiments and Results

We evaluate the effectiveness of `ULT` in simulated environments, mainly focusing on three metrics: the average linear and angular velocity tracking return per step for task-related performance and the final total episode reward return for the overall locomotion quality. All the reported results are averaged over 5000 trajectories collected across all terrain types and levels and normalized over the performance of a pretrained privilege Oracle policy adopted from [3] on respective terrain, which is also used as the common teacher for all baselines for a fair comparison.

5.4.1 Action Mixer Ratio

The first question we want to answer is: what is the optimal value for the Action Mixer introduced in Chapter 5.3.2? This mix ratio directly decides the proportion of the trajectories of the teacher and the student during the exploration and eventually affects the final performance due to the difference in their information density and the combined optimization objective with the next prediction and imitation of actin. To answer it, we train multiple `ULT` models with different values of α and keep all other configurations untouched. Figure 5.2 shows the key metrics for different terrains during testing.

We observe that the teacher policy suffers when the mix ratio is too low, as it cannot gather enough trajectories to reach an optimal policy, even with the help from all the privilege information. When the mix ratio is too high, the performance of the student starts to drop, as the training process is overly dependent on the guidance from the teacher policy and does not have enough exploration experience to handle out-of-the-distribution situations in complex environments.

In most cases, the knowledge is transferred efficiently, and the student policy can achieve similar performance as its respective teacher. One special case is $\alpha = 1$, where only the teacher trajectory is generated and used during training while the student is trained in a purely supervised manner. Although it produces one of the best performing teacher policies, the student acts poorly due to the lack of exploration, resulting in low survival rate. Another special case of $\alpha = 0$ will be

discussed in Chapter 5.4.3 as it has a fundamental difference due to the lack of teacher participation and optimization.

For the rest of this chapter, we will use the policies trained with a ratio of $\alpha = 0.6$ unless specified otherwise.

5.4.2 Comparison with Baselines

We compare `ULT` with several knowledge transfer solutions and their variants with a base teacher network similar to [9]:

- **Supervised Transfer.** Based on [3, 8, 9], we implemented different variants of supervised knowledge transfer with direct action imitation. **Offline-Only**: single stage with Oracle pre-trained policy used for trajectory generation [8]; **Online-Only**: single stage with the student generating online trajectories with [3]; **Two-Stages**: combining two stages with offline pre-training first, followed by online correction [9].
- **Joint Transfer.** Following [20], combining the RL exploration of the student with online supervised transfer with the joint ratio of policy imitation gradually annealed to zero by the mid-point of training.
- **CENet.** Implementation of an auto-encoder model from DreamWaQ [139] with VAE loss for the next observation, base velocity and latent space estimation with asymmetric actor-critic architecture.
- **PPO.** We use vanilla PPO [140] to train `ULT` solely on proprioceptive observation and RL loss, which is equivalent to an action mixer ratio of $\alpha = 0$, and with the teacher head and other loss modules disabled.

The comparison results are summarized in Table 5.4. Although `ULT` only uses the same number of trajectories as Oracle training, both the teacher and student policies achieve the similar performance level of Oracle performance and outperform other baseline models, which require many more trajectories to be generated for most cases. This shows the high efficiency of our proposed framework. With the increased difficulty of using omnidirectional control on all types of terrain, single-stage offline supervised transfer cannot efficiently capture the dynamics of the environment with

just good trajectories, and the rough terrains make it hard for the student to survive. Although online supervised transfer and two-stage transfer significantly improve performance, ULT still outperforms supervised transfer with fewer trajectories used. Joint transfer require manual tuning of the joint ratio and struggles to handle the increased environmental challenges with similar performance to vanilla PPO.

Learning with VAE loss shows strong performance, with only slight disadvantages compared to ULT, making it one of the best performing baselines. These results strongly support the shared idea of DreamWaQ and ULT, that understanding the state-action transition can significantly enhance policy optimization and overall performance.

These results demonstrate that with our ULT framework, optimal teacher and student policies can be achieved at the same time with a more compact unified network, without the need for multiple stages of knowledge transfer with complex network and loss function design. This greatly eases the training setup and the difficulty of knowledge transfer for quadruped locomotion for sim-to-real deployment.

5.4.3 Ablation Studies

To better evaluate the main components of ULT, we create different variants by removing the next state-action predictions, imitation of actions, and both from the unified optimization pipeline. The results are shown in Figure 5.3. With unified training, our core transformer benefits from both the next state-action prediction module and action imitation module, to increase its capability of understanding state-action transition with high-quality action guided by privilege information.

In addition, we explore the special case of the mix ratio $\alpha = 0$, which is equipment to remove the action mixer module. As the teacher head will not be optimized, there is also no need to imitate the action. It is clear that missing the privilege information makes it difficult to optimize the next state-action prediction resulted in a performance similar to vanilla PPO. ULT needs all modules to work together in order to achieve the sample and learning efficiency.

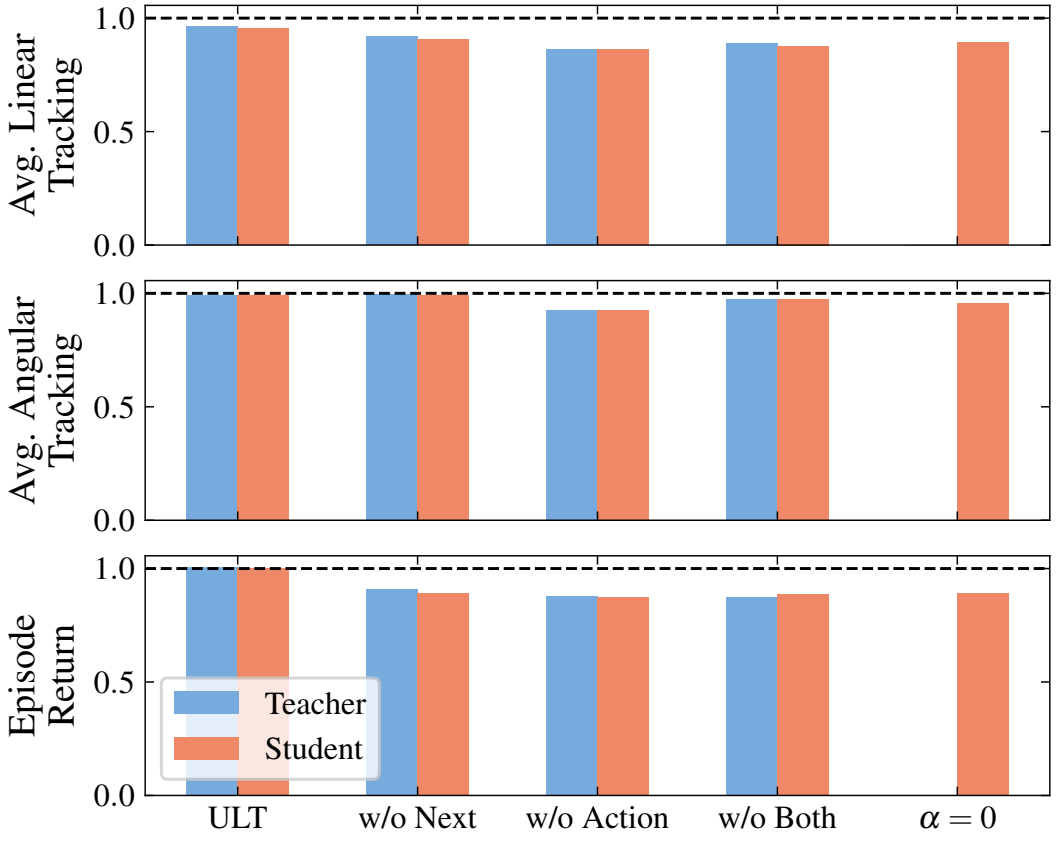


FIGURE 5.3: Normalized metrics for ULT and its ablated variants. The return is averaged across trails on all five terrains.

TABLE 5.5: Performance of $\alpha = 1$ student before and after supervised knowledge transfer.

