

Copyright
by
Mingyo Seo
2026

The Dissertation Committee for Mingyo Seo
certifies that this is the approved version of the following dissertation:

Embodiment-Aware Skill Learning for Diverse Robots

Committee:

Yuke Zhu, Supervisor

Luis Sentis, Co-supervisor

Peter Stone

Amy Zhang

Dongheui Lee

Embodiment-Aware Skill Learning for Diverse Robots

by
Mingyo Seo

Dissertation

Presented to the Faculty of the Graduate School of
The University of Texas at Austin
in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

The University of Texas at Austin
May 2026

Acknowledgments

This dissertation reflects my Ph.D. journey, shaped by achievements, challenges, and continuous ups and downs. Looking back to the time before I started my Ph.D., I found myself uncertain and searching for direction through moments of failure and disappointment. Over time, this journey gave me the time to reflect, explore, and gradually discover my own goals. I was fortunate to grow along the way with the support of my mentors, friends, family, and many others around me.

Foremost, I would like to express my deepest gratitude to my advisors, Yuke Zhu and Luis Sentis, for their unwavering support throughout my Ph.D. journey. As educators, they encouraged me to pursue my intellectual curiosity while shaping my research direction and fostering my intellectual independence. As mentors, they patiently guided me through challenges and supported my growth at every stage. Reflecting on this journey, starting my Ph.D. under their guidance was one of the most fortunate opportunities I have had. I also note that many of my works are named using musical terms, following Yuke’s suggestion, which became a small but memorable source of enjoyment throughout my Ph.D. research.

I would also like to express my sincere gratitude to my dissertation committee members, Peter Stone, Amy Zhang, and Dongheui Lee. Their rigorous and insightful feedback played a key role in shaping the research philosophy of this dissertation and strengthening my critical thinking. Their guidance has been invaluable in preparing me for the job search and for engaging with the broader research community.

I am also grateful for the opportunity to interact and collaborate with colleagues in both the Robot Perception and Learning Lab and the Human-Centered Robotics Lab at UT Austin, including Yifeng Zhu, Huihan Liu, Zhenyu Jiang, Rutav Shah, Jake Grigsby, Heeyong Huh, Dongho Kang, Steve Han, Alexy Skoutnev, Jack Pittenger, Yiheng Ji, and many others whom I have not named here. It was a pleasure to exchange insights and, from time to time, share a few lighthearted jokes.

I also feel fortunate to have worked with wonderful collaborators both within and beyond the university, including Jamie Cho, Yoonchang Sung, Beomjoon Kim, Siqi Shang, Lillian Chin, H. Andy Park, Shelli Yuan, and many others. Interacting with them broadened my perspective, pushed me beyond my comfort zone, and helped me grow both intellectually and professionally.

Finally, and most importantly, I have been fortunate to receive unwavering love and support from my family and friends. Beyond being a researcher, their presence has grounded me throughout this journey. While pursuing intellectual curiosity is an important part of my life, it is ultimately rooted in relationships that give it meaning. My Ph.D. has been one chapter of that life, made all the more meaningful by the people who have been part of it. I am deeply grateful to them.

I am sincerely grateful to everyone who has been part of this journey. It would not have been possible without their support.

Abstract

Embodiment-Aware Skill Learning for Diverse Robots

Mingyo Seo, PhD
The University of Texas at Austin, 2026

SUPERVISORS: Yuke Zhu, Luis Sentis

Robotic systems are rapidly diversifying in both form and application, spanning humanoids, mobile manipulators, and industrial platforms. Achieving generalist robot autonomy therefore requires skills that can be shared across robots while accounting for embodiment-specific constraints. Recent efforts have pursued this goal by scaling data across multiple platforms, learning structural differences and invariances from large datasets. However, robots, as engineered systems, exhibit behavior shaped by design choices in hardware, control, and representation. Rather than relying on learning algorithms to infer this structure from data alone, we develop embodiment-aware skill learning, which explicitly incorporates these system-level factors as structural priors to guide learning toward physically feasible and transferable behaviors.

This dissertation presents a holistic approach that combines hardware interfaces, control abstractions, and embodiment-aware representations. It begins with hardware design that enables consistent physical interaction and sensing across diverse robots. It then introduces control abstractions that structure learning within compact state-action spaces for high-degree-of-freedom systems. Finally, it develops representations that separate shared, object-centric task intent from embodiment-specific geometric constraints, enabling efficient skill transfer while maintaining feasibility and performance across different robot morphologies.

Our findings show that incorporating structural priors complements and strengthens skill learning, enabling robots to accumulate and transfer skills across embodiments while maintaining physical feasibility. This accelerates progress toward generalist robot autonomy, supporting scalable skill acquisition and reliable performance across a broad spectrum of rapidly evolving robot platforms.

Table of Contents

List of Tables	12
List of Figures	13
List of Algorithms	15
Chapter 1: Introduction	16
1.1 Background	16
1.2 Dissertation Overview	19
1.3 Other Doctoral Research Work	27
Chapter 2: Hardware Design for a Shared Interface	28
2.1 Introduction	28
2.2 Related Work	31
2.3 Method	33
2.3.1 Problem Formulation	33
2.3.2 Actions Based on the Handheld Gripper	34
2.3.3 Whole-body Motion Retargeting	35
2.3.4 Training of Visuomotor Policies	36
2.3.5 Motion-Invariant Regularization Loss	38
2.4 Experiments	40
2.4.1 LEGATO Gripper	40
2.4.2 Experimental Setup	42
2.4.3 Quantitative Evaluation in Simulation	43
2.4.4 Varying Demonstration Embodiments	46
2.4.5 Impact of Training with the Motion-Invariant Regularization	47
2.4.6 Real-Robot Experiments	48
2.5 Conclusion	49
Chapter 3: Enhancing Robot Capabilities through Sensor Embedding	51
3.1 Introduction	51
3.2 Related Work	53
3.3 Method	54
3.3.1 Hardware Design and Sensorization	55
3.3.2 Force Estimation Algorithm	56
3.3.3 Slip Definition and Detection	57
3.4 Characterization	59

3.4.1	Grip Force Estimation	59
3.4.2	Slip Detection	60
3.5	Experiments	63
3.5.1	Experimental Setup	64
3.5.2	Grasping Performance	64
3.5.3	Slip Detection Performance	66
3.6	Conclusion	67
Chapter 4:	Visuomotor Skill Learning with Model-Based Control	69
4.1	Introduction	69
4.2	Related Work	71
4.3	Method	72
4.3.1	Problem Formulation	73
4.3.2	Teleoperation System	74
4.3.3	Whole-body Control	75
4.3.4	Training of Visuomotor Policies	76
4.4	Experiments	77
4.4.1	Experimental Setup	78
4.4.2	Quantitative Evaluation in Simulation	80
4.4.3	Real-Robot Deployment	85
4.5	Conclusion	87
Chapter 5:	Visuomotor Skill Learning with Learning-Based Control	88
5.1	Introduction	88
5.2	Related Work	90
5.3	Method	92
5.3.1	Problem Formulation	92
5.3.2	Hierarchical Perceptive Locomotion Policy	93
5.4	Experiments	96
5.4.1	Experimental Setup	96
5.4.2	Quantitative Evaluation	99
5.5	Conclusion	105

Chapter 6: Capturing Task Intent from Demonstration through Object-Centric Representation	106
6.1 Introduction	106
6.2 Related Work	108
6.3 Method	109
6.3.1 Problem Formulation	110
6.3.2 Reference Plan Generation	110
6.3.3 Object-Aware Retargeting	112
6.4 Experiments	113
6.4.1 Experimental Setup	113
6.4.2 Quantitative Evaluation	115
6.4.3 Impact of Body Motion Retargeting	116
6.4.4 Robustness to Demonstrator Variability	117
6.5 Conclusion	117
Chapter 7: Embodiment-Aware Trajectory Generation through Configuration-Space Representation	119
7.1 Introduction	119
7.2 Related Work	122
7.3 Method	123
7.3.1 Problem Formulation	124
7.3.2 Background	124
7.3.3 Environment Representation	125
7.3.4 Training Conditional Diffusion Models	127
7.3.5 Trajectory Generation	129
7.3.6 Incorporating Guidance During Sampling	130
7.4 Experiments	131
7.4.1 Experimental Setup	131
7.4.2 Quantitative Evaluation	133
7.4.3 Ablation Studies	135
7.4.4 Impact of Guided-Sampling	137
7.4.5 Computation Time Details	138
7.5 Conclusion	139

Chapter 8: Conclusion	140
8.1 Summary	140
8.2 Future Research	140
8.2.1 Designing Hardware for Task Diversity	141
8.2.2 Devising Principles of Control for Skill Learning	142
8.2.3 Scaling Knowledge Sharing with Embodiment Awareness	143
8.3 Final Remarks	143
Works Cited	145

List of Tables

1.1	Control abstraction in skill learning	25
1.2	Action space design in skill learning	26
1.3	Action representation design in skill learning	26
3.1	Slip detection parameters	59
3.2	Grasp success rates and slip detection performance	68
4.1	Quantitative results in simulation	81
4.2	Ablation studies of TRILL in simulation	85
5.1	Comparison of PRELUDE and baselines in simulation	99
5.2	Evaluation of gait controllers	102

List of Figures

1.1	Designing robot hardware for consistent physical interaction	20
1.2	Bridging control architectures and skill learning	22
1.3	Learning embodiment-aware representations for skill transfer	23
2.1	Overview of LEGATO	29
2.2	LEGATO’s cross-embodiment learning pipeline	33
2.3	High-level visuomotor policy architecture	37
2.4	LEGATO Gripper design	41
2.5	Timelapse of deploying LEGATO in simulation	43
2.6	Quantitative results in simulation	44
2.7	Simulation evaluation across different demonstration embodiments . .	46
2.8	Impact of training with the motion-invariant regularization	47
2.9	Timelapse of deploying LEGATO in the real-robot system	49
3.1	Overview of FORTE	52
3.2	Hardware design of FORTE	55
3.3	Grip force data collection	60
3.4	Grip force estimation results	61
3.5	Slip detection characterization	62
3.6	Experimental setup	63
4.1	Overview of TRILL	70
4.2	Model architecture of TRILL	74
4.3	Timelapse of deploying TRILL in simulation	79
4.4	NASA TLX evaluation and completion times using the teleoperation system	82
4.5	Simulation evaluation across different dataset sizes	83
4.6	Timelapse of deploying TRILL in the real-robot system	86
5.1	Overview of PRELUDE	89
5.2	Model architecture of PRELUDE	93
5.3	Hardware platforms of PRELUDE	98
5.4	Real-robot deployment in novel environments	103

5.5	Real-robot experiments	104
6.1	Overview of OKAMI	107
6.2	Timelapse of human demonstrations and robot rollouts	114
6.3	Task performance and failure analysis of OKAMI	115
6.4	Task performance of OKAMI under demonstrator variability	117
7.1	Overview of PRESTO	121
7.2	Trajectory generation pipeline of PRESTO	127
7.3	Main results	132
7.4	Ablation study results	136
7.5	Results with guided-sampling	137
7.6	Ablation studies with guided-sampling	138
7.7	Breakdown of computation time for PRESTO	139

List of Algorithms

7.1	Key-Configuration Selection	126
7.2	PRESTO Trajectory Generation	130
7.3	Guidance Step with Cost Gradients	131

Chapter 1: Introduction

1.1 Background

Recent advances in robotics, ranging from household assistance to industrial assembly and logistics, highlight the potential for robots to transform both daily life and industry. As demand for robotic systems continues to grow and hardware technologies rapidly evolve, robots are becoming increasingly diverse in their designs and capabilities. Some robots are built at human scale to operate in everyday environments [1, 4], while others are designed for heavy-duty, long-duration, or hazardous tasks [2, 3, 108]. These differences reflect the wide range of roles robots are expected to perform across varied environments and applications. This growing diversity also introduces new challenges for autonomy, as robots must perform a wide range of tasks in complex and unpredictable real-world settings.

To support such flexibility, modern robot autonomy has increasingly shifted toward data-driven approaches. In this paradigm, robotic skills are learned by training policies that map observations to actions using datasets collected from robot interactions [131, 144, 148, 251]. This allows robots to acquire behaviors that adapt to variability in objects and environments. In contrast to traditional rule-based methods, where behaviors were explicitly programmed and required detailed modeling of task dynamics and environmental conditions, data-driven approaches enable robots to learn complex behaviors without requiring explicit modeling of every possible situation. As a result, they provide a practical way to develop robotic capabilities that operate in diverse and uncertain real-world environments.

Despite their flexibility, data-driven approaches often rely on large datasets collected on specific robot platforms. This reliance makes it difficult to transfer learned capabilities across different robot systems and to adapt to rapidly evolving hardware driven by technological advances and industrial demands. Variations in

sensing, actuation, and dynamics limit the reuse of learned behaviors, often requiring substantial data collection for each platform. Consequently, skill development remains costly and difficult to scale across diverse robotic systems. Addressing these challenges requires generalist robot autonomy that operates across diverse robots by reusing previously acquired knowledge. This, in turn, depends on identifying which aspects of knowledge can be shared across embodiments and which must adapt to each robot’s physical structure.