	Avg. Linear Tracking	**Avg. Angular Tracking**	**Episode Return**
Original	0.717	0.898	0.233
After Online Transfer	0.760	0.893	0.865

5.4.4 ULT with Supervised Transfer

Training with proper mixed actions ensures that we can achieve optimal teacher and student policy at the same time, but a teacher-only policy still has slightly better performance. Can ULT act as the teacher first and then as the classic supervised knowledge transfer in a single network? Table 5.5 shows the average performance of the original student agent and after an online supervised transfer phase. Although we can recover some of the performance with additional transfer stage, it is still not comparable to ULT as we can hardly update the transformer while fixing the output

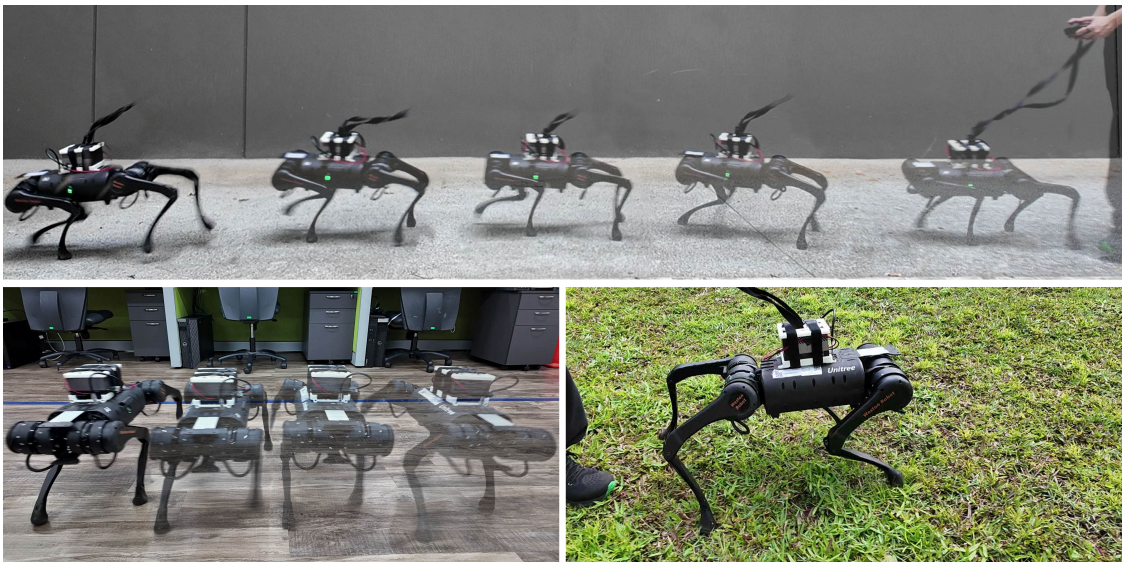


FIGURE 5.4: Footage snapshots showing a single trained ULT policy deployed in the real world with zero-shot transfer on Unitree A1 equipped with Jetson Orin AGX for inference on different terrains with motion in omnidirections.

of the teacher head, demonstrating the importance and efficiency from the action mixer while training the framework as a whole.

5.4.5 Physical Deployment

Following Sector 5.3.3, we directly extract the ULT framework and use onboard sensor observations to achieve zero-shot sim-to-real transfer. The policy is exported as JIT for portability and edge inference on a Unitree A1 robot equipped with a Jetson AGX Orin Developer Kit. The policy can run at up to 300Hz and we set the control frequency to 50Hz, $K_p = 30$ and $K_d = 0.7$ while communicating with A1's onboard low-level controller. Figure 5.4 shows some snapshots from the deployment test. Please refer to the supplementary video for more information.

5.5 Conclusion

We introduce ULT, a unified framework based on transformers for simultaneous optimization of teacher and student policies for quadruped locomotion. With next state-action prediction and action imitation, ULT can efficiently extract valuable transition information and provide guidance with privileged information for mixed

exploration to improve the training process. This greatly reduces the complexity and trajectory data needed for sim-to-real transfer, enabling the direct deployment of the agent on physical systems.

Chapter 6

Conclusion and Future Work

In this chapter, we summarize the key contributions of the thesis and outline potential directions for future research.

6.1 Conclusion

This thesis has addressed fundamental challenges in robust and generalized physical AI systems for quadrupedal locomotion through three primary contributions that collectively advance the state-of-the-art in learning-based robotic control. The presented methodologies demonstrate significant progress toward bridging the persistent gaps between simulation training and real-world deployment, while enhancing the robustness and generalizability of autonomous robotic systems.

The Masked Sensory-Temporal Attention (`MSTA`) mechanism presents a novel solution to the sensor generalization challenges that have long plagued robotic systems operating in various hardware configurations. By employing transformer-based attention mechanisms with strategic masking techniques, `MSTA` achieves remarkable robustness when facing incomplete or degraded sensory information, maintaining performance even with substantial sensor data loss. This capability is essential for deploying robotic systems in real-world environments where sensor reliability cannot be guaranteed.

The Fault-Tolerant Locomotion (`FTL`) framework introduced represents a crucial advancement in ensuring operational reliability for quadrupedal robots in unpredictable

environments. Through the teacher-student reinforcement learning paradigm with domain randomization, our approach successfully enables zero-shot transfer from simulation to physical platforms, demonstrating robust locomotion recovery under various hardware failure conditions. This contribution addresses a critical limitation in current robotic deployment scenarios where hardware malfunctions can lead to catastrophic system failures.

The Unified Locomotion Transformer (`ULT`) framework represents a significant step toward simplifying the complex multi-stage training processes traditionally required for effective sim-to-real transfer. By integrating privilege information processing and proprioceptive observation handling within a single transformer architecture, `ULT` achieves performance comparable to conventional approaches while substantially reducing computational complexity and training requirements.

The experimental validation across both simulation and real-world platforms, particularly the extensive testing on Unitree A1 quadrupedal robots, demonstrates the practical viability of these contributions. The zero-shot deployment capabilities achieved without fine-tuning represent a significant milestone toward realizing truly autonomous robotic systems capable of reliable operation in complex, dynamic environments.

6.2 Future Work

The foundations established in this thesis open several promising avenues for future research that can further advance the field of robust physical AI systems. A natural and important next step is to find ways to combine the three proposed frameworks with `MSTA` for sensor generalization, `FTL` for fault-tolerant locomotion, and `ULT` for unified knowledge transfer, so that we can efficiently train a generalized and robust controller. By utilizing sensor-level tokenization from `MSTA` and privilege information integration from `FTL` with a better masking strategy, the temporal-sensory information could directly work with privilege information within `ULT`'s unified optimization, and be trained in a more comprehensive environment with comprehensive failure scenarios. Beyond this integration, there are several domains that can be further explored.

- **Cross-Embodiment Learning with Embodiment Awareness.** While there has been some work in cross-embodiment learning, most approaches rely on gathered trajectories, which are expensive and time-consuming to collect. It also requires new data to be gathered for fine-tuning to new robots. This limitation stems from the lack of foundational understanding of each morphology and embodiment in current physical AI systems, such as how physical variations affect control strategies and performance outcomes, necessitating costly trajectory data collection processes that hinder scalability. The adoption of multimodal LLMs and VLA models presents unprecedented opportunities for addressing these limitations by potentially leveraging information about robot physical parameters. Future research should explore parsing design information, e.g., from Unified Robot Description Format (URDF) files, to extract critical morphological parameters and kinematic configurations, enabling automated tuning through calibration processes that could significantly enhance both cross-embodiment learning and sim-to-real transfer. The development of embodiment-aware representations will be crucial for achieving truly universal robotic policies capable of seamless adaptation across diverse configurations without requiring extensive trajectory data collection.

- **Advanced Simulation and World Foundation Model Development.** With the current development of simulators, robot learning has advanced significantly, enabling training and testing robotic policies in controlled virtual environments before deployment. However, the simulators are not perfect, which remains a significant blocker for effective sim-to-real transfer, as the reality gap continues to limit the direct deployment of learned policies. Current simulation-first approaches still require better simulators with enhanced rendering capabilities, more accurate physical engines to bridge this gap. The incorporation of differentiable physics engines enables end-to-end learning through gradient-based optimization to improve data efficiency in policy training. Furthermore, current simulation frameworks require extensive manual configuration of simulation parameters, asset creation, and environment modeling, especially for large-scale scenes, that limits scalability and accessibility. The integration of 3D Gaussian splatting for automatic scene generation presents a promising solution for reducing manual setup overhead while providing photorealistic scene reconstruction with physical properties like mesh generation. We can further leverage world foundation models to accelerate

both virtual scene generation and trajectory generation, with their capability in handling complex, long-horizon scenes. The combination of automated environment reconstruction techniques with world foundation models will enable efficient synthetic data generation and streamlined simulation setup, reducing both the reliance on expensive real-world trajectory collection and the manual labor required for virtual environment creation while improving sim-to-real transfer capabilities.

- **Integration of Reinforcement Learning and Imitation Learning.** Previously, reinforcement learning was primarily utilized for its autonomous exploration capability, enabling robots to discover optimal behaviors through trial-and-error interactions with the environment. Though reinforcement learning represents a theoretically sound approach for learning complex behaviors, it is well-known for its data inefficiency, requiring extensive exploration and numerous interactions to achieve satisfactory performance levels. With the adoption of large-scale transformers and foundation models, current methods increasingly rely on imitation learning to learn from demonstrations to accelerate training processes, while reinforcement learning is added mainly to handle the sim-to-real gap with a primary focus on stability and robustness. However, imitation learning faces significant limitations in learning new functions beyond the demonstrated behaviors, requiring new data collection for each novel task or capability, which restricts its adaptability and generalization potential. Future research should explore better methodologies to integrate reinforcement learning and imitation learning, where reinforcement learning learns foundational tasks and motor primitives, while imitation learning adds high-level task-specific behaviors and complex coordination patterns. With a well established fundamental capability in locomotion, manipulation and environmental interaction skills, new task-specific behaviors can be built more efficiently to achieve a more generalized agent.

- **Efficient On-Board Model Deployment.** The practical deployment of advanced AI models on resource-constrained robotic platforms requires significant advances in model compression and edge computing techniques. Future research should prioritize the development of small language models and compressed neural networks specifically optimized for robotic applications while maintaining high performance. Key areas for investigation include parameter-efficient fine-tuning techniques that enable rapid adaptation to new

environments without requiring full model retraining. The development of quantization and pruning strategies tailored for robotic control tasks will be essential for enabling deployment of complex models on embedded platforms with limited computational resources. Furthermore, research should explore dynamic neural networks that can adapt their computational complexity based on available resources and task requirements. The integration of edge AI capabilities with advanced cross-embodiment agent will require careful balance between model capability and computational efficiency. Future work should investigate hardware-software co-design approaches that optimize both model architecture and deployment platform for maximum performance under strict resource constraints.

List of Publications

- **Dikai Liu**, Tianwei Zhang, Jianxiong Yin, and Simon See. **Unified Locomotion Transformer with Simultaneous Sim-to-Real Transfer for Quadrupeds**. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2025.

- **Dikai Liu**, Tianwei Zhang, Jianxiong Yin, and Simon See. **Masked Sensory-Temporal Attention for Sensor Generalization in Quadruped Locomotion**. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

- **Dikai Liu**, Jianxiong Yin, and Simon See. **Towards Fault-tolerant Quadruped Locomotion with Reinforcement Learning**. In *IEEE Conference on Artificial Intelligence (CAI)*, 2024. (Honorable Mention for Best Paper)

- Zhehua Zhou, Jiayang Song, Xuan Xie, Zhan Shu, Lei Ma, **Dikai Liu**, Jianxiong Yin, and Simon See. **Towards Building AI-CPS with NVIDIA Isaac Sim: An Industrial Benchmark and Case Study for Robotics Manipulation**. In *Proceedings of the 46th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, 2024.

Bibliography

[1] Erik D. Demaine and Martin L. Demaine. “Every Author as First Author”. In: *arXiv preprint arXiv:2304.01393* (2023) ix

[2] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. “Isaac Gym: High Performance GPU Based Physics Simulation for Robot Learning”. In: *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, Neurips Datasets and Benchmarks 2021, December 2021, Virtual*. 2021 xxii, 2, 13, 18, 33, 51–53, 59, 74

[3] Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. “RMA: Rapid Motor Adaptation for Legged Robots”. In: *Robotics: Science and Systems XVII*. 2021 xxii, 1–4, 18, 20–22, 30, 35, 36, 41, 45, 50, 51, 53, 56–61, 72, 75, 76, 78, 80, 83, 84

[4] Gabriel B. Margolis, Ge Yang, Kartik Paigwar, Tao Chen, and Pulkit Agrawal. “Rapid Locomotion via Reinforcement Learning”. In: *International Journal of Robotics Research* 43.4 (2024), pp. 572–587 xxii, 2, 18, 21, 22, 50, 51, 53, 56–58

[5] Zhiyuan Xu, Kun Wu, Junjie Wen, Jinming Li, Ning Liu, Zhengping Che, and Jian Tang. “A Survey on Robotics with Foundation Models: Toward Embodied AI”. In: *arXiv preprint arXiv:2402.02385* (2024) 1, 3

[6] Yang Liu, Weixing Chen, Yongjie Bai, Xiaodan Liang, Guanbin Li, Wen Gao, and Liang Lin. “Aligning Cyber Space with Physical World: A Comprehensive Survey on Embodied AI”. In: *arXiv preprint arXiv:2407.06886* (2025) 1

[7] Jie Tan, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. “Sim-to-Real: Learning Agile Locomotion for Quadruped Robots”. In: *Robotics: Science and Systems XIV*. 2018 1–3, 20, 22, 30, 72

[8] Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. “Learning Quadrupedal Locomotion over Challenging Terrain”. In: *Science Robotics* 5.47 (2020), eabc5986 1–4, 15, 20–22, 30, 47, 50, 51, 56–59, 72, 75, 76, 78, 84

[9] Hang Lai, Weinan Zhang, Xialin He, Chen Yu, Zheng Tian, Yong Yu, and Jun Wang. "Sim-to-Real Transfer for Quadrupedal Locomotion via Terrain Transformer". In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. 2023, pp. 5141–5147 1–4, 21, 22, 31–33, 37, 39, 41, 45, 47, 58, 62, 72, 73, 76, 78, 84

[10] Ilija Radosavovic, Bike Zhang, Baifeng Shi, Jathushan Rajasegaran, Sarthak Kamat, Trevor Darrell, Koushil Sreenath, and Jitendra Malik. "Humanoid Locomotion as next Token Prediction". In: *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, Neurips 2024, Vancouver, BC, Canada, December 10 - 15, 2024*. 2024 1, 31, 32, 37, 41, 72, 73, 78

[11] Nvidia, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. "GR00T N1: An Open Foundation Model for Generalist Humanoid Robots". In: *arXiv preprint arXiv:2503.14734* (2025) 1, 3, 4, 13, 26, 28

[12] *Helix: A Vision-Language-Action Model for Generalist Humanoid Control*. FigureAI. 2025. URL: https://www.figure.ai/news/helix 1, 28

[13] Carlos E Garcia, David M Prett, and Manfred Morari. "Model Predictive Control: Theory and Practice—a Survey". In: *Automatica* 25.3 (1989), pp. 335–348 2

[14] Jared Di Carlo, Patrick M. Wensing, Benjamin Katz, Gerardo Bledt, and Sangbae Kim. "Dynamic Locomotion in the MIT Cheetah 3 through Convex Model-Predictive Control". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2018, pp. 1–9 2, 50

[15] Michael Neunert, Markus Stauble, Markus Giftthaler, Carmine D. Bellicoso, Jan Carius, Christian Gehring, Marco Hutter, and Jonas Buchli. "Whole-Body Nonlinear Model Predictive Control through Contacts for Quadrupeds". In: *IEEE Robotics and Automation Letters* 3.3 (2018), pp. 1458–1465 2