One natural response is to scale data-driven learning across many robots. The intuition is that sufficient diversity of experience can expose embodiment variation directly in the data. Driven by large academic collaborations and significant industrial investment, recent work has begun to scale up robot datasets [40, 110]. These efforts have demonstrated generalization across robot embodiments using large end-to-end models trained on diverse data [18, 20, 160]. However, collecting large-scale physical interaction data remains costly and time-consuming. Physical interaction is inherently complex, diverse, and uncertain, often involving long-horizon dependencies between perception, action, and environment dynamics. As a result, achieving broad generalization in robotics may require even more data than domains such as vision or language. Consequently, scaling robot data to support generalist robot autonomy remains difficult in practice.

These limitations call for a different approach. Unlike language and vision, in which meaning arises from naturally developed perceptual and cognitive processes, robots are engineered systems with well-defined dynamics and structural designs. Ignoring these intrinsic properties forces learning systems to rediscover physical principles from data alone. This dissertation therefore aims to develop generalizable robot skills across diverse embodiments through embodiment-aware learning that explicitly accounts for a robot’s physical structure and control constraints. Leveraging such domain priors reduces reliance on brute-force data scaling and improves the efficiency and transferability of learned skills across robot platforms. Since structure funda-

mentally shapes robot behavior, generalizable skill learning must explicitly account for embodiment.

The proposed embodiment-aware approach builds on this idea by incorporating robot-specific structure into the skill learning process, rather than relying on data to rediscover it. By separating shared task intent from embodiment-specific constraints, it enables skill transfer across diverse robots. This research integrates structural priors from robotic systems to unify skill learning and transfer across embodiments, reducing robot-specific complexity and improving data efficiency in collection, learning, and deployment. This perspective enables transfer across distinct robots without requiring large datasets or complex models.

To this end, this dissertation advances three key components of an embodiment-aware approach:

1. **Designing robot hardware for consistent physical interaction:** Robot skills are constrained by hardware capabilities. When robots interact with the world through incompatible end-effectors or sensing modalities, learned behaviors become difficult to transfer. This work develops end-effector tools that provide a shared physical interaction interface across robots, enabling consistent action-observation spaces and supporting skill transfer and scalable data collection across diverse robot embodiments.
2. **Bridging control architectures and skill learning:** Robot dynamics are complex and vary across embodiments, making end-to-end learning from raw observations to low-level control challenging to generalize. Control architectures provide structure that simplifies these learning problems by handling dynamics and stability constraints. This work integrates control with learning so that policies operate in compact, task-relevant action spaces, enabling efficient skill acquisition and deployment on robots with complex dynamics.

3. **Learning embodiment-aware representations for skill transfer:** Differences in morphology, sensing, and actuation often create incompatibilities that hinder skill reuse across robots. Representations capture embodiment-specific variations while preserving task-relevant structure shared across robots. This work develops such representations to enable efficient skill acquisition and transfer across diverse robot embodiments.

1.2 Dissertation Overview

This dissertation is structured into three main parts. Part I focuses on hardware designs that enable consistent physical interactions across robots, allowing skills to be defined in a shared interaction space for reliable transfer. Through a shared hardware interface, these interactions can be applied across diverse robots, enabling skills learned in one embodiment to transfer to others while compensating for capabilities that may be absent in the original robot hardware. Part II investigates control architectures that structure the complex dynamics of robot systems to make skill learning more tractable. To address the difficulty of end-to-end learning from raw sensory observations to actions, this part introduces control-driven abstractions that handle complex robot dynamics while preserving expressive motion. These abstractions are developed through model-based control approaches that leverage known physical priors, as well as learning-based control methods when accurate models are unavailable. Part III addresses the remaining challenges caused by differences in robot morphology. This part focuses on learning embodiment-aware representations that enable skills to adapt across different robots. Object-centric representations capture task intent from demonstrations and support knowledge sharing across embodiments, while embodiment-specific configuration-space representations realize these tasks by generating feasible trajectories for the target robot.

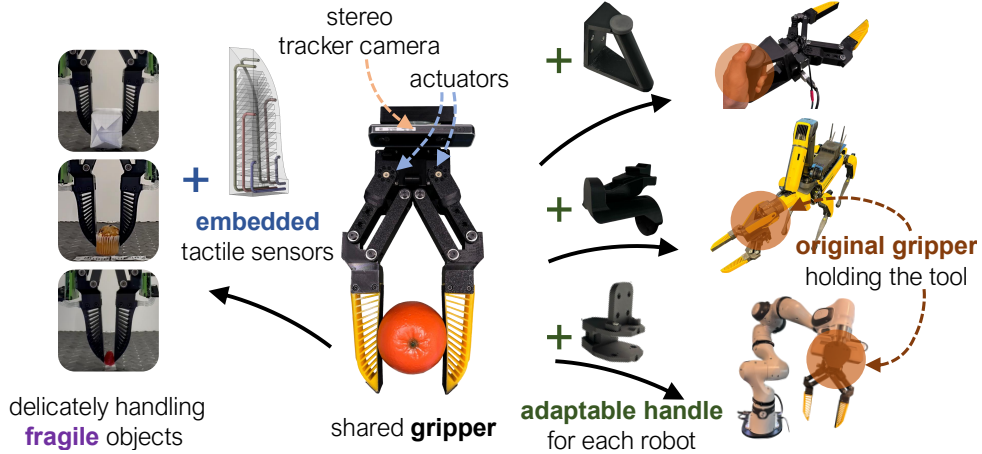


Figure 1.1: Designing robot hardware for consistent physical interaction. A shared end-effector interface standardizes grasping, sensing, and manipulation across diverse robots while providing unified sensor modalities that enable richer interaction capabilities for diverse task requirements.

Part I: Designing Robot Hardware for Consistent Physical Interaction

A core challenge in robot autonomy is that each platform operates within its own domain, making skills hardware-specific and difficult to transfer. While most hardware components are fixed, interaction with the environment occurs primarily through end-effectors, making them a natural locus for co-design with sensorimotor abstractions. In Part I, we develop a series of end-effector grippers [193, 196] that unify how robots grasp, sense, and manipulate objects. These grippers standardize visual and tactile perception as well as contact interactions across morphologies, allowing abstraction layers to interpret tactile and force feedback consistently. Serving as a shared interface for skill learning, they establish a unified action-observation space that supports cross-embodiment transfer, skill reuse, and scalable data collection.

Chapter 2 introduces transferable demonstrations across robots and humans using a handheld gripper [193]. Users can perform tasks directly for data collection, reducing cognitive load compared to teleoperation, while robots holding the same gripper can be teleoperated to gather demonstrations. A visuomotor policy is trained within the gripper’s unified action-observation space, and the resulting target motions

are retargeted through inverse kinematics, enabling high-degree-of-freedom execution across robot arms, humanoids, and quadrupeds.

Building on this foundation, Chapter 3 extends the capabilities of the shared gripper beyond the sensing limitations of the original robot hardware. By embedding high-frequency, low-latency sensors into the compliant fingers of the shared handheld gripper [196], the system enables force estimation, slip detection, and shape adaptability across platforms. This design enables reliable manipulation of fragile, slippery, and deformable objects such as raspberries, potato chips, billiard balls, and aluminum cans, demonstrating how hardware co-design can enhance sensing capabilities for contact-rich manipulation across diverse robot platforms with varying sensing configurations.

Together, these works illustrate that hardware is not merely a constraint but an active component of robot learning. They show how handheld grippers can augment robots’ native capabilities while serving as a unified interface for transferable skill learning across diverse robotic platforms.

Part II: Bridging Control Architectures and Skill Learning

High-degree-of-freedom robots such as humanoids and legged systems offer rich capabilities but are difficult to control due to high-dimensional state-action spaces and complex dynamics. These challenges make skill learning inefficient, as even simple tasks can require large datasets. Introducing abstractions through intermediate controllers can reduce this complexity while preserving expressive robot motion and control. In Part II, we develop hybrid learning frameworks [190, 191] that manage complex whole-body dynamics within lower-dimensional action spaces by optimizing control outputs under dynamic constraints. These control-driven abstractions simplify demonstrations, improve data efficiency, and enable scalable skill learning on robot systems whose dynamics would otherwise be difficult to learn directly.

Chapter 4 develops control-driven abstractions through a model-based con-

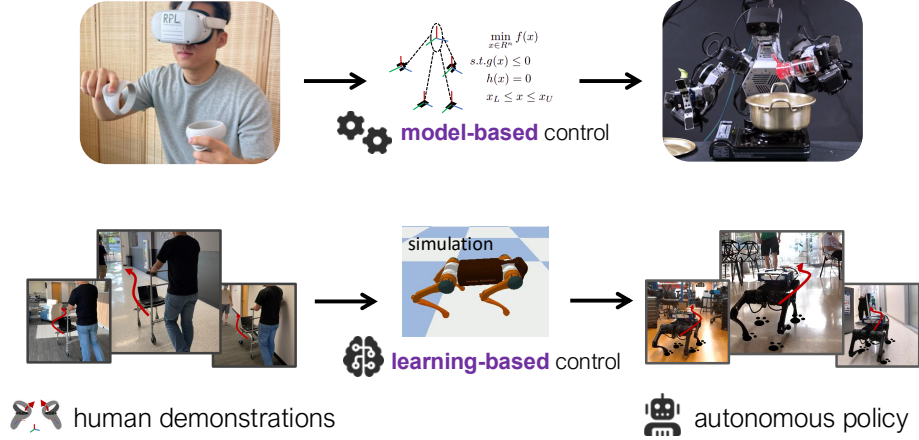


Figure 1.2: Bridging control architectures and skill learning. Control architectures provide structured abstractions that reduce high-dimensional robot dynamics to compact action spaces, simplifying policy learning from demonstrations. This improves data efficiency and supports scalable learning across diverse embodiments.

trol architecture when accurate robot dynamics are available. In humanoid locomanipulation [191], we introduce a whole-body controller based on a floating-base dynamics model that maps task-space hand trajectories and locomotion commands to torque-level actions. This architecture reduces the cognitive load on demonstrators while enabling visuomotor policies to be trained in a simplified state-action space. As a result, the learned policies remain robust to complex whole-body dynamics and enable reliable execution of challenging bimanual manipulation tasks on a real humanoid robot.

However, model-based controllers often rely on simplifying assumptions to make control computation tractable, such as convexity assumptions in robot dynamics and the availability of accurate dynamic models. When these assumptions break down, complementary learning-based control methods can provide greater flexibility. Chapter 5 therefore explores scenarios in which these assumptions are violated, such as when robots must perform agile motions in dynamic environments and reactively avoid nearby humans for safety. In this setting, a learning-based controller [190] is trained in simulation using reinforcement learning to achieve agile and stable locomotion. Navigation skills are then learned from human demonstrations collected at

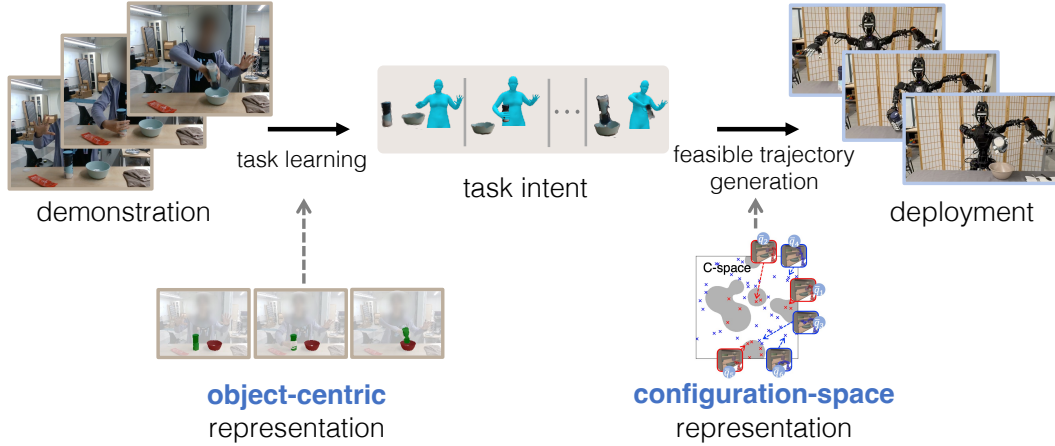


Figure 1.3: Learning embodiment-aware representations for skill transfer. Representations can be designed to decompose task intent and embodiment-specific constraints for cross-embodiment transfer. Object-centric representations capture embodiment-agnostic task intent from demonstrations, while configuration-space representations model embodiment-specific constraints and enable feasible trajectory generation on the target robot.

human-scale walking speeds, without requiring the demonstrator to directly operate the robot. These skills are subsequently transferred to real robots operating in previously unseen environments.

Collectively, these works demonstrate how combining model-based and learning-based control architectures enables scalable skill learning by allowing each component to be developed in the domain where it is most effective.

Part III: Learning Embodiment-Aware Representations for Skill Transfer

Achieving scalable robot autonomy requires skills that transfer across diverse robot embodiments while adapting to each platform’s physical constraints. While shared hardware interfaces and control abstractions enable transferable interactions, successful deployment still requires adapting learned behaviors to differences in morphology and kinematics. In Part III, we develop embodiment-aware representations that capture both task intent and embodiment-specific feasibility, enabling skills learned from demonstrations to transfer across robots.