[16] Philip Wolfe. "The Simplex Method for Quadratic Programming". In: *Econometrica: Journal of the Econometric Society* (1959), pp. 382–398 2

[17] Bartolomeo Stellato, Goran Banjac, Paul Goulart, Alberto Bemporad, and Stephen Boyd. "OSQP: An Operator Splitting Solver for Quadratic Programs". In: *Mathematical Programming Computation* 12.4 (2020), pp. 637–672 2

[18] Antoine Bambade, Sarah El-Kazdadi, Adrien Taylor, and Justin Carpentier. "Prox-Qp: Yet Another Quadratic Programming Solver for Robotics and Beyond". In: *RSS 2022-Robotics: Science and Systems*. 2022 2

[19] Yunlong Song, Angel Romero, Matthias Müller, Vladlen Koltun, and Davide Scaramuzza. "Reaching the Limit in Autonomous Racing: Optimal Control versus Reinforcement Learning". In: *Science Robotics* 8.82 (2023), eadg1462 2

[20] Ilija Radosavovic, Tete Xiao, Bike Zhang, Trevor Darrell, Jitendra Malik, and Koushil Sreenath. "Real-World Humanoid Locomotion with Reinforcement Learning". In: *Science Robotics* 9.89 (2024), eadi9579 2, 21, 22, 31, 32, 37, 47, 59, 72, 73, 78, 84

[21] Emanuel Todorov, Tom Erez, and Yuval Tassa. "MuJoCo: A Physics Engine for Model-Based Control". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2012, pp. 5026–5033 2, 13, 19

[22] Genesis Authors. *Genesis: A Universal and Generative Physics Engine for Robotics and Beyond*. 2024. URL: https://github.com/Genesis-Embodied-AI/Genesis 2

[23] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. "Playing Atari with Deep Reinforcement Learning". In: *arXiv preprint arXiv:1312.5602* (2013) 2

[24] Nikita Rudin, David Hoeller, Philipp Reist, and Marco Hutter. "Learning to Walk in Minutes Using Massively Parallel Deep Reinforcement Learning". In: *Conference on Robot Learning, 8-11 November 2021, London, UK*. Vol. 164. 2021, pp. 91–100 2, 18–20, 33–35, 56, 59, 60, 69, 74, 76

[25] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. "Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2017, pp. 23–30 3, 51

[26] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. "Sim-to-Real Transfer of Robotic Control with Dynamics Randomization". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 2018, pp. 3803–3810 3, 51

[27] Jemin Hwangbo, Joonho Lee, Alexey Dosovitskiy, Dario Bellicoso, Vassilios Tsounis, Vladlen Koltun, and Marco Hutter. "Learning Agile and Dynamic Motor Skills for Legged Robots". In: *Science Robotics* 4.26 (2019), eaau5872 3, 22, 30, 51, 72

[28] Zhehua Zhou, Jiayang Song, Xuan Xie, Zhan Shu, Lei Ma, Dikai Liu, Jianxiong Yin, and Simon See. "Towards Building AI-CPS with NVIDIA Isaac Sim: An Industrial Benchmark and Case Study for Robotics Manipulation". In: *Proceedings of the 46th International Conference on Software Engineering: Software Engineering in Practice*. 2024, pp. 263–274 3, 19

[29] Yevgen Chebotar, Ankur Handa, Viktor Makoviychuk, Miles Macklin, Jan Issac, Nathan Ratliff, and Dieter Fox. "Closing the Sim-to-Real Loop: Adapting Simulation Randomization with Real World Experience". In: *2019 International Conference on Robotics and Automation (ICRA)*. 2019, pp. 8973–8979 3, 21

[30] Konstantinos Bousmalis, Alex Irpan, Paul Wohlhart, Yunfei Bai, Matthew Kelcey, Mrinal Kalakrishnan, Laura Downs, Julian Ibarz, Peter Pastor, Kurt Konolige, Sergey Levine, and Vincent Vanhoucke. "Using Simulation and Domain Adaptation to Improve Efficiency of Deep Robotic Grasping". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 2018, pp. 4243–4250 3

[31] Kuan Fang, Yunfei Bai, Stefan Hinterstoisser, Silvio Savarese, and Mrinal Kalakrishnan. "Multi-Task Domain Adaptation for Deep Learning of Instance Grasping from Simulation". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 2018, pp. 3516–3523 3

[32] Xiaoning Du, Xiaofei Xie, Yi Li, Lei Ma, Yang Liu, and Jianjun Zhao. "DeepStellar: Model-Based Quantitative Analysis of Stateful Deep Learning Systems". In: *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 2019, pp. 477–487 3

[33] Xiaofei Xie, Lei Ma, Felix Juefei-Xu, Minhui Xue, Hongxu Chen, Yang Liu, Jianjun Zhao, Bo Li, Jianxiong Yin, and Simon See. "DeepHunter: A Coverage-Guided Fuzz Testing Framework for Deep Neural Networks". In: *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis*. 2019, pp. 146–157 3

[34] Junwen Cui, Zhan Li, Jing Qiu, and Tianxiao Li. "Fault-Tolerant Motion Planning and Generation of Quadruped Robots Synthesised by Posture Optimization and Whole Body Control". In: *Complex & Intelligent Systems* 8.4 (2022), pp. 2991–3003 3, 24, 50, 53

[35] Nvidia, Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, Daniel Dworakowski, Jiaojiao Fan, Michele Fenzi, Francesco Ferroni, Sanja Fidler, Dieter Fox, Songwei Ge, Yunhao Ge, Jinwei Gu, Siddharth Gururani, Ethan He, Jiahui Huang, Jacob Huffman, Pooya Jannaty, Jingyi Jin, Seung Wook Kim, Gergely Klár, Grace Lam, Shiyi Lan, Laura Leal-Taixe, Anqi Li, Zhaoshuo Li, Chen-Hsuan Lin, Tsung-Yi Lin, Huan Ling, Ming-Yu Liu, Xian Liu, Alice Luo, Qianli Ma, Hanzi Mao, Kaichun Mo, Arsalan Mousavian, Seungjun Nah, Sriharsha Niverty, David Page, Despoina Paschalidou, Zeeshan Patel, Lindsey Pavao, Morteza Ramezanali, Fitsum Reda, Xiaowei Ren, Vasanth Rao Naik Sabavat, Ed Schmerling, Stella Shi, Bartosz Stefaniak, Shitao Tang, Lyne Tchapmi, Przemek Tredak, Wei-Cheng Tseng, Jibin Varghese, Hao Wang, Haoxiang Wang, Heng Wang, Ting-Chun Wang, Fangyin Wei, Xinyue Wei, Jay Zhangjie Wu, Jiashu Xu, Wei Yang, Lin Yen-Chen, Xiaohui Zeng, Yu Zeng, Jing Zhang, Qinsheng Zhang, Yuxuan Zhang, Qingqing Zhao, and Artur Zolkowski. "Cosmos World Foundation Model Platform for Physical AI". In: *arXiv preprint arXiv:2501.03575* (2025) 3

[36] Open X.-Embodiment Collaboration et al. "Open X-embodiment: Robotic Learning Datasets and RT-X Models". In: *arXiv preprint arXiv:2310.08864* (2024) 3, 4, 13, 26, 27, 31, 73

[37] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. "π_0: A Vision-Language-Action Flow Model for General Robot Control". In: *arXiv preprint arXiv:2410.24164* (2024) 3, 4, 13, 26, 27

[38] Robert Kirk, Amy Zhang, Edward Grefenstette, and Tim Rocktäschel. "A Survey of Zero-Shot Generalisation in Deep Reinforcement Learning". In: *Journal of Artificial Intelligence Research* 76 (2023), pp. 201–264 4, 14, 19, 20

[39] Gilbert Feng, Hongbo Zhang, Zhongyu Li, Xue Bin Peng, Bhuvan Basireddy, Linzhu Yue, Zhitao Song, Lizhi Yang, Yunhui Liu, Koushil Sreenath, and Sergey Levine. "GenLoco: Generalized Locomotion Controllers for Quadrupedal Robots". In: *Conference on Robot Learning, Corl 2022, 14-18 December 2022, Auckland, New Zealand*. Vol. 205. 2022, pp. 1893–1903 4, 24, 30

[40] Nico Bohlinger, Grzegorz Czechmanowski, Maciej Piotr Krupka, Piotr Kicki, Krzysztof Walas, Jan Peters, and Davide Tateo. "One Policy to Run Them All: An End-to-End Learning Approach to Multi-Embodiment Locomotion". In: *Conference on Robot Learning, 6-9 November 2024, Munich, Germany*. Vol. 270. 2024, pp. 3356–3378 4, 24, 25

[41] Ria Doshi, Homer Rich Walke, Oier Mees, Sudeep Dasari, and Sergey Levine. "Scaling Cross-Embodied Learning: One Policy for Manipulation, Navigation, Locomotion and Aviation". In: *Conference on Robot Learning, 6-9 November 2024, Munich, Germany*. Vol. 270. 2024, pp. 496–512 4, 24

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. "Attention Is All You Need". In: *arXiv preprint arXiv:1706.03762* (2023) 4, 15, 25, 26, 31, 72, 78

[43] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. "RT-1: Robotics Transformer for Real-World Control at Scale". In: *Robotics: Science and Systems XIX, Daegu, Republic of Korea, July 10-14, 2023*. 2023 4, 13, 15, 26, 27, 31, 37, 73

[44] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North*. 2019, pp. 4171–4186 4, 26, 31

[45] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. "Language Models Are Few-Shot Learners". In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, Neurips 2020, December 6-12, 2020, Virtual*. 2020 4, 26, 31

[46] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. "An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. 2021 4, 26, 31, 38, 46

[47] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. "Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 9992–10002 4, 26, 31

[48] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollar, and Ross Girshick. "Masked Autoencoders Are Scalable Vision Learners". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 15979–15988 4, 26, 31, 38

[49] Peng Xu, Xiatian Zhu, and David A. Clifton. "Multimodal Learning with Transformers: A Survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.10 (2023), pp. 12113–12132 4

[50] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. "Zero-Shot Text-to-Image Generation". In: *arXiv preprint arXiv:2102.12092* (2021) 4

[51] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. "Conformer: Convolution-Augmented Transformer for Speech Recognition". In: *Interspeech 2020*. 2020, pp. 5036–5040 4

[52] OpenAI et al. "GPT-4 Technical Report". In: *arXiv preprint arXiv:2303.08774* (2024) 4, 26, 31

[53] Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. "A Generalist Agent". In: *Trans. Mach. Learn. Res.* (2022) 4, 19, 24, 31, 37, 73

[54] Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong T. Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. "RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control". In: *Conference on Robot Learning, Corl 2023, 6-9 November 2023, Atlanta, GA, USA*. Vol. 229. 2023, pp. 2165–2183 4, 13, 26, 27, 31, 37, 73

[55] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Paul Foster, Pannag R. Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. "OpenVLA: An Open-Source Vision-Language-Action Model". In: *Conference on Robot Learning, 6-9 November 2024, Munich, Germany*. Vol. 270. 2024, pp. 2679–2713 4, 13, 26, 27

[56] Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Remi Cadene. "SmolVLA: A Vision-Language-Action Model for Affordable and Efficient Robotics". In: *arXiv preprint arXiv:2506.01844* (2025) 4, 13, 26

[57] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. "$\pi_{0.5}$: A Vision-Language-Action Model with Open-World Generalization". In: *arXiv preprint arXiv:2504.16054* (2025) 4, 13, 26

[58] Stephane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. "A Reduction of Imitation Learning and Structured Prediction to No-regret Online Learning". In: *arXiv preprint arXiv:1011.0686* (2010) 5, 21, 58, 72

[59] Eric Krotkov, Douglas Hackett, Larry Jackel, Michael Perschbacher, James Pippine, Jesse Strauss, Gill Pratt, and Christopher Orlowski. "The DARPA Robotics Challenge Finals: Results and Perspectives". In: *Journal of Field Robotics* 34.2 (2017), pp. 229–240 13, 72

[60] C. Dario Bellicoso, Marko Bjelonic, Lorenz Wellhausen, Kai Holtmann, Fabian Günther, Marco Tranzatto, Péter Fankhauser, and Marco Hutter. "Advances in Real-World Applications for Legged Robots". In: *Journal of Field Robotics* 35.8 (2018), pp. 1311–1326 13, 22, 50, 72

[61] Zhiming Chen, Tingxiang Fan, Xuan Zhao, Jing Liang, Cong Shen, Hua Chen, Dinesh Manocha, Jia Pan, and Wei Zhang. "Autonomous Social Distancing in Urban Environments Using a Quadruped Robot". In: *IEEE Access* 9 (2021), pp. 8392–8403 13, 50, 72

[62] Max Asselmeier, Jane Ivanova, Ziyi Zhou, Patricio A. Vela, and Ye Zhao. "Hierarchical Experience-Informed Navigation for Multi-Modal Quadrupedal Rebar Grid Traversal". In: *2024 IEEE International Conference on Robotics and Automation (ICRA)*. 2024, pp. 8065–8072 13

[63] Joshua Hooks, Min Sung Ahn, Jeffrey Yu, Xiaoguang Zhang, Taoyuanmin Zhu, Hosik Chae, and Dennis Hong. "ALPHRED: A Multi-Modal Operations Quadruped Robot for Package Delivery Applications". In: *IEEE Robotics and Automation Letters* 5.4 (2020), pp. 5409–5416 13, 50, 72

[64] Alexander Herzog, Kanishka Rao, Karol Hausman, Yao Lu, Paul Wohlhart, Mengyuan Yan, Jessica Lin, Montserrat Gonzalez Arenas, Ted Xiao, Daniel Kappler, Daniel Ho, Jarek Rettinghouse, Yevgen Chebotar, Kuang-Huei Lee, Keerthana Gopalakrishnan, Ryan Julian, Adrian Li, Chuyuan Fu, Bob Wei, Sangeetha Ramesh, Khem Holden, Kim Kleiven, David J. Rendleman, Sean Kirmani, Jeffrey Bingham, Jonathan Weisz, Ying Xu, Wenlong Lu, Matthew Bennice, Cody Fong, David Do, Jessica Lam, Yunfei Bai, Benjie Holson, Michael Quinlan, Noah Brown, Mrinal Kalakrishnan, Julian Ibarz, Peter Pastor, and Sergey Levine. "Deep RL at Scale: Sorting Waste in Office Buildings with a Fleet of Mobile Manipulators". In: *Robotics: Science and Systems XIX, Daegu, Republic of Korea, July 10-14, 2023*. 2023 13

[65] *Newton-Physics/Newton*. newton-physics, 2025. URL: https://github.com/newton-physics/newton 13

[66] Unitree Robotics. *Unitree A1*. URL: https://www.unitree.com/a1 13, 30

[67] Benjamin Katz, Jared Di Carlo, and Sangbae Kim. "Mini Cheetah: A Platform for Pushing the Limits of Dynamic Quadruped Control". In: *2019 International Conference on Robotics and Automation (ICRA)*. 2019, pp. 6295–6301 13, 30

[68] Marco Hutter, Christian Gehring, Dominic Jud, Andreas Lauber, C. Dario Bellicoso, Vassilios Tsounis, Jemin Hwangbo, Karen Bodie, Peter Fankhauser, Michael Bloesch, Remo Diethelm, Samuel Bachmann, Amir Melzer, and Mark Hoepflinger. "ANYmal - a Highly Mobile and Dynamic Quadrupedal Robot". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2016, pp. 38–44 13, 30

[69] Boston Dynamics. *Spot*. Spot® - the Agile Mobile Robot. URL: https://bostondynamics.com/products/spot/ 13, 30

[70] Unitree Robotics. *Unitree H1 / H1-2 Unitree's First Universal Humanoid Robot*. URL: https://www.unitree.com/h1 13