Chapter 6 focuses on extracting task intent from demonstrations through object-centric representations [133, 195]. Because demonstrations are often performed by specific embodiments, robots must infer task intent from how demonstrators interact with objects. To achieve this, keyframes corresponding to task-relevant object states are extracted from the demonstrations. Since object states alone are insufficient to capture how objects are manipulated throughout the task, the motions of the demonstrators are additionally reconstructed. Together, these form an object-centric representation that captures the relationships between object states and embodiment actions. At test time, the robot localizes objects and estimates transformations between states to retarget the interaction plan.

While these representations capture task intent, embodiment-specific constraints such as self-collision and kinematic limits can still prevent direct execution. To address this, Chapter 7 develops embodiment-aware representations based on embodiment-specific configuration spaces (C-space) that enable feasible trajectory generation on the target robot [192]. This representation models not only object states in the workspace but also feasibility under embodiment-specific constraints in C-space. Unlike traditional motion planning methods that explicitly model the full C-space and incur high computational cost, the proposed approach approximates the C-space by encoding collision states at task-relevant joint configurations. Generative models trained on these representations can then produce collision-free trajectories in cluttered environments while maintaining computational efficiency.

Taken together, these works demonstrate how embodiment-aware representations enable flexible skill transfer across robots by separating task intent from embodiment-specific motion generation while fully leveraging each robot’s capabilities.

Table 1.1: Control abstraction in skill learning. Different control methods support different classes of behaviors, determined by their modeling assumptions and priors. These choices shape the types of behaviors that can be reliably achieved under varying dynamics and uncertainty.

	Inverse kinematics	Inverse dynamics	Data-driven policy
Approach	Model-based	Model-based	Model-free / learning-based
Assumption	Neglect of system dynamics	Explicit dynamics model	Minimal structural assumptions
Priors	Kinematic structures	Rigid-body dynamics and contact models	Data and simulation priors
Usage	High-gain tracking regimes	Dynamic behaviors with accurate system identification	Complex behaviors under modeling uncertainty
Example	IK optimization (Chapter 2)	Whole-body control (Chapter 4)	Reinforcement learning (Chapter 5)

Guidelines for Control and Action Design in Skill Learning

In this dissertation, we employ a range of design choices across control abstraction, action space, and action representation to support diverse robotic behaviors, as summarized in Tables 1.1–1.3. These components are selected based on task requirements and the characteristics of robot embodiments.

Control abstraction, discussed in Part II, is primarily determined by task requirements and system dynamics. As shown in Table 1.1, different controllers, ranging from inverse kinematics and inverse dynamics to fully data-driven policies, impose varying levels of structure through their modeling assumptions. Model-based approaches leverage kinematic and dynamic priors for reliable, data-efficient execution when accurate models are available, while learning-based controllers offer flexibility under uncertainty at the cost of higher data requirements. The choice of control abstraction therefore defines how robot dynamics are handled and which classes of behaviors can be robustly achieved, making it a critical interface between learned policies and physical execution.

Action space and representation design are closely tied to representation learning and interaction interfaces. As summarized in Table 1.2, actions can be defined in task space or joint space. Task-space actions specify desired body or end-effector motions, providing an intuitive and often embodiment-agnostic interface for learning and demonstration. Joint-space actions directly operate on robot configurations, enforcing embodiment-specific constraints and enabling precise control. The choice of action space determines how behaviors are expressed and significantly impacts

Table 1.2: Action space design in skill learning. Action spaces define how behaviors are specified with respect to embodiment and control interfaces.

	Task-space	Joint-space
Definition	Targeted body poses	Joint states
Advantage	Embodiment-agnostic, expressive	Embodiment-aware, enforcing kinematic constraints
Limitation	Limited transferability, requiring retargeting	Requiring low-level specification
Usage	Targeted body tracking (Chapters 2, 4, 5)	Full-body control (Chapter 7)

Table 1.3: Action representation design in skill learning. Action representations determine the temporal structure of behaviors and influence responsiveness and adaptability.

	Setpoints	Receding-horizon	Full-trajectory
Definition	Single states	Finite-horizon states	Full trajectories
Advantage	Fast execution, responsive	Sequence-aware, online adaptive	Global consistency; complete behavior specification
Limitation	Limited temporal structure	Limited horizon, higher computation	Limited adaptability to changing environments
Usage	Reactive control (Chapters 2, 4, 5)	Online control with feedback (Chapter 2)	Static or slowly varying environments (Chapters 6, 7)

transferability across embodiments.

Table 1.3 further outlines action representations that define the temporal structure of behavior. Setpoint-based representations enable fast and reactive control, while receding-horizon approaches incorporate short-term temporal context for improved adaptability. Full-trajectory representations provide global consistency and are suitable for tasks with well-defined objectives. These representations govern the trade-off between responsiveness and long-horizon planning, influencing how effectively a system can adapt to dynamic environments while maintaining coherent behavior.

Rather than relying on a fixed formulation, these design choices are adapted to balance structure and flexibility, enabling efficient learning while ensuring feasible execution under embodiment-specific constraints. Through this perspective, the presented methods illustrate how different combinations of control abstractions and action designs can be systematically leveraged to support behaviors ranging from reactive control to long-horizon skill execution while facilitating transfer across heterogeneous robotic systems.

1.3 Other Doctoral Research Work

This dissertation primarily focuses on my doctoral research on skill learning and transfer across diverse robotic embodiments. Beyond this central theme, my research also explores complementary directions across hardware, control, and representation to support robust and scalable robot manipulation. These works include advanced hand hardware designs supporting dexterous manipulation [105], model-predictive control methods that enable fast yet reliable manipulation while avoiding singularities [128], and scalable interfaces for collecting robot operation demonstrations through shared autonomy [138]. Although these contributions are not discussed in detail in this dissertation, they complement the presented work by supporting the broader goal of building scalable and capable autonomous robot systems.

Chapter 2: Hardware Design for a Shared Interface

2.1 Introduction

Recent advancements in robot hardware, from wheeled manipulators to humanoid robots [2, 3, 5, 8, 108], have greatly increased access to diverse robotic platforms. To fully leverage these advancements in supporting human activities, robots must autonomously perform a wide range of complex tasks. Deep imitation learning has shown promise in training autonomous policies for sensorimotor skills, reducing the need for extensive human programming compared to traditional rule-based approaches. It has yielded impressive results in complex robotic systems [191, 233] and across diverse dexterous manipulation tasks [149, 204, 254]. However, such an approach typically requires demonstration data from a specific target robot, which limits scalability due to high hardware costs and the intensive workload involved in operating the robot during demonstrations. Additionally, individualized training for each robot restricts cross-embodiment applications, as data cannot be transferred to different robot systems, even for similar tasks.

To enable scalable demonstration collection, pioneering works have introduced data collection tools that allow humans to directly manipulate during demonstrations [37, 165, 190, 202, 241]. These approaches enable training visuomotor policies that are deployable to specified target robots, reducing human workload and avoiding the costs and risks associated with using real robots for data collection. However, these methods require either designing specialized data collection tools for specific robot gripper mechanisms or replacing robots’ original grippers with customized hardware. This limitation restricts the applicability of these tools for robots with diverse gripper mechanisms. Additionally, variations in control latency and trajectory-tracking errors across robots, absent in human demonstrations, complicate policy transfer between embodiments. Chi et al. [37] addressed this by compensating for control and observation latencies, while Song et al. [202] used fine-tuning through

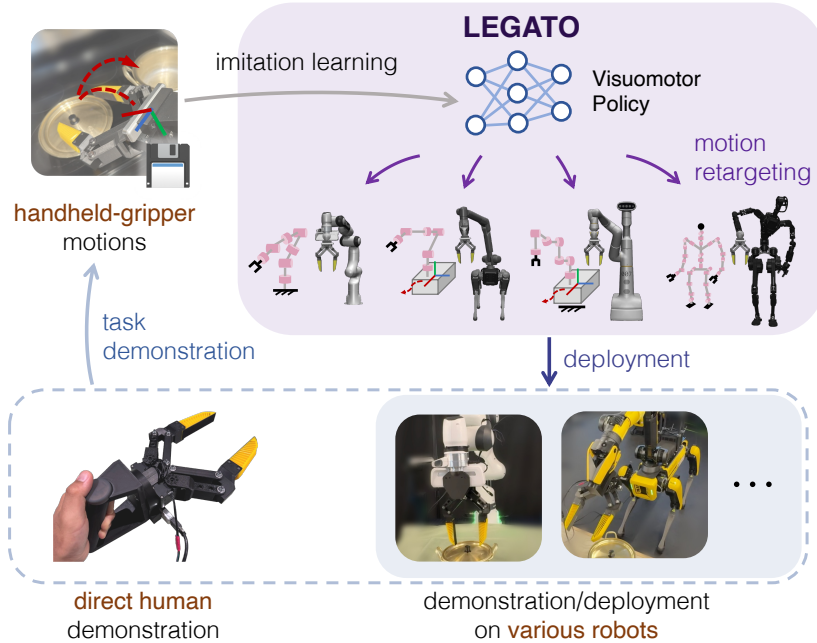


Figure 2.1: Overview of LEGATO. LEGATO addresses the challenge of transferring visuomotor skills across diverse robot embodiments. We present a cross-embodiment imitation learning framework using a versatile handheld grasping tool that ensures consistent physical interactions across different embodiments. Visuomotor policies trained on demonstrations by humans or teleoperated robots using the tool can be deployed across various robots equipped with the same gripper. Motion retargeting enables the execution of trajectories on different robots without requiring robot-specific training data.

trial and error on the target robot system. However, these strategies are difficult to generalize across diverse robots, as control latency varies between platforms due to differences in hardware and controllers.

Our key idea to address hardware differences across robots is to integrate a handheld grasping tool that can be shared across various robot embodiments for performing the same tasks (see Figure 2.1). We name our method LEGATO (Learning with a Handheld Grasping Tool). In our method, the LEGATO Gripper, a custom-designed handheld gripper, acts as a versatile and adaptive tool, representing tasks through its trajectories and grasping actions to ensure consistency across embodiments during both demonstration and deployment. This handheld gripper—compatible with various robot grippers—enables a single visuomotor policy to be applied across

diverse robot systems without requiring modifications to the original systems. Sharing the gripper across robots and directly manipulating human demonstrators ensures consistent, actively actuated grasping actions. In contrast, other data collection methods often create discrepancies in grasping actions because demonstrations rely on passive actuation by demonstrators, while robots require active actuation during deployment.

Our framework incorporates low-level motion retargeting through inverse kinematics (IK) optimization, tailored to each robot, along with a high-level transferable visuomotor policy (see Figure 2.2). Commands from the visuomotor policy for the handheld gripper are converted into whole-body motions using IK, enabling adaptation across robotic systems with only kinematic information, thereby avoiding extensive robot-specific training. While the IK optimization adapts gripper motions to each robot, variations in hardware and kinematics introduce differences in control latency and errors. To address this, we incorporate regularization on the gripper’s trajectories in a motion-invariant space during training, preventing bias toward the demonstration embodiment and effectively learning motor skills. This approach ensures that gripper trajectories from the visuomotor policy can be consistently translated into whole-body robot motions, regardless of control latency and IK response differences. Together, these components enable the learned visuomotor policy in LEGATO to be effectively transferred across diverse robotic systems.

We validate our approach through simulation and real-robot experiments, demonstrating its cross-embodiment transferability. In simulation, visuomotor policies trained on human demonstrations are successfully deployed across various embodiments, including a tabletop manipulator, a wheeled robot, a quadruped, and a humanoid. We further demonstrate the reusability of demonstration data by transferring a policy trained on one robot to another. In real-world tabletop manipulator setups, our method achieves a 72% success rate in complex manipulation tasks through policy transfer from direct human demonstrations.

2.2 Related Work

Cross-Embodiment Skill Learning Cross-embodiment policy transfer enables collecting demonstration data from easier or less costly embodiments and reusing it across different robot embodiments. One approach involves learning from videos of direct human demonstrations, which has been extensively explored in earlier works [133, 197, 200, 221, 235, 245]. However, robot deployment is limited by its reliance on third-person visual observations and the absence of real-world physical interaction data. Another approach integrates tools specifically designed for demonstration collection. One method involves leader-follower systems, which have been shown to be successful [57, 116]. However, this method requires designing leader hardware with a kinematic structure identical to a target robot, enabling its joint states to be directly mapped onto the robot. Alternatively, recent works use handheld tools to record on-hand visual observations and corresponding motions from demonstrations [37, 72, 165, 202, 241]. While these approaches simplify tool design, they still require a system-specific tool or modifications to a robot’s original hardware to align the end-effector’s physical interactions and the recorded visual observations. Unlike prior methods, we aim to generalize cross-embodiment learning by incorporating an adaptable handheld gripper and flexible kinematics-based motion retargeting. As a result, our learned policies are deployable across various types of robots without requiring robot-specific hardware for demonstrations.

Whole-body Motion Retargeting Motion retargeting enables the generation of practical motions across different embodiments. A substantial body of research exists on motion retargeting between similar morphologies, such as from humans to humanoid robots, employing either model-based methods [15, 46, 169, 209] or data-driven approaches [63, 78, 172]. In contrast, translating human movements into equivalent motions for varying target morphologies presents a challenge due to the inherently ambiguous nature of the task. Recent studies have investigated motion retargeting across embodiments with diverse morphologies [117, 134, 237]. While

these studies have successfully demonstrated motion retargeting from humanoid to non-humanoid morphologies, they often rely on either embodiment-specific or task-specific models. Our method shares similarities with previous works [46, 169] in utilizing kinematic optimization for motion retargeting. Unlike these works, however, we use a lower-dimensional action space based on the motions of the handheld gripper. This approach enables flexible motion retargeting across diverse robot embodiments with varying morphologies.