[71] Fourier Robotics. *Fourier GR-1 Your Ideal Robot Assistant For Everything*. URL: https://www.fftai.com/products-gr1 13

[72] Boston Dynamics. *An Electric New Era for Atlas*. URL: https://bostondynamics.com/blog/electric-new-era-for-atlas/ 13

[73] Mikko Lauri, David Hsu, and Joni Pajarinen. "Partially Observable Markov Decision Processes in Robotics: A Survey". In: *IEEE Transactions on Robotics* 39.1 (2023), pp. 21–40 14

[74] Marco Camurri, Milad Ramezani, Simona Nobili, and Maurice Fallon. "Pronto: A Multi-Sensor State Estimator for Legged Robots in Real-World Scenarios". In: *Frontiers in Robotics and AI* 7.68 (2020), pp. 1–18 14

[75] William Talbot, Julian Nubert, Turcan Tuna, Cesar Cadena, Frederike Dümbgen, Jesus Tordesillas, Timothy D. Barfoot, and Marco Hutter. "Continuous-Time State Estimation Methods in Robotics: A Survey". In: *arXiv preprint arXiv:2411.03951* (2024) 14

[76] Liuping Wang et al. *Model Predictive Control System Design and Implementation Using MATLAB*. Vol. 3. 2009 15

[77] Farbod Farshidian et al. *OCS2: An Open Source Library for Optimal Control of Switched Systems* 15, 69

[78] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling". In: *arXiv preprint arXiv:1803.01271* (2018) 15, 41, 59

[79] Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. "Temporal Convolutional Networks for Action Segmentation and Detection". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 156–165 15

[80] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling". In: *arXiv preprint arXiv:1412.3555* (2014) 15, 26, 31, 41

[81] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9.8 (1997), pp. 1735–1780 15, 26, 31

[82] Yu Xiang and Dieter Fox. "DA-RNN: Semantic Mapping with Data Associated Recurrent Neural Networks". In: *Robotics: Science and Systems XIII, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, July 12-16, 2017*. 2017 15

[83] Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy Lillicrap, and Martin Riedmiller. "DeepMind Control Suite". In: *arXiv preprint arXiv:1801.00690* (2018) 15, 57

[84] Michael Laskin, Aravind Srinivas, and Pieter Abbeel. "CURL: Contrastive Unsupervised Representations for Reinforcement Learning". In: *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*. Vol. 119. 2020, pp. 5639–5650 15, 57

[85] Haoran He, Peilin Wu, Chenjia Bai, Hang Lai, Lingxiao Wang, Ling Pan, Xiaolin Hu, and Weinan Zhang. "Bridging the Sim-to-Real Gap from the Information Bottleneck Perspective". In: *Conference on Robot Learning, 6-9 November 2024, Munich, Germany*. Vol. 270. 2024, pp. 473–495 15

[86] Richard S Sutton, Andrew G Barto, et al. *Reinforcement Learning: An Introduction*. Vol. 1. 1. 1998 16

[87] John Schulman, Sergey Levine, Pieter Abbeel, Michael I. Jordan, and Philipp Moritz. "Trust Region Policy Optimization". In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*. Vol. 37. 2015, pp. 1889–1897 17

[88] OpenAI, Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, Jonas Schneider, Nikolas Tezak, Jerry Tworek, Peter Welinder, Lilian Weng, Qiming Yuan, Wojciech Zaremba, and Lei Zhang. "Solving Rubik's Cube with a Robot Hand". In: *arXiv preprint arXiv:1910.07113* (2019) 18

[89] Jack Collins, Shelvin Chand, Anthony Vanderkop, and David Howard. "A Review of Physics Simulators for Robotic Applications". In: *IEEE Access* 9 (2021), pp. 51416–51431 18, 19

[90] N. Koenig and A. Howard. "Design and Use Paradigms for Gazebo, an Open-Source Multi-Robot Simulator". In: *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vol. 3. 2004, 2149–2154 vol.3 19

[91] Erwin Coumans and Yunfei Bai. *PyBullet, a Python Module for Physics Simulation for Games, Robotics and Machine Learning*. 2016 19

[92] Jemin Hwangbo, Joonho Lee, and Marco Hutter. "Per-Contact Iteration Method for Solving Contact Dynamics". In: *IEEE Robotics and Automation Letters* 3.2 (2018), pp. 895–902 19

[93] Haoran Geng, Feishi Wang, Songlin Wei, Yuyang Li, Bangjun Wang, Boshi An, Charlie Tianyue Cheng, Haozhe Lou, Peihao Li, Yen-Jen Wang, Yutong Liang, Dylan Goetting, Chaoyi Xu, Haozhe Chen, Yuxi Qian, Yiran Geng, Jiageng Mao, Weikang Wan, Mingtong Zhang, Jiangran Lyu, Siheng Zhao, Jiazhao Zhang, Jialiang Zhang, Chengyang Zhao, Haoran Lu, Yufei Ding, Ran Gong, Yuran Wang, Yuxuan Kuang, Ruihai Wu, Baoxiong Jia, Carlo Sferrazza, Hao Dong, Siyuan Huang, Yue Wang, Jitendra Malik, and Pieter Abbeel. "RoboVerse: Towards a Unified Platform, Dataset and Benchmark for Scalable and Generalizable Robot Learning". In: *arXiv preprint arXiv:2504.18904* (2025) 19

[94] Karl Cobbe, Christopher Hesse, Jacob Hilton, and John Schulman. "Leveraging Procedural Generation to Benchmark Reinforcement Learning". In: *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*. Vol. 119. 2020, pp. 2048–2056 20

[95] Antonio Loquercio, Elia Kaufmann, Rene Ranftl, Alexey Dosovitskiy, Vladlen Koltun, and Davide Scaramuzza. "Deep Drone Racing: From Simulation to Reality with Domain Randomization". In: *IEEE Transactions on Robotics* 36.1 (2020), pp. 1–14 20

[96] OpenAI, Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Józefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, Jonas Schneider, Szymon Sidor, Josh Tobin, Peter Welinder, Lilian Weng, and Wojciech Zaremba. "Learning Dexterous In-Hand Manipulation". In: *International Journal of Robotics Research* 39.1 (2020), pp. 3–20 20

[97] Jingru Luo and Kris Hauser. "Robust Trajectory Optimization under Frictional Contact with Iterative Learning". In: *Autonomous Robots* 41.6 (2017), pp. 1447–1461 20, 69

[98] Bowen Baker, Ingmar Kanitscheider, Todor Markov, Yi Wu, Glenn Powell, Bob McGrew, and Igor Mordatch. "Emergent Tool Use from Multi-Agent Autocurricula". In: *International Conference on Learning Representations*. 2019 20

[99] Bhairav Mehta, Manfred Diaz, Florian Golemo, Christopher J Pal, and Liam Paull. "Active Domain Randomization". In: *Conference on Robot Learning*. 2020, pp. 1162–1176 20

[100] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. "Distilling the Knowledge in a Neural Network". In: *arXiv preprint arXiv:1503.02531* (2015) 21

[101] Gabriel B. Margolis, Tao Chen, Kartik Paigwar, Xiang Fu, Donghyun Kim, Sangbae Kim, and Pulkit Agrawal. "Learning to Jump from Pixels". In: *Conference on Robot Learning, 8-11 November 2021, London, UK*. Vol. 164. 2021, pp. 1025–1034 21

[102] Takahiro Miki, Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. "Learning Robust Perceptive Locomotion for Quadrupedal Robots in the Wild". In: *Science Robotics* 7.62 (2022), eabk2822 21, 50

[103] Ziwen Zhuang, Zipeng Fu, Jianren Wang, Christopher G. Atkeson, Sören Schwertfeger, Chelsea Finn, and Hang Zhao. "Robot Parkour Learning". In: *Conference on Robot Learning, Corl 2023, 6-9 November 2023, Atlanta, GA, USA*. Vol. 229. 2023, pp. 73–92 23