Trajectory Representations To effectively transfer motions across diverse embodiments, motion primitives offer a practical solution. These primitives facilitate the assembly of elemental motions, as extensively explored in the literature [164, 185, 257]. However, their adaptability is inherently limited by design, making them less effective for novel tasks or environments. Recent advancements, such as encoder-decoder frameworks that project trajectories into a latent space, have significantly improved the learning of motion primitives [126, 159, 216]. Encoding motions into learned latent spaces facilitates the transfer of human motions to simulated humanoid models [143] and adapts them to robot embodiments with varying kinematic structures [237]. Nevertheless, these approaches often encounter challenges in achieving broad generalization across different scenarios. To address these challenges, our method employs motion representations as training regularization elements rather than directly using them for motion generation. Specifically, we utilize the Denavit-Hartenberg Bidirectional (DHB) invariant representation [127], which offers invariance to rotational-translational shifts and scaling. The regularization in this motion-invariant space ensures robust alignment with demonstrated trajectories, enhancing policy generalizability across various embodiments.

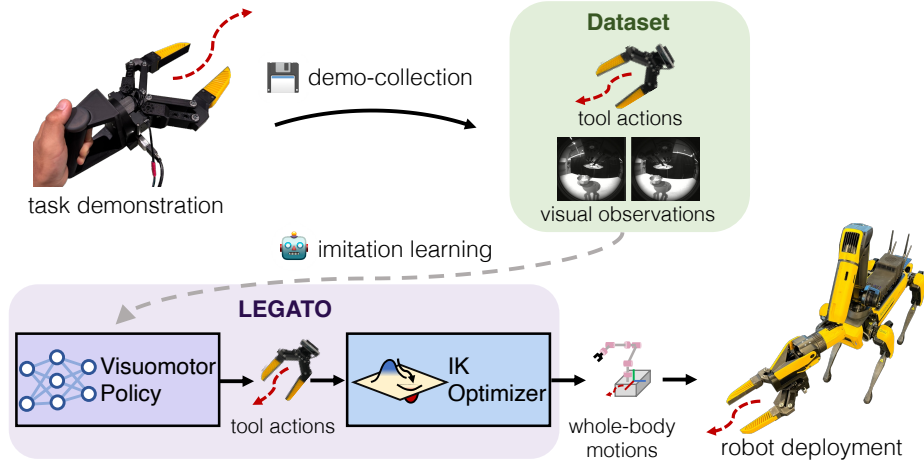


Figure 2.2: LEGATO’s cross-embodiment learning pipeline. During data collection, the LEGATO Gripper records its trajectories, grasping actions, and visual observations captured by its egocentric stereo camera. A visuomotor policy is then trained on these demonstrations through imitation learning. During deployment, the visuomotor policy’s outputs are retargeted to the robots’ whole-body motions through IK optimization.

2.3 Method

Here, we introduce LEGATO, a cross-embodiment imitation learning framework comprising a visuomotor policy at the high level and motion retargeting at the low level, as illustrated in Figure 2.2. The visuomotor policy is trained through imitation learning on task demonstrations, collected either from humans directly using the tool or from teleoperated robots holding the tool. The consistency of the action space of handheld gripper motions and egocentric visual observations across different robots enables deployment to various embodiments. The low-level motion retargeting realizes these gripper trajectories as whole-body motions across different robot platforms.

2.3.1 Problem Formulation

We model the problem of cross-embodiment manipulation as a discrete-time Markov Decision Process $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, R, \gamma, \rho_0)$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P}(\cdot|s, a)$ is the transition probability, $R(s, a, s')$ is the reward function,

$\gamma \in [0, 1)$ is the discount factor, and $\rho_0(\cdot)$ is the initial state distribution. Our objective is to learn a closed-loop visuomotor policy $\pi(a_t|s_t)$ that maximizes the expected return $\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1})]$ across different robot systems. In our problem, $\mathcal{S}$ is the space of visual observations captured by the handheld gripper’s egocentric cameras, the robot’s proprioceptive feedback, and previous actions, while $\mathcal{A}$ is the space of the robot’s joint-space commands. These two spaces vary across different robot systems, reflecting the diversity in sensory capabilities and actuation mechanisms inherent to each robot platform. $R(s, a, s')$ is the reward function designed for the manipulation task, and π is a closed-loop policy deployed on the robot.

To handle the complexity of visuomotor skills and ensure that the policy π is deployable across various robot systems, we decompose the policy into a two-level hierarchy. At the high level is a cross-embodiment visuomotor policy π_H , implemented as neural networks that compute target motion trajectories of the handheld gripper u . We train π_H through imitation learning with demonstrations collected by manipulating the handheld gripper. At the low level, we use a whole-body motion optimizer π_L , which determines target joint configurations to follow the trajectories u established by π_H through IK. This eliminates the need for additional training on target robot systems. With this hierarchical structure, the entire policy can be represented as $\pi(a_t|s_t) = \pi_L(a_t|s_t, u_t)\pi_H(u_t|s_t)$. This hierarchical decomposition may also provide a way to handle complex robot dynamics by formulating the learning problem in a more compact domain, for example, by mapping from joint-space actions to task-space commands. Further discussion is provided in Chapters 4 and 5.

2.3.2 Actions Based on the Handheld Gripper

We incorporate the LEGATO Gripper, which can be shared across different robot systems. Inspired by Noguchi et al. [158], each robot uses its own gripper to hold the handheld gripper, integrating it as part of the embodiment. The handheld gripper maintains a consistent shape and viewpoint, unifying visual observation and action spaces across embodiments. This reduces the complexity of the cross-embodiment

problem, streamlining the mapping of handheld-gripper motions and coordinating whole-body motions to execute them.

We define the action space as the differential pose in $SE(3)$ between consecutive time steps. This action space is suitable for generalizable whole-body manipulation on floating-base robots, as it eliminates reliance on a fixed reference frame. Actions are thus represented in the current handheld-gripper frame. Differential pose actions, sampled from a Gaussian Mixture Model [224], capture the multimodal nature of human demonstrations and essential motion information within the motion-invariant space, as described in Section 2.3.4.

2.3.3 Whole-body Motion Retargeting

In this section, we detail our approach for mapping the trajectory actions of the handheld gripper u_t into whole-body robot motions a_t . We employ an optimization-based IK method to handle the constrained IK problem. This enables our motion optimizer π_L to map commands effectively to robot motions by leveraging degree-of-freedom (DOF) redundancy while respecting actuation bounds and other constraints. Our approach addresses kinematic differences, constraints, and the diversity in DOFs across robot embodiments by using kinematic redundancy to satisfy joint and other constraints during deployment, without requiring additional robot-specific demonstrations.

The motion retargeting formulation is expressed as the following quadratic programming (QP) problem:

$$\begin{aligned}
& \min_{a_t} \sum \frac{1}{2} a_t^\top H a_t \\
& \text{subject to} \\
& J_i(q_t) a_t = \dot{x}_i^{\text{des}}, \\
& \underline{L}_i \leq C_i a_t \leq \bar{L}_i,
\end{aligned} \tag{2.1}$$

where $\underline{L}_i$, $\bar{L}_i$ and C_i define the velocity-level constraints for joint positions q , velocities $\dot{q}$, and accelerations $\ddot{q}$, along with other Cartesian limits (e.g., virtual walls and

collision distance bounds). H is a positive-definite weighting matrix. $\dot{x}_i^{\text{des}}$ represents the desired velocities at the prioritized tasks, formulated in the following order:

$$\begin{aligned} J_1 &= J_{\text{grip}}, \quad \dot{x}_1^{\text{des}} = K_{\text{grip}}(x_{\text{des}} - x(q)), \\ J_2 &= I_{n_j}, \quad \dot{x}_2^{\text{des}} = K_{\text{bias}}(q_{\text{bias}} - q), \end{aligned} \tag{2.2}$$

where J_{grip} and I_{n_j} represent the Jacobians for gripper pose control and maintaining the configuration bias q_{bias} of n_j DOFs, respectively, with the control gains K_{grip} and K_{bias} . x_{des} represents the target pose of the handheld gripper, as determined by the visuomotor policy outputs u_t .

To solve the QP problem with the hierarchical priorities and the inequality constraints, we utilize the extended Saturation in the Null Space algorithm [60]. The constrained optimization problem of Equation 2.1 is formulated as follows:

$$\begin{aligned} &\text{maximize} \quad \sum_i c_i \\ &\text{subject to} \\ &\quad J_i(q_t)a_t = c_i\dot{x}_i^{\text{des}}, \\ &\quad \underline{L}_i \leq C_i a_t \leq \overline{L}_i. \end{aligned} \tag{2.3}$$

Here, c_i represents scaling factors within the range [0,1] that allow for scaling speed of the prioritized tasks defined in Equation 2.2, while accurately tracking their trajectories. This optimization problem, maximizing the sum of scaling factors, yields the closest motion mapping while meeting all constraints. Our framework ensures that robot movements not only follow the prescribed hierarchy but also adhere to the constraints, guaranteeing flexible task execution and robust motion generalization.

2.3.4 Training of Visuomotor Policies

Task demonstrations can be performed on any embodiment capable of executing the tasks within its joint space. The collected demonstration dataset $\mathcal{D}$ consists of state-action pairs $\mathcal{D} = \{(s_i, u_i)\}_{i=1}^N$. Here, N represents the total number of data

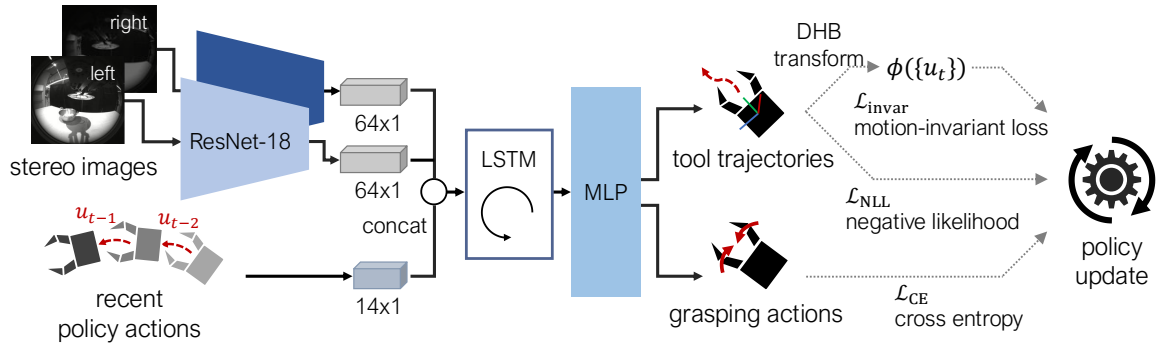


Figure 2.3: High-level visuomotor policy architecture. The trained policies generate desired handheld-gripper trajectories and grasping actions u_t at 10 Hz from egocentric stereo camera observations and previous policy actions. These action and observation spaces, defined in the handheld-gripper frame, remain consistent across various robot platforms. To learn actions on handheld-gripper trajectories, we apply two action losses: the negative log-likelihood loss $\mathcal{L}_{NLL}$ for the distribution in SE(3) and the L2 loss $\mathcal{L}_{invar}$ in the DHB motion-invariant space. The grasping actions are trained using the cross-entropy loss $\mathcal{L}_{CE}$.

points. The observations s_i comprise stereo images from the handheld gripper’s on-board camera and a history buffer of previous actions $u_{k:k=t-T:t-1}$. The visual observations are provided as stereo grayscale images with a resolution of 128×128 pixels. The demonstration commands u_i include the subsequent setpoints for the handheld gripper (6D) and the grasping actions (1D).

We train our policy π_H using a deep imitation learning algorithm, specifically using a behavioral cloning policy with LSTM networks [83, 148], as shown in Figure 2.3. The visuomotor policy employs two separate ResNet18-based image encoders [76], trained end-to-end. The encoded features are flattened and processed by two-layer LSTM networks, each with 400 hidden units. The policy outputs are generated by a three-layer multi-layer perceptron (MLP), with each layer containing 2048 hidden units. For hand trajectories, the policy outputs Gaussian Mixture Model (GMM) parameters to determine the next target pose based on spatial and rotational differences from the previous frame, using a 5-mode GMM. Grasping actions are provided as binary classifications for opening and closing the gripper.

For imitation learning, we employ behavioral cloning with the following train-

ing loss:

$$\mathcal{L} = \mathcal{L}_{\text{NLL}} + \mathcal{L}_{\text{invar}} + \mathcal{L}_{\text{CE}}. \quad (2.4)$$

Here, $\mathcal{L}_{\text{NLL}}$ and $\mathcal{L}_{\text{invar}}$, associated with the trajectories of the handheld gripper, represent the negative log-likelihood loss and the regularization term in the motion-invariant space, respectively. Additionally, $\mathcal{L}_{\text{CE}}$ denotes the cross-entropy loss for binary grasping actions. The term $\mathcal{L}_{\text{invar}}$ is calculated as the L2 loss in the motion-invariant space, specifically using the DHB invariant transformation [127], $\phi(\cdot)$. This transformation is applied to the differential poses of the gripper in SE(3) from the policy outputs u_t and the demonstrations $\hat{u}_t$:

$$\mathcal{L}_{\text{invar}} = \sum \|\phi(u_t, \{\hat{u}_k\}_{k=t-T}^{t-1}) - \phi(\{\hat{u}_k\}_{k=t-T}^t)\|^2. \quad (2.5)$$

Incorporating $\mathcal{L}_{\text{invar}}$ into the training loss influences the distribution of handheld-gripper motions, preventing the policy from being biased by embodiment-specific properties like tracking errors and control latency. Unlike SE(3), these invariants represent motion by breaking it down into magnitude and directional changes, unaffected by embodiment-specific factors such as viewpoints, reference frames, pose offsets, or scales. Leveraging motion invariance ensures the policy captures essential motion information from demonstrations without bias toward specific embodiments. This is critical for cross-embodiment learning, as it makes the policy robust to domain mismatches across different robot embodiments.

2.3.5 Motion-Invariant Regularization Loss

The motion-invariant transform $\phi(\cdot)$, used to compute $\mathcal{L}_{\text{invar}}$ in Equation 2.5, follows the DHB motion-invariant framework [127]. Given trajectories $\{u_k\}_{k=t-T}^t$ with $T \geq 2$, we compute the relative positions p_k and orientations r_k of the gripper with respect to the initial frame at $t - T$, where p_{t-T} and r_{t-T} are at the origin.

The differences $\Delta \mathbf{p}_k = \mathbf{p}_{k+1} - \mathbf{p}_k$ and $\Delta \mathbf{r}_k = \mathbf{r}_{k+1} - \mathbf{r}_k$ represent the linear and angular trajectory changes between $k + 1$ and k . The initial linear frames are defined

as:

$$\hat{\mathbf{x}}_{p,k} = \frac{\Delta \mathbf{p}_k}{\|\Delta \mathbf{p}_k\|}, \quad \hat{\mathbf{y}}_{p,k} = \frac{\hat{\mathbf{x}}_{p,k} \times \hat{\mathbf{x}}_{p,k+1}}{\|\hat{\mathbf{x}}_{p,k} \times \hat{\mathbf{x}}_{p,k+1}\|}, \quad \hat{\mathbf{z}}_{p,k} = \hat{\mathbf{x}}_{p,k} \times \hat{\mathbf{y}}_{p,k}.$$

Similarly, the initial angular frames are:

$$\hat{\mathbf{x}}_{r,k} = \frac{\Delta \mathbf{r}_k}{\|\Delta \mathbf{r}_k\|}, \quad \hat{\mathbf{y}}_{r,k} = \frac{\hat{\mathbf{x}}_{r,k} \times \hat{\mathbf{x}}_{r,k+1}}{\|\hat{\mathbf{x}}_{r,k} \times \hat{\mathbf{x}}_{r,k+1}\|}, \quad \hat{\mathbf{z}}_{r,k} = \hat{\mathbf{x}}_{r,k} \times \hat{\mathbf{y}}_{r,k}.$$

The directions of the axes in both frames are chosen to prevent discontinuities across time steps.

In the DHB transformation, the motion of a rigid body is separated into position and orientation components using two frames. Two invariants are the norms of the relative positions and orientations between these frames: $m_{p,k} = \|\Delta \mathbf{p}_k\|$, $m_{r,k} = \|\Delta \mathbf{r}_k\|$. These invariants, m_p and m_r , describe the translation of the linear and angular frames. Four additional values describe their rotation:

$$\begin{aligned} \theta_{p,k}^1 &= \arctan \left(\frac{\hat{\mathbf{x}}_{p,k} \times \hat{\mathbf{x}}_{p,k+1}}{\hat{\mathbf{x}}_{p,k} \cdot \hat{\mathbf{x}}_{p,k+1}} \cdot \hat{\mathbf{y}}_{p,k} \right), \\ \theta_{p,k}^2 &= \arctan \left(\frac{\hat{\mathbf{y}}_{p,k} \times \hat{\mathbf{y}}_{p,k+1}}{\hat{\mathbf{y}}_{p,k} \cdot \hat{\mathbf{y}}_{p,k+1}} \cdot \hat{\mathbf{x}}_{p,k+1} \right), \\ \theta_{r,k}^1 &= \arctan \left(\frac{\hat{\mathbf{x}}_{r,k} \times \hat{\mathbf{x}}_{r,k+1}}{\hat{\mathbf{x}}_{r,k} \cdot \hat{\mathbf{x}}_{r,k+1}} \cdot \hat{\mathbf{y}}_{r,k} \right), \\ \theta_{r,k}^2 &= \arctan \left(\frac{\hat{\mathbf{y}}_{r,k} \times \hat{\mathbf{y}}_{r,k+1}}{\hat{\mathbf{y}}_{r,k} \cdot \hat{\mathbf{y}}_{r,k+1}} \cdot \hat{\mathbf{x}}_{r,k+1} \right). \end{aligned}$$

This process produces the linear and angular invariant values $(m_{p,k}, \theta_{p,k}^1, \theta_{p,k}^2)$ and $(m_{r,k}, \theta_{r,k}^1, \theta_{r,k}^2)$, as established in the original work.

To ensure continuity, the computed frame rotations are transformed using $\sin(\cdot)$ and $\sin(2\cdot)$. The final transformation applied in our regularization loss thus yields 10 variables of length $T - 1$:

$$\phi(\{u_k\}_{k=t-T}^t) = \left\{ \begin{bmatrix} m_{p,k} \\ \sin(\theta_{p,k}^1) \\ \sin(2\theta_{p,k}^1) \\ \sin(\theta_{p,k}^2) \\ \sin(2\theta_{p,k}^2) \\ m_{r,k} \\ \sin(\theta_{r,k}^1) \\ \sin(2\theta_{r,k}^1) \\ \sin(\theta_{r,k}^2) \\ \sin(2\theta_{r,k}^2) \end{bmatrix} \right\}_{k=t-T}^{t-2}.$$

When computing $\mathcal{L}_{\text{invar}}$, we transform two types of trajectories: 1) $\phi(\{\hat{u}_k\}_{k=t-T}^t)$, the transformed values from the demonstration trajectories, and 2) $\phi(u_t, \{\hat{u}_k\}_{k=t-T}^{t-1})$, the transformed values from the given previous trajectories $\{\hat{u}_k\}_{k=t-T}^{t-1}$ and the predicted target u_t at time t . By calculating the L2 loss between these two transformed values and using it as a training loss, the predicted trajectories u_t are aligned with the demonstration trajectories in the motion-invariant space, given $\{\hat{u}_k\}_{k=t-T}^{t-1}$.

2.4 Experiments

In this section, we demonstrate the feasibility and effectiveness of LEGATO for cross-embodiment transfer of visuomotor policies, both in MuJoCo simulation [212] and real-world settings. Leveraging the scalability and ease of simulation environments, we employ them to investigate the following research questions: 1) How does the regularization in the motion-invariant space impact the training of cross-embodiment policies? 2) How do differences in morphology and controllability affect the task capacity of robot embodiments?

2.4.1 LEGATO Gripper

The LEGATO Gripper design described in Section 2.3.2 is implemented on real hardware. It is designed for intuitive use in both direct human demonstrations and robot operations, as illustrated in Figure 2.4. During human demonstrations,

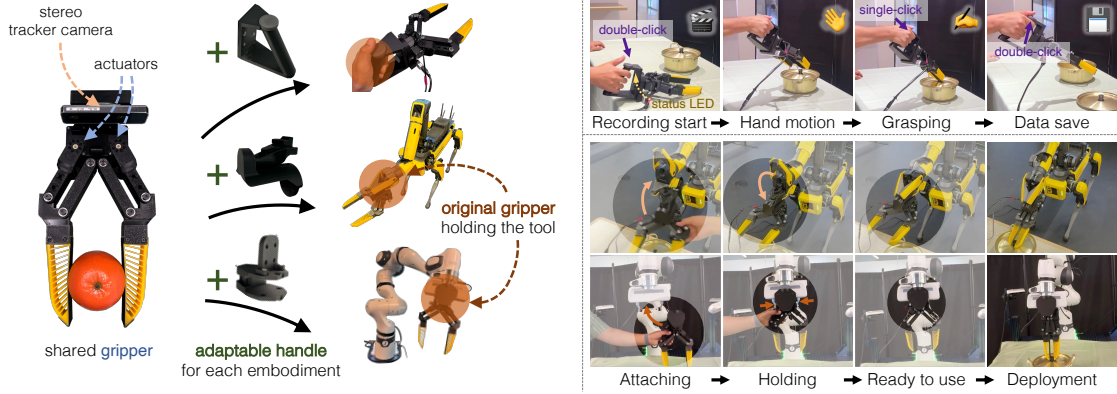


Figure 2.4: LEGATO Gripper design. The LEGATO Gripper is designed for both human demonstration collection and robot deployment. **(Left)** It features a shared actuated gripper with adaptable handles, ensuring reliable human handling and consistent grasping across robots while minimizing components. **(Right top)** A human demonstrator can directly perform tasks by carrying the LEGATO Gripper in hand. The design includes a simple yet intuitive button interface with a status LED, allowing data recording to start and end with a double-click and grasping actions to trigger with a single click. **(Right bottom)** The LEGATO Gripper is easily installed on various robots, securely held by their original grippers, and is ready for immediate use.

a user carries the LEGATO Gripper and controls the grasping actions. Robots, on the other hand, can hold the tool with their original grippers without needing any hardware modifications.

To enable easy attachment and replacement, the design is modularized, particularly the handle parts that robots grip, as presented in Figure 2.4. Thus, only the handle parts need replacement for different robots, allowing all other core components to be shared across robots. These handle parts can be easily designed using CAD models of target robots provided by manufacturers, ensuring adaptability to diverse robot systems. The LEGATO Gripper features two pairs of parallel four-bar linkage mechanisms, each actuated by one DOF, enabling a wider and more flexible range of opening distances. The tool employs fingertips of compliant fin ray mechanisms for high compliance and adaptability during contact. Made from 3D-printed parts, with TPU 95A for the fingertips and PLA for other components, the LEGATO Gripper is significantly lighter than typical commercial grippers. These key features facilitate the concept of integrating the tool as part of the embodiment, even under robots’

limited payload capacities, while supporting generalizable grasping actions.

The LEGATO Gripper is equipped with a Realsense T265 [7] camera, with fisheye stereo cameras and an IMU for streaming visual observations and estimating the handheld gripper’s motions via visual odometry. During demonstrations, both stereo images and visual odometry data are recorded to form observation-action pairs. In contrast, during robot deployment, only stereo images are streamed.

In simulation, the mechanism and geometry of the real hardware are modeled. Visual observations are emulated by adapting the properties of the tracker camera used in the hardware design. The LEGATO Gripper is attached to each robot’s original gripper with an offset.

2.4.2 Experimental Setup

We designed the following three realistic manipulation tasks to study the cross-embodiment deployment of our method for both simulation and real-robot systems.

- ***Closing the lid:*** A robot grasps the lid and places it on the pot within reach. This task requires precise manipulation to accurately close the lid.
- ***Cup shelving:*** A robot places the cup into the shelf. In addition to requiring a large workspace, this task involves complex collision-free motion to position the cup against the shelf’s non-convex shapes.
- ***Ladle reorganization:*** A robot picks up the ladle from the pot and places it into the utensil organizer. This task involves complex manipulation and handling objects potentially out of view due to limited visibility and occlusion.

A task is considered successful if a robot accomplishes the designated goals within a specific time limit. Across all baselines and tasks, the initial states of robots are consistent, and the initial poses of objects, other than the table, are uniformly randomized.

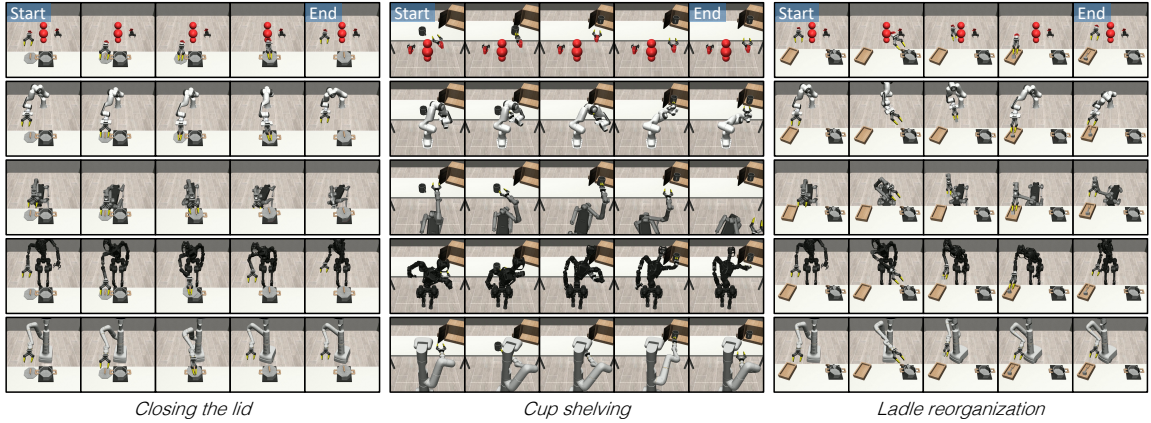


Figure 2.5: Timelapse of deploying LEGATO in simulation. We trained visuomotor policies using demonstrations from the *Abstract* embodiment and deployed them on robots with diverse morphologies, from the top: *Abstract*, *Panda*, *Spot*, *GR-1*, and *Google Robot*. The timelapse of deploying these policies reveals consistent time steps. The tracking performance of the IK motion retargeting varies with morphology, leading to domain gaps across embodiments. Despite these challenges, LEGATO achieves successful deployment on various robots.