[104] David Hoeller, Nikita Rudin, Dhionis Sako, and Marco Hutter. "ANYmal Parkour: Learning Agile Navigation for Quadrupedal Robots". In: *arXiv preprint arXiv:2306.14874* (2023) 23

[105] Xuxin Cheng, Kexin Shi, Ananye Agarwal, and Deepak Pathak. "Extreme Parkour with Legged Robots". In: *2024 IEEE International Conference on Robotics and Automation (ICRA)*. 2024, pp. 11443–11450 23

[106] Ziwen Zhuang, Shenzhe Yao, and Hang Zhao. "Humanoid Parkour Learning". In: *Conference on Robot Learning, 6-9 November 2024, Munich, Germany*. Vol. 270. 2024, pp. 1975–1991 23

[107] Yuhai Zhong, Runxiao Wang, Huashan Feng, and Yasheng Chen. "Analysis and Research of Quadruped Robot's Legs: A Comprehensive Review". In: *International Journal of Advanced Robotic Systems* 16.3 (2019), p. 1729881419844148 23

[108] Jung-Min Yang. "Kinematic Constraints on Fault-Tolerant Gaits for a Locked Joint Failure". In: *Journal of Intelligent and Robotic Systems* 45.4 (2006), pp. 323–342 24

[109] C. F. Pana, I. C. Resceanu, and D. M. Patrascu. "Fault-Tolerant Gaits of Quadruped Robot on a Constant-Slope Terrain". In: *2008 IEEE International Conference on Automation, Quality and Testing, Robotics*. Vol. 1. 2008, pp. 222–226 24

[110] M. M. Gor, P. M. Pathak, A. K. Samantaray, J. -M. Yang, and S. W. Kwak. "Fault Accommodation in Compliant Quadruped Robot through a Moving Appendage Mechanism". In: *Mechanism and Machine Theory* 121 (2018), pp. 228–244 24

[111] Sylvain Koos, Antoine Cully, and Jean-Baptiste Mouret. "Fast Damage Recovery in Robotics with the T-resilience Algorithm". In: *International Journal of Robotics Research* 32.14 (2013), pp. 1700–1723 24

[112] Dibya Ghosh, Homer Rich Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Quan Vuong, Ted Xiao, Pannag R. Sanketi, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. "Octo: An Open-Source Generalist Robot Policy". In: *Robotics: Science and Systems XX, Delft, the Netherlands, July 15-19, 2024*. 2024 24

[113] Konstantinos Bousmalis, Giulia Vezzani, Dushyant Rao, Coline Devin, Alex X Lee, Maria Bauza, Todor Davchev, Yuxiang Zhou, Agrim Gupta, Akhil Raju, Antoine Laurens, Claudio Fantacci, Valentin Dalibard, Martina Zambelli, Murilo Martins, Rugile Pevceviciute, Michiel Blokzijl, Misha Denil, Nathan Batchelor, Thomas Lampe, Emilio Parisotto, Konrad Żołna, Scott Reed, Sergio Gómez Colmenarejo, Jon Scholz, Abbas Abdolmaleki, Oliver Groth, Jean-Baptiste Regli, Oleg Sushkov, Tom Rothörl, José Enrique Chen, Yusuf Aytar, Dave Barker, Joy Ortiz, Martin Riedmiller, Jost Tobias Springenberg, Raia Hadsell, Francesco Nori, and Nicolas Heess. "RoboCat: A Self-Improving Foundation Agent for Robotic Manipulation". In: *arXiv preprint arXiv:2306.11706* () 24

[114] Jonathan Heewon Yang, Catherine Glossop, Arjun Bhorkar, Dhruv Shah, Quan Vuong, Chelsea Finn, Dorsa Sadigh, and Sergey Levine. "Pushing the Limits of Cross-Embodiment Learning for Manipulation and Navigation". In: *Robotics: Science and Systems XX, Delft, the Netherlands, July 15-19, 2024*. 2024 24

[115] Lawrence Yunliang Chen, Kush Hari, Karthik Dharmarajan, Chenfeng Xu, Quan Vuong, and Ken Goldberg. "Mirage: Cross-Embodiment Zero-Shot Policy Transfer with Cross-Painting". In: *arXiv preprint arXiv:2402.19249* (2024) 24

[116] DeepSeek-AI et al. "DeepSeek-V2: A Strong, Economical, and Efficient Mixture-of-Experts Language Model". In: *arXiv preprint arXiv:2405.04434* (2024) 26

[117] DeepSeek-AI et al. "DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning". In: *arXiv preprint arXiv:2501.12948* (2025) 26

[118] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. "Learning Transferable Visual Models From Natural Language Supervision". In: *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*. Vol. 139. 2021, pp. 8748–8763 26

[119] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. "Sigmoid Loss for Language Image Pre-Training". In: *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2023, pp. 11941–11952 26

[120] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. "Visual Instruction Tuning". In: *arXiv preprint arXiv:2304.08485* (2023) 26

[121] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor

Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. "Llama 2: Open Foundation and Fine-Tuned Chat Models". In: *arXiv preprint arXiv:2307.09288* (2023) 26

[122] Roya Firoozi, John Tucker, Stephen Tian, Anirudha Majumdar, Jiankai Sun, Weiyu Liu, Yuke Zhu, Shuran Song, Ashish Kapoor, Karol Hausman, Brian Ichter, Danny Driess, Jiajun Wu, Cewu Lu, and Mac Schwager. "Foundation Models in Robotics: Applications, Challenges, and the Future". In: *Int. J. Robotics Res.* 44.5 (2025), pp. 701–739 26

[123] Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. "PaLM-E: An Embodied Multimodal Language Model". In: *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*. Vol. 202. 2023, pp. 8469–8488 26

[124] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. "Diffusion Policy: Visuomotor Policy Learning via Action Diffusion". In: *arXiv preprint arXiv:2303.04137* (2024) 26

[125] Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. "RDT-1B: A Diffusion Foundation Model for Bimanual Manipulation". In: *arXiv preprint arXiv:2410.07864* (2025) 26

[126] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. "LoRA: Low-Rank Adaptation of Large Language Models". In: *arXiv preprint arXiv:2106.09685* (2021) 26

[127] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. "Decision Transformer: Reinforcement Learning via Sequence Modeling". In: *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, Neurips 2021, December 6-14, 2021, Virtual*. 2021, pp. 15084–15097 27

[128] Yujin Tang, Wenhao Yu, Jie Tan, Heiga Zen, Aleksandra Faust, and Tatsuya Harada. "SayTap: Language to Quadrupedal Locomotion". In: *Conference on Robot Learning, Corl 2023, 6-9 November 2023, Atlanta, GA, USA*. Vol. 229. 2023, pp. 3556–3570 27, 31, 72

[129] Ruihan Yang, Minghao Zhang, Nicklas Hansen, Huazhe Xu, and Xiaolong Wang. "Learning Vision-Guided Quadrupedal Locomotion End-to-End with Cross-Modal Transformers". In: *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. 2022 27, 31

[130] Qingwen Bu, Hongyang Li, Li Chen, Jisong Cai, Jia Zeng, Heming Cui, Maoqing Yao, and Yu Qiao. "Towards Synergistic, Generalized, and Efficient Dual-System for Robotic Manipulation". In: *arXiv preprint arXiv:2410.08001* (2025) 28

[131] Dikai Liu, Tianwei Zhang, Jianxiong Yin, and Simon See. "Masked Sensory-Temporal Attention for Sensor Generalization in Quadruped Locomotion". In: *arXiv preprint arXiv:2409.03332* (2025) 29, 72, 73

[132] Milad Shafiee, Guillaume Bellegarda, and Auke Ijspeert. "ManyQuadrupeds: Learning a Single Locomotion Policy for Diverse Quadruped Robots". In: *2024 IEEE International Conference on Robotics and Automation (ICRA)*. 2024, pp. 3471–3477 30

[133] Carmelo Sferrazza, Dun-Ming Huang, Fangchen Liu, Jongmin Lee, and Pieter Abbeel. "Body Transformer: Leveraging Robot Embodiment for Policy Learning". In: *Conference on Robot Learning, 6-9 November 2024, Munich, Germany*. Vol. 270. 2024, pp. 3407–3424 30, 73