2.4.3 Quantitative Evaluation in Simulation

In these experiments, we demonstrate the effectiveness of our domain-transfer method across different embodiments. To systematically evaluate our method, we use embodiments representing various kinematic morphologies, as shown in Figure 2.1: Franka Emika *Panda* [73], Boston Dynamics *Spot* [3], Fourier *GR-1* [5], and *Google Robot* [80].

- ***Abstract*:** an idealized embodiment designed for simulation that can manipulate hands along continuous trajectories without speed or workspace limitations. Human motion commands are directly mapped to the simulation, replicating direct human demonstration in real-world settings.
- ***Panda*:** a 7-DOF tabletop manipulator, used to demonstrate the impact of redundant DOFs on motion retargeting compared to the robots listed below.
- ***Spot*:** a quadrupedal robot with 6 DOFs in its arm and 6 DOFs in its body pose. To demonstrate achieving an extensive workspace through whole-body

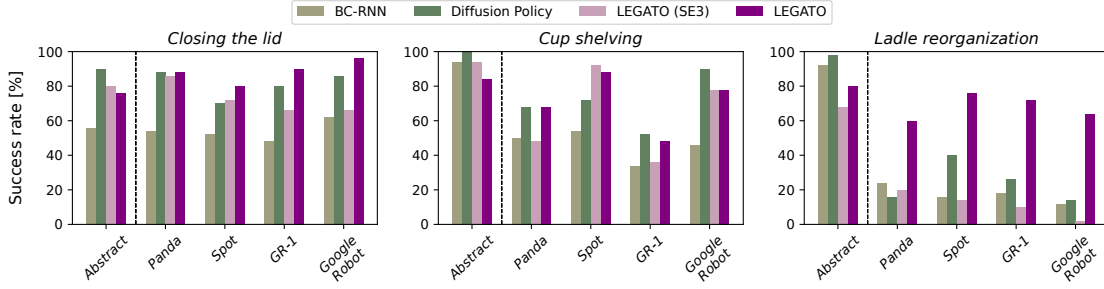


Figure 2.6: Quantitative results in simulation. We report success rates on 50 trials of our LEGATO policies compared to baselines. On average, LEGATO outperforms all other methods in cross-embodiment deployment by 28.9%, 10.5%, and 21.1%, compared to BC-RNN, Diffusion Policy, and LEGATO (SE3), respectively. Notably, unlike the baselines that achieve high success rates only on specific robot embodiments, typically the *Abstract* embodiment used for training, LEGATO demonstrates consistent success rates across different embodiments.

motion alone, the robot’s locomotion is not considered; instead, the leg joints track the robot’s body within a limited SE(3) range.

- **GR-1:** a humanoid robot with 7 DOFs per arm and 3 DOFs each for the head and torso. Similar to *Spot*, the robot’s locomotion is not considered, and the leg joints are fixed. This demonstrates the application of our method to highly redundant DOF systems.
- **Google Robot:** a wheeled mobile robot with 7 DOFs in its arm and 3 DOFs in planar base motion, used to show how handheld-gripper trajectories map to mobile manipulation.

We focus on demonstrations from the *Abstract* embodiment, where commands from human demonstrators are directly mapped, enabling task performance without being affected by tracking errors that occur in robots. The trained policies are then transferred directly to the robots without any adaptation, using the pre-defined IK optimizer, as shown in Figure 2.5.

Figure 2.6 reports quantitative evaluations of our simulated tasks, comparing our model with the following baselines.

- **BC-RNN:** a baseline that uses recurrent neural networks to learn manipulation skills from teleoperated demonstrations, as introduced by Mandlekar et al. [148]. It employs flat GMM outputs and is trained with the negative log-likelihood loss.
- **Diffusion Policy:** a baseline that employs a receding-horizon visuomotor policy based on a diffusion model, as introduced by Chi et al. [36]. In particular, we adapt the Velocity Diffusion Policy. Due to the requirement for precise tracking and state estimation of the end-effector pose with respect to the global frame, the Position Diffusion Policy is not applicable in our settings.
- **LEGATO (SE3):** a variant of our final model that excludes the motion-invariant regularization loss $\mathcal{L}_{\text{invar}}$ from the training loss, with all other components unchanged. This variant is designed to evaluate the impact of the regularization in the motion-invariant space on cross-embodiment deployment.

All baselines are trained on the same dataset of 150 task demonstrations using the *Abstract* embodiment with identical policy inputs, as shown in Figure 2.3. They output handheld gripper trajectories realized by the same IK optimizer, using consistent parameters within a single robot.

As shown in Figure 2.6, LEGATO outperforms the three baseline methods across various robot embodiments. The success rates for the *Abstract* embodiment indicate that all baselines are well-trained within that domain. However, the higher success rates of LEGATO (SE3) and Diffusion Policy, which uses the state-of-the-art diffusion model and performs best in the *Abstract* domain, are followed by a notable decline when applied to other robots. This suggests that exclusive training in SE(3) may introduce domain bias toward the training embodiment. In contrast, the performance improvement of LEGATO across diverse embodiments suggests that its regularization in the motion-invariant space enhances policy robustness across different robot domains, even with a simple model architecture. In addition, LEGATO

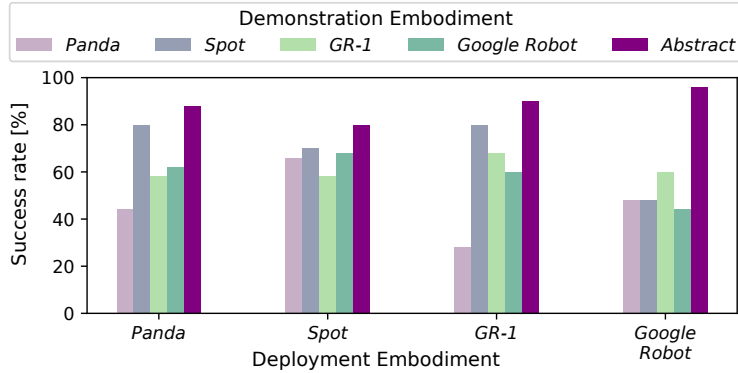


Figure 2.7: Simulation evaluation across different demonstration embodiments. We show success rate changes on 50 trials from policies trained on demonstrations of different embodiments.

and the baselines show reduced performance overall in the *Cup shelving* task with the *Panda* and *GR-1* robots, as well as in the *Ladle reorganization* task with the *Panda* robot. This underscores the limitations of fixed-base robots for tasks requiring extensive motion, highlighting the need for whole-body manipulation.

2.4.4 Varying Demonstration Embodiments

We investigate how different embodiments used in demonstrations affect policy performance in simulation. We collected 150 trajectories by teleoperating each robot in Section 2.4.3 for the *Closing the lid* task and trained policies on these demonstrations. Figure 2.7 shows the success rates for deploying these policies across the robots. The policies trained on the *Abstract* embodiment performed best, indicating that joint controller latency and the IK motion retargeting affect demonstration quality. Among the policies trained on robot teleoperation, demonstrations on the *Spot* robot yield the highest success rates, whereas those on the *Panda* robot yield the lowest. This suggests that whole-body motion capability and redundancy influence task demonstration quality, as the *Spot* robot has full 6 DOFs in its body, while the *Panda* robot has less redundancy. Notably, policies trained on the *Panda* and *Google Robot* exhibit lower success rates when deployed on the robot used for demonstrations compared to other robots. This highlights that the trajectory tracking capabilities

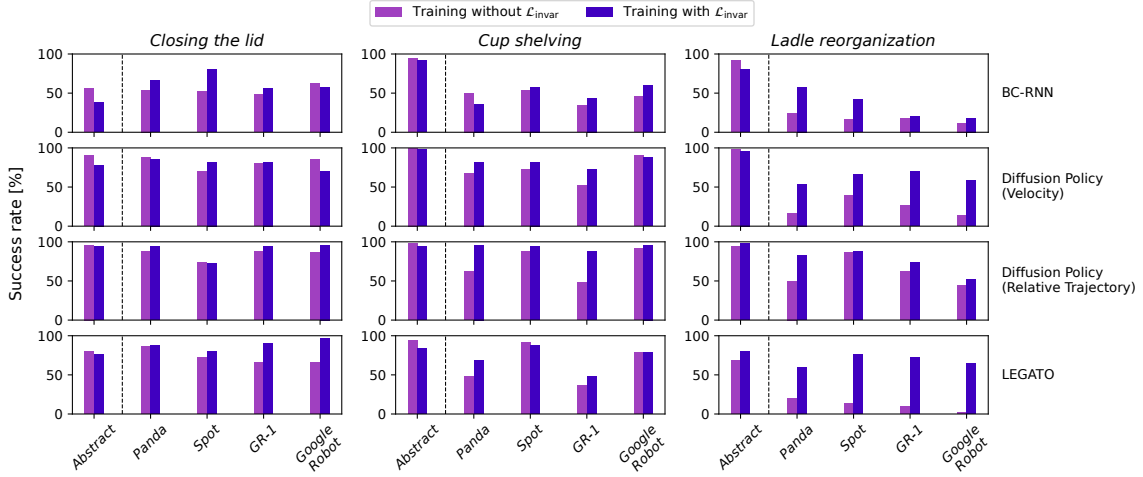


Figure 2.8: Impact of training with the motion-invariant regularization We evaluate the success rates of policies trained with and without the motion-invariant regularization, $\mathcal{L}_{invar}$, over 50 trials across varied architectures. Each policy is trained on demonstrations from the *Abstract* embodiment. While the success rates on the *Abstract* embodiment (used for demonstrations) decrease with the motion-invariant regularization, the success rates improve in cross-embodiment settings, where policies are deployed to other robots.

constrained by the deployment embodiments affect task complexity.

2.4.5 Impact of Training with the Motion-Invariant Regularization

We provide an additional ablation study on the complementary use of the motion-invariant regularization loss to enhance cross-embodiment transferability in policies, as shown in Figure 2.8. Specifically, we applied the motion-invariant regularization loss $\mathcal{L}_{invar}$, as described in Section 2.3.4, to BC-RNN [148], Diffusion Policy (Velocity) adapted from the Velocity Diffusion Policy introduced by Chi et al. [36], and Diffusion Policy (Relative Trajectory) adapted from the Diffusion Policy using relative end-effector trajectories as the action space in Universal Manipulation Interface [37]. Similar to the quantitative evaluation in Section 2.4.3, all baselines are trained on the same dataset of 150 task demonstrations using the *Abstract* embodiment. Additionally, the same IK optimizer with consistent parameters is used within a single robot.

As outlined in Equation 2.4 of Section 2.3.4, the motion-invariant regular-

ization loss is added to the original loss functions for each baseline. In BC-RNN, the motion-invariant regularization loss is added to the negative log-likelihood between the predicted and demonstration actions. For the Diffusion Policy baselines, the motion-invariant regularization loss is integrated with the L2-based DDPM loss during the denoising process. The motion-invariant regularization loss is adapted for the sequence of receding-horizon actions used in the Diffusion Policy baselines and is defined as:

$$\mathcal{L}_{\text{invar}} = \sum \|\phi(\{u_k\}_{k=t}^{t+P}, \{\hat{u}_k\}_{k=t-T}^{t-1}) - \phi(\{\hat{u}_k\}_{k=t-T}^{t+P})\|^2,$$

where P is the prediction horizon, and the sequence of actions of length $P + 1$ is transformed into the motion-invariant space.

Our findings indicate that incorporating the motion-invariant regularization during training generally reduces success rates when deploying policies on the *Abstract* embodiment but enhances performance in cross-embodiment settings with different robot embodiments, regardless of the neural network architectures. This highlights the applicability of leveraging motion invariance across various neural network architectures and its effectiveness for cross-embodiment learning.

2.4.6 Real-Robot Experiments

We aim to validate the robustness of our method against variations caused by controller limitations and sensor inaccuracies in real-world settings. We collected direct human demonstrations through the LEGATO Gripper to train policies and deployed them, particularly on the *Panda* robot, the most challenging due to its limited workspace at the IK motion retargeting, as shown in Section 2.4.3. During the evaluation of each task, the same visuomotor policy was attempted for 20 trials on the *Panda* robot. Our method succeeded in 16 trials of the *Closing the lid* task, 13 trials of the *Cup shelving* task, and 14 trials of the *Ladle reorganization* task, respectively (see Figure 2.9).

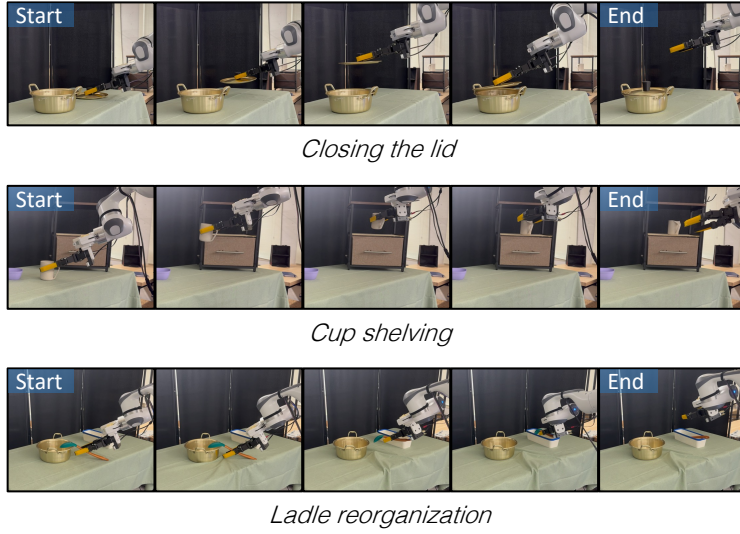


Figure 2.9: Timelapse of deploying LEGATO in the real-robot system. We trained visuomotor policies on direct human demonstrations and successfully deployed them on the *Panda* robot system. The timelapse of deploying these policies demonstrates consistent time intervals.

2.5 Conclusion

We introduced LEGATO, a cross-embodiment learning framework for transferring visuomotor skills across diverse robot morphologies. By using a handheld gripper for consistent observations and actions, our framework enables visuomotor policies to transfer across embodiments without hardware modifications. Handheld-gripper trajectories from the visuomotor policies are mapped to whole-body robot motions through IK optimization. Although trained on specific embodiments, regularization in a motion-invariant space allows these policies to adapt easily to different robots, managing variations in control latency and tracking errors. Our current focus is on whole-body manipulation, employing redundant DOFs for flexibility and an extended workspace through coordinated body movements, though our method is limited to non-walking scenarios. Future work will incorporate loco-manipulation, integrating manipulation with walking to enable legged robots to perform diverse tasks with larger workspaces. Additionally, while our method currently relies on designing handles specific to target robots, this can be addressed by identifying grasping loca-

tions to enable the use of universal handles across robots. As this work demonstrates, hardware design can play an active role in cross-embodiment learning rather than serving as a limitation, and our approach is applicable to a wide range of tools and applications beyond the LEGATO Gripper design.

Chapter 3: Enhancing Robot Capabilities through Sensor Embedding

3.1 Introduction

In Chapter 2, we discussed how shared hardware tools can provide a consistent interaction interface across diverse robots, enabling skills learned on one embodiment to transfer to others. Such a shared tool defines a capability envelope that standardizes how robots interact with objects and the environment. However, when the hardware interface is designed primarily for cross-embodiment transfer, its capabilities may be insufficient for tasks requiring specialized physical interactions, motivating the need to augment shared interfaces with task-specific capabilities. As a result, robotic tools and end-effectors must also be co-designed with task requirements to fully exploit the physical capabilities required for those interactions. By leveraging the idea that tools can augment the natural capabilities of robots, task-specific hardware can extend what robots are able to sense and control during manipulation.

One important example is delicate object manipulation. Humans regulate contact forces with high precision, applying just enough force to prevent slip while avoiding damage [68, 121, 228]. This capability is enabled by mechanoreceptors in the skin that provide real-time feedback on force and slip [100, 227]. In contrast, although robots can perform many manipulation tasks with simple parallel grippers [20, 36, 160, 254], these systems typically lack direct grasp feedback. They rely primarily on vision and coarse open-close control, limiting their ability to regulate contact forces. This gap highlights the need to integrate tactile sensing into end-effectors for precise force modulation. Embedding tactile sensing into shared handheld grippers offers one practical approach, providing a sense of touch while preserving a unified interface across robots.

To improve force regulation and enable reliable contact feedback, prior work

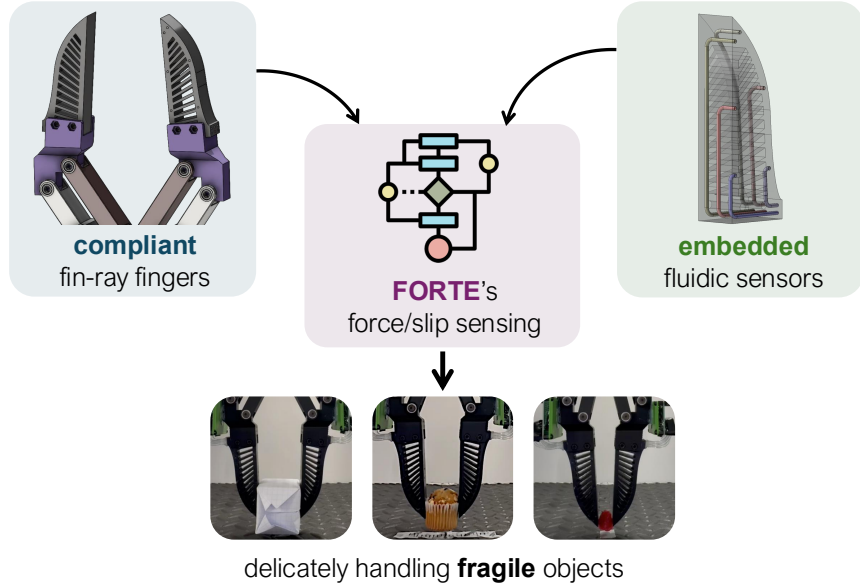


Figure 3.1: Overview of FORTE. FORTE delicately manipulates fragile objects by leveraging passive compliance and tactile feedback. This is achieved by integrating compliant fin-ray fingers and embedded fluidic sensors with algorithms for force estimation and slip detection.

has explored enhancing rigid parallel grippers along two main directions: introducing compliance and incorporating tactile sensing. Compliant designs, such as fin-ray fingers [42], passively adapt to object geometry and reduce damage, but offer limited control over interaction forces [42, 49]. Tactile sensing enables feedback for force regulation and slip detection, but existing solutions often rely on expensive, fragile sensors [139, 223, 236, 247] or complex custom fabrication and signal processing [54, 75, 162, 175, 198]. A simple end-effector that integrates compliance with robust, embedded tactile sensing is therefore key to enabling human-like delicate manipulation while remaining compatible with existing robotic systems.

To address this challenge, we introduce FORTE (Fragile Object Grasping with Tactile Sensing), a tactile sensing system designed to regulate grasping force and detect slip during the manipulation of delicate objects. The system combines two key components: 3D-printed fin-ray compliant fingers and fluidic innervation, as illustrated in Figure 3.1. Rather than attaching external tactile sensors, FORTE integrates air channels directly within the compliant finger structure. As the finger

deforms during contact, these channels deform as well, generating pressure signals that are measured using off-the-shelf transducers. Because these signals reflect the internal deformation of the gripper, FORTE enables accurate force estimation. The fast response and high temporal resolution of the sensors further allow slip detection through frequency-based analysis of the pressure signal.

We design and fabricate compliant fingers with embedded air channels that enable tactile sensing through internal structural deformation. The sensors detect contact events at a sampling rate of 2 kHz, enabling fast and responsive sensing during manipulation. Using this mechanism, FORTE achieves accurate force estimation in the range of 0–8 N with an error of ± 0.19 N. We further develop an analytical slip detection algorithm based on the frequency characteristics of the tactile signal, achieving an F1 score of 0.91 across 31 objects, including 29 previously unseen items. Finally, we evaluate FORTE through single-trial grasping experiments on 31 items spanning fragile, slippery, and deformable objects. Across 310 grasp trials, the system achieves a 91.9% success rate.

3.2 Related Work

To integrate tactile sensing into a compliant gripper, the sensing mechanism must avoid introducing additional stiffness. Sensors with rigid backing, for example, can create localized stiff regions that compromise finger compliance. Vision-based tactile sensing offers one approach that preserves softness by combining a deformable elastomer surface with an internal camera. By tracking surface deformation, these systems provide high-resolution measurements of contact interactions [139, 223, 247]. However, the camera hardware increases structural complexity, and the elastomer surface can degrade or tear under repeated or high-force contact [253].

Another approach uses acoustic sensing, where microphones are embedded beneath the contact surface of the finger. The resulting acoustic signals capture high-frequency vibrations generated during contact, enabling estimation of contact

force and location [262], identification of material properties [220], and learning of contact-rich manipulation policies [141]. Despite these capabilities, acoustic sensing is often sensitive to environmental noise and requires complex signal interpretation when interactions occur through compliant materials.

Barometric sensing provides an alternative strategy by incorporating internal air chambers within soft fingers. Because air is compressible, deformation of the finger changes the pressure inside the chamber, producing measurable signals that reflect contact interactions. Prior work has shown that such sensors can detect dynamic events such as sliding, measure contact forces, and localize contact points [38, 70, 77]. Compared with vision-based systems, barometric sensing avoids fragile elastomer skins and typically requires simpler signal processing than acoustic approaches, but it often relies on fabrication processes that are difficult to scale or reproduce.

In this work, we apply fluidic innervation, a form of barometric sensing, to a fin-ray compliant gripper. Fluidic innervation simplifies fabrication by enabling the finger structure and internal air channels to be produced simultaneously through 3D printing. This approach supports straightforward manufacturing while providing reliable force estimation and slip detection capabilities [213, 248].

3.3 Method

We present a sensorized compliant gripper that enables high-frequency tactile sensing for contact-rich manipulation. Our approach combines hardware design with lightweight inference algorithms to estimate interaction forces and detect slip in real time. Specifically, we integrate fluidic-innervated fin-ray fingers to capture deformation signals, which are processed to estimate grip force and identify slip events through temporal and spectral features. This design enables robust sensing under minimal normal force while remaining computationally efficient and broadly applicable across manipulation tasks.

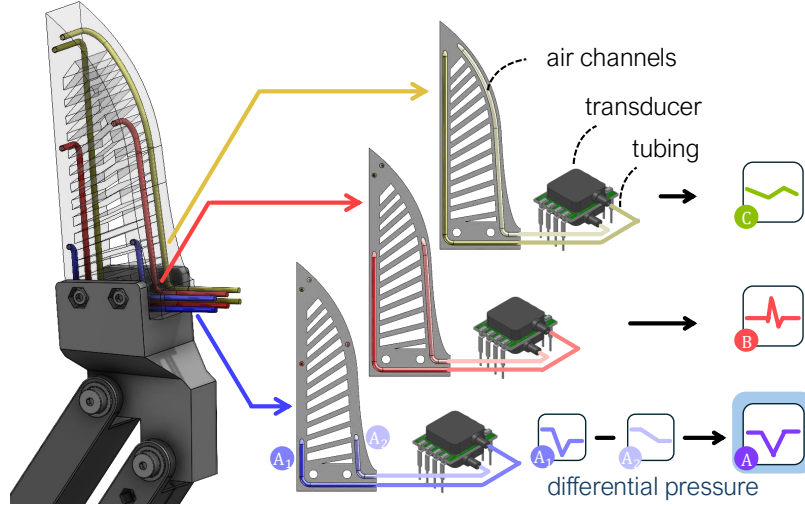


Figure 3.2: Hardware design of FORTE. Fin-ray fingers with internal empty air channels are 3D printed as a single structure. The air channels act as a tactile sensor by measuring their pressure with an off-the-shelf pressure transducer. Each pair of channels (marked as the same color) under the inner and outer surfaces of the finger is sealed on the side surface and connected to one differential air pressure transducer.

3.3.1 Hardware Design and Sensorization

We design FORTE’s fingers based on fin-ray structures [42]. When deformed by a grasped object, the geometry passively conforms to the object, enabling a secure grasp. However, this large deformation makes it difficult to integrate tactile sensing. To address this, we sensorize the fin-rays using fluidic innervation, a barometric tactile sensing approach implemented via 3D-printed internal air channels [213, 248]. As the structure deforms, the air channels deform accordingly, producing measurable changes in air pressure captured by off-the-shelf pressure transducers. As fluidic innervation does not change a structure’s overall geometry, it preserves the fin-ray’s high compliance.

The fin-ray fingers are fabricated from EPU 40 using a Carbon M2 printer, following the material selection and fabrication procedure of Truby et al. [213]. A standard fin-ray geometry is modified to include internal air channels along the inner and outer surfaces, as shown in Figure 3.2. These channels are cleared with solvent, sealed at one end, and connected to a pressure transducer at the other. The sensing

mechanism is passive: the channels contain ambient air and are neither pressurized nor actuated. As the fin-ray deforms under contact, the internal air pressure changes, capturing the deformation of the structure. The channels are arranged in an L-shape to capture both vertical and horizontal deformation. Each channel has a diameter of 2 mm, with a 1 mm wall separating it from the surface. The distal ends are sealed using silicone epoxy (Sil-Poxy, Smooth-On). Each pressure transducer (All Sensors ELVR-L01D, 1inH₂O) is connected to a pair of inner and outer channels via silicone tubing. Because the transducers measure differential pressure, each sensor captures the net pressure change between the paired channels.

Each finger is sensorized with 3 transducers, resulting in 6 sensor signals total across FORTE. The transducers are connected to the analog inputs of an ESP32-S3 and sampled at 2 kHz with 11-bit resolution to capture rapid and transient dynamics that occur over short time scales. The analog readings are normalized to the range of -1 to 1 and transmitted at 2 kHz to a host PC. The sensor signals are filtered with a median filter of size 11 to reduce impulsive noise while preserving transient features, inducing at most 2.5 msec of latency.