[134] Wanming Yu, Chuanyu Yang, Christopher McGreavy, Eleftherios Triantafyllidis, Guillaume Bellegarda, Milad Shafiee, Auke Jan Ijspeert, and Zhibin Li. "Identifying Important Sensory Feedback for Learning Locomotion Skills". In: *Nature Machine Intelligence* 5.8 (8 2023), pp. 919–932 30

[135] Gavin Kenneally, Avik De, and D. E. Koditschek. "Design Principles for a Family of Direct-Drive Legged Robots". In: *IEEE Robotics and Automation Letters* 1.2 (2016), pp. 900–907 30

[136] Zipeng Fu, Tony Z. Zhao, and Chelsea Finn. "Mobile ALOHA: Learning Bimanual Mobile Manipulation with Low-Cost Whole-Body Teleoperation". In: *arXiv preprint arXiv:2401.02117* (2024) 31

[137] Ken Caluwaerts, Atil Iscen, J. Chase Kew, Wenhao Yu, Tingnan Zhang, Daniel Freeman, Kuang-Huei Lee, Lisa Lee, Stefano Saliceti, Vincent Zhuang, Nathan Batchelor, Steven Bohez, Federico Casarini, Jose Enrique Chen, Omar Cortes, Erwin Coumans, Adil Dostmohamed, Gabriel Dulac-Arnold, Alejandro Escontrela, Erik Frey, Roland Hafner, Deepali Jain, Bauyrjan Jyenis, Yuheng Kuang, Edward Lee, Linda Luu, Ofir Nachum, Ken Oslund, Jason Powell, Diego Reyes, Francesco Romano, Feresteh Sadeghi, Ron Sloat, Baruch Tabanpour, Daniel Zheng, Michael Neunert, Raia Hadsell, Nicolas Heess, Francesco Nori, Jeff Seto, Carolina Parada, Vikas Sindhwani, Vincent Vanhoucke, and Jie Tan. "Barkour: Benchmarking Animal-Level Agility with Quadruped Robots". In: *arXiv preprint arXiv:2305.14654* (2023) 31, 32

[138] Christoph Feichtenhofer, Haoqi Fan, Yanghao Li, and Kaiming He. "Masked Autoencoders As Spatiotemporal Learners". In: *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, Neurips 2022, New Orleans, LA, USA, November 28 - December 9, 2022*. 2022 32, 38

[139] I Made Aswin Nahrendra, Byeongho Yu, and Hyun Myung. "DreamWaQ: Learning Robust Quadrupedal Locomotion with Implicit Terrain Imagination via Deep Reinforcement Learning". In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. 2023, pp. 5078–5084 33, 35, 69, 72–74, 76, 84

[140] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. "Proximal Policy Optimization Algorithms". In: *arXiv preprint arXiv:1707.06347* (2017) 36, 45, 58, 81, 84

[141] Lucas Beyer, Xiaohua Zhai, and Alexander Kolesnikov. "Better Plain ViT Baselines for ImageNet-1k". In: *arXiv preprint arXiv:2205.01580* (2022) 37

[142] Feyza Duman Keles, Pruthuvi Mahesakya Wijewardena, and Chinmay Hegde. "On the Computational Complexity of Self-Attention". In: *International Conference on Algorithmic Learning Theory, February 20-23, 2023, Singapore*. Vol. 201. 2023, pp. 597–619 38

[143] Dikai Liu, Jianxiong Yin, and Simon See. "Towards Fault-Tolerant Quadruped Locomotion with Reinforcement Learning". In: *2024 IEEE Conference on Artificial Intelligence (CAI)*. 2024, pp. 1438–1441 49

[144] Dikai Liu, Tianwei Zhang, Jianxiong Yin, and Simon See. "Saving the Limping: Fault-Tolerant Quadruped Locomotion via Reinforcement Learning". In: *arXiv preprint arXiv:2210.00474* (2023) 49

[145] Zhijun Chen, Qingxing Xi, Feng Gao, and Yue Zhao. "Fault-Tolerant Gait Design for Quadruped Robots with One Locked Leg Using the GF Set Theory". In: *Mechanism and Machine Theory* 178 (2022), p. 105069 50

[146] Hae-Won Park, Patrick M. Wensing, and Sangbae Kim. "Jumping over Obstacles with MIT Cheetah 2". In: *Robotics and Autonomous Systems* 136 (2021), p. 103703 50

[147] Fan Shi, Timon Homberger, Joonho Lee, Takahiro Miki, Moju Zhao, Farbod Farshidian, Kei Okada, Masayuki Inaba, and Marco Hutter. "Circus ANYmal: A Quadruped Learning Dexterous Manipulation with Its Limbs". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. 2021, pp. 2316–2323 50

[148] Christian Gehring, Stelian Coros, Marco Hutter, Carmine Dario Bellicoso, Huub Heijnen, Remo Diethelm, Michael Bloesch, Peter Fankhauser, Jemin Hwangbo, Mark Hoepflinger, and Roland Siegwart. "Practice Makes Perfect: An Optimization-Based Approach to Controlling Agile Motions for a Quadruped Robot". In: *IEEE Robotics and Automation Magazine* 23.1 (2016), pp. 34–43 50

[149] Timothée Anne, Jack Wilkinson, and Zhibin Li. "Meta-Learning for Fast Adaptive Locomotion with Uncertainties in Environments and Robot Dynamics". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2021, Prague, Czech Republic, September 27 - Oct. 1, 2021*. 2021, pp. 4568–4575 51, 53, 54, 70

[150] Wataru Okamoto, Hiroshi Kera, and Kazuhiko Kawamoto. "Reinforcement Learning with Adaptive Curriculum Dynamics Randomization for Fault-Tolerant Robot Control". In: *arXiv preprint arXiv:2111.10005* (2021) 51, 53, 54

[151] Ananye Agarwal, Ashish Kumar, Jitendra Malik, and Deepak Pathak. "Legged Locomotion in Challenging Terrains Using Egocentric Vision". In: *Conference on Robot Learning, Corl 2022, 14-18 December 2022, Auckland, New Zealand*. Vol. 205. 2022, pp. 403–415 58, 62

[152] Haiping Wu, Khimya Khetarpal, and Doina Precup. "Self-Supervised Attention-Aware Reinforcement Learning". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 35.12 (12 2021), pp. 10311–10319 59

[153] Seunghyun Lee, I. Made Aswin Nahrendra, Dongkyu Lee, Byeongho Yu, Minho Oh, and Hyun Myung. "DreamFLEX: Learning Fault-Aware Quadrupedal Locomotion Controller for Anomaly Situation in Rough Terrains". In: *arXiv preprint arXiv:2502.05817* (2025) 69

[154] Tianyu Xu, Yaoyu Cheng, Pinxi Shen, and Lin Zhao. "AcL: Action Learner for Fault-Tolerant Quadruped Locomotion Control". In: *arXiv preprint arXiv:2503.21401* (2025) 70

[155] Dikai Liu, Tianwei Zhang, Jianxiong Yin, and Simon See. "Unified Locomotion Transformer with Simultaneous Sim-to-Real Transfer for Quadrupeds". In: *arXiv preprint arXiv:2503.08997* (2025) 71

[156] Wenhao Yu, Jie Tan, C. Karen Liu, and Greg Turk. "Preparing for the Unknown: Learning a Universal Policy with Online System Identification". In: *Robotics: Science and Systems XIII, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, July 12-16, 2017*. 2017 72

[157] Gwanghyeon Ji, Juhyeok Mun, Hyeongjun Kim, and Jemin Hwangbo. "Concurrent Training of a Control Policy and a State Estimator for Dynamic and Robust Legged Locomotion". In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 4630–4637 72

[158] Lerrel Pinto, Marcin Andrychowicz, Peter Welinder, Wojciech Zaremba, and Pieter Abbeel. "Asymmetric Actor Critic for Image-Based Robot Learning". In: *Robotics: Science and Systems XIV*. 2018 72