3.3.2 Force Estimation Algorithm

We estimate the gripping force directly from the filtered internal pressure readings. We employ Support Vector Regression (SVR) with a radial basis function (RBF) kernel. To account for sensor drift and stress relaxation, three additional features are introduced for each channel, representing the mean values of the most recent 2.5, 5, and 10 seconds of historical data. Let $S(t) \in \mathbb{R}^6$ be the sensor signal frame at time t . The feature vector $z(t)$ is defined as:

$$z(t) = \bar{S}_{2.5}(t) \oplus \bar{S}_5(t) \oplus \bar{S}_{10}(t)$$

where $\bar{S}_\tau(t) \in \mathbb{R}^6$ denotes the mean of the past τ seconds of sensor data. The most recent sensor frame $S(t)$ is concatenated with $z(t)$ to form the input vector $v = S(t) \oplus z(t) \in \mathbb{R}^{24}$.

3.3.3 Slip Definition and Detection

Slip includes *incipient slip* (partial local motion) and *gross slip* (full contact motion) [33, 85], typically initiating locally when friction is exceeded and manifesting as translational, rotational, or stick-slip motion [25]. Detecting such events is challenging, and existing approaches fall into three categories. Motion-based methods (e.g., vision-based tactile sensing) track surface displacement with high resolution but often require large normal forces, limiting their use in delicate tasks [55, 64, 132]. Vibration-based methods detect micro-motions through frequency features, but are sensitive to noise and material variation, making them difficult to generalize [33, 58, 115, 135, 178]. Learning-based methods avoid manual feature design but require large datasets [69, 104, 151, 186, 219]. To overcome these limitations, we leverage the high temporal resolution of our sensors and detect slip via a second-order spectral feature that directly captures stick-slip dynamics. This approach is robust to noise, operates under minimal normal force, and enables simple, analytical slip detection.

First, we apply the Discrete Fourier Transform to the sensor signal $S_i(t)$ for each sensor index i . The signal is segmented into K overlapping windows of length N with a step size of $N - O \cdot N$, where O denotes the overlap factor. For each segment k , the signal is windowed and transformed into the frequency domain as

$$X_{i,k}(f) = \sum_{n=0}^{N-1} x_i(n)w(n)e^{-j2\pi fn}, \quad (3.1)$$

where $x_i(n) = S_i\left(t_k - \frac{N-1-n}{f_s}\right)$ represents the n -th sample in the k -th segment ending at time t_k , $w(n)$ is the Hann window, and $j = \sqrt{-1}$. Thus, the sensor signal for segment k of sensor i is implicitly represented in $X_{i,k}(f)$.

The quantity $|X_{i,k}(f)|^2$ estimates the signal power at frequency f for segment k . Normalizing this yields the power spectral density (PSD) $P_i(f)$ for sensor i , with units $(\text{inH}_2\text{O})^2/\text{Hz}$. For each sensor i , we compute the PSD over normalized frequen-

cies $f_k = \frac{k}{N}$, $0 \leq k < N$, using Welch's method:

$$P_i(f_k) = \frac{1}{f_s \sum_{n=0}^{N-1} w^2(n)} \left| X_{i,k} \left(\frac{k}{N} \right) \right|^2. \quad (3.2)$$

From the resulting PSD, we extract a scalar feature by selecting the maximum value across a specified frequency range:

$$P_i^{\max} = \max_{f \in [f_{\min}, f_{\max}]} \{10 \log_{10}(P_i(f) + \epsilon)\}, \quad (3.3)$$

where a small constant $\epsilon = 1 \times 10^{-12}$ is added to avoid numerical instability when taking the logarithm. This feature captures the peak power magnitude within the selected frequency band $[f_{\min}, f_{\max}]$.

To quantify temporal fluctuations in spectral power and search for our large-magnitude frequency feature, we compute the moving variance of the PSD magnitude as a second-order statistical feature. For each sensor i , we maintain a double-ended queue of $P_i^{\max}$ as a history H_i , which has a fixed length V . We then compute the moving variance over the feature history:

$$\sigma_i^2 = \begin{cases} \text{Var}(H_i), & \text{if } H_i(k+1) - H_i(k) > \delta, \forall k, \\ 0, & \text{otherwise.} \end{cases} \quad (3.4)$$

Each sensor i belongs to either the right finger or the left finger. To detect slip at the finger level and reduce noise, we average the moving variance across sensors on each finger. Let $G = \{R, L\}$ denote the set of fingers. The averaged variance is:

$$\bar{\sigma}_g = \begin{cases} \frac{1}{|g|} \sum_{i \in g} \sigma_i^2, & \text{if } \bar{\sigma}_g \geq \alpha, \forall g \in G, \\ 0, & \text{otherwise.} \end{cases} \quad (3.5)$$

Finally, we define the slip indicator η to be 1 if either finger exceeds a pre-defined threshold T :

$$\eta = \begin{cases} 1, & \text{if } \max_{g \in G} \bar{\sigma}_g > T, \\ 0, & \text{otherwise.} \end{cases}$$

Table 3.1: Slip detection parameters. We report the parameter settings used in our slip detection algorithm.

Parameter	Value
Median filter window size	$M = 11$
FFT window size	$N = 400$
Overlap	$O = 0.99$
Frequency band	$[f_{\min}, f_{\max}] = [10, 50] \text{ Hz}$
PSD history length (moving window size)	$V = 15$
Minimal increment for monotonicity	$\delta = 0.1 \text{ dB}$
Minimum group average variance	$\alpha = 0.6 \text{ dB}^2$
Slip detection threshold	$T = 2 \text{ dB}^2$

We summarize the parameter values we use for FORTE in Table 3.1. These parameters result in a slip detection frequency of 500 Hz and a moving variance computed from PSD features over the last 30 msec. These parameters lead to FFT bins that span the frequency range $[7.5, 52.5] \text{ Hz}$, approximating the range of mechanoreceptors in the human hand [99].

3.4 Characterization

In this section, we characterize FORTE’s ability to estimate grip force and detect slip during an attempted grasp. We attach the FORTE fin-ray fingers to a linkage-based gripper [193] and mount the entire assembly to a Franka Emika *Panda* robot [73]. We define $\theta_{\text{closed}} = 0^\circ$ as the servo position when the gripper is fully closed. We define θ_{zeroF} as the servo position when the gripper’s fingers just touch the object, applying zero force. Each object has its own specific θ_{zeroF} , depending on its size.

3.4.1 Grip Force Estimation

To evaluate force estimation across different object shapes, we perform repeated grasps on custom load-cell indenters (see Figure 3.3). The load cell is centered within the gripper, and a simple *close gripper* action is executed. For each of the six indenters, we collect 40 trials. In each trial, the gripper closes by a random amount, uniformly sampled from θ_{closed} to θ_{zeroF} , generating varying contact forces measured

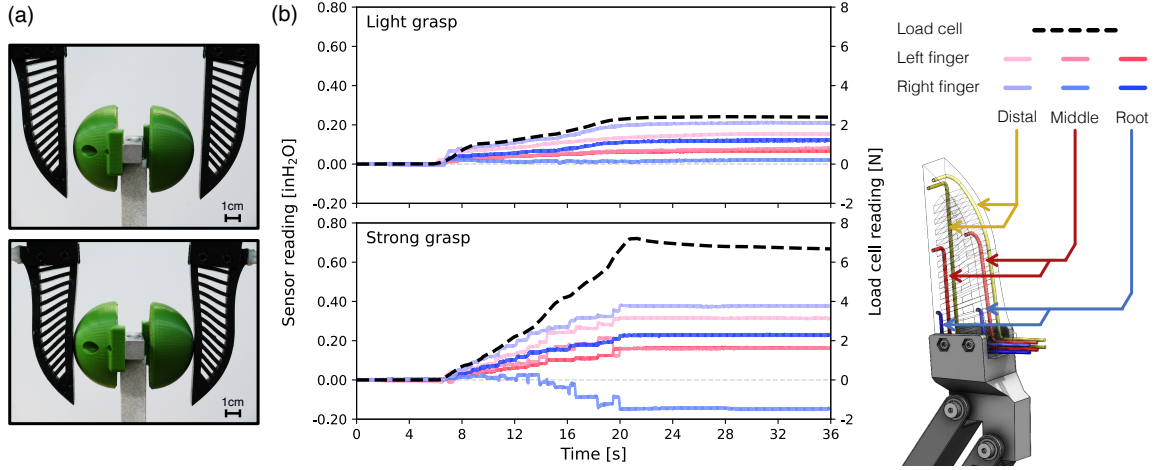


Figure 3.3: Grip force data collection. (a) Snapshots of the gripper in open and closed poses during data collection. (b) Sensor and load cell readings over time for light and strong grips. Red and blue lines correspond to the left and right fingers, with each line representing distal, middle, or root sensing locations. Sensor outputs increase monotonically with grip force, with the largest response at the distal sensors, indicating that pressure signals scale with finger deformation and support force estimation.

by the load cell and compared with FORTE readings. Sensor signals are normalized by subtracting the initialization mean and filtered with a median filter (kernel size 11). The resulting dataset spans 3 hours and contains 108k data points.

The filtered sensor signals correlate well with grip force, as shown by 10-fold trial-wise cross-validation (see Figure 3.4). Our method achieves an average root mean square error (RMSE) of 0.187 N over a 0–8 N force range across all indenter shapes. This outperforms prior work reporting a 0.39 N error over 0.9–4.5 N using a barometric strain gauge array [38], and is comparable to a 0.17 N error over 0.3–3.0 N using an external camera [236]. It also improves over the baseline RMSE of 0.211 N without the feature vector in Section 3.3.2, demonstrating the effectiveness of our approach.

3.4.2 Slip Detection

To characterize slip detection, we generate slip events by lifting objects with insufficient grasp force, as presented in Figure 3.5. The robot lifts at a constant

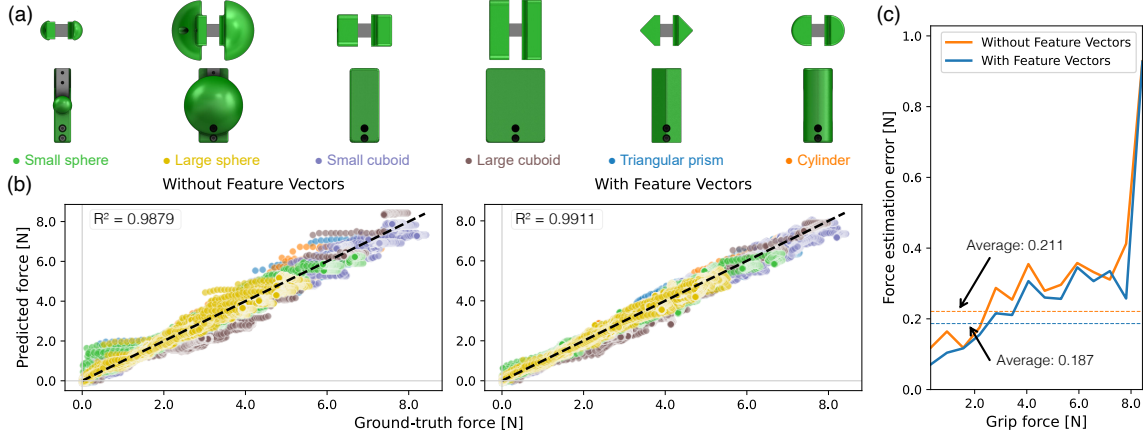


Figure 3.4: Grip force estimation results. (a) Renders of the custom load-cell rigs used for grip force data collection. (b) Predicted versus ground-truth force for models trained without (left) and with (right) the feature vectors from Section 3.3.2. Colored points correspond to different testing rigs, and the dashed black line denotes perfect prediction. (c) Force estimation error as a function of applied grip force. Solid lines show the root mean square error within each force bin, and horizontal dashed lines indicate the overall average error. Using feature vectors reduces the average error from 0.211 N to 0.187 N.

velocity of 1 mm/s at 20 Hz using Cartesian velocity control with Deoxys [258]. We evaluate our method on an apple and a jam jar. For each object, we define θ_{grasp} as the minimum joint position that yields a stable grasp, determined by gradually decreasing from θ_{zeroF} until the object can be lifted. In each trial, we sample a gripper position $\hat{\theta} \in [\theta_{\text{zeroF}}, \theta_{\text{grasp}})$, producing a force below the stable grasping force (< 0.5 N) that induces slip. The sampled ranges are $[0.552, 0.527]$ rad for the apple and $[0.476, 0.460]$ rad for the jam jar. The lifting motion is recorded, and the object’s rotation and translation are estimated from pixels. We collect 5 trials per object.

Analysis of the slip data shows that the two fingers often exhibit asymmetric responses due to uneven contact. We define the finger with higher contact force as the *leading finger* and the other as the *trailing finger*. This asymmetry is reflected in the finger-wise moving variance, as shown in Figure 3.5 (d). In our examples, the right finger leads in slip event 1, while the left finger leads in events 2 and 3. In some cases, the trailing force is dominated by table friction, as the applied grip force at θ_{grasp} (~ 0.5 N) is much smaller than the object weight (2.27 N for the apple and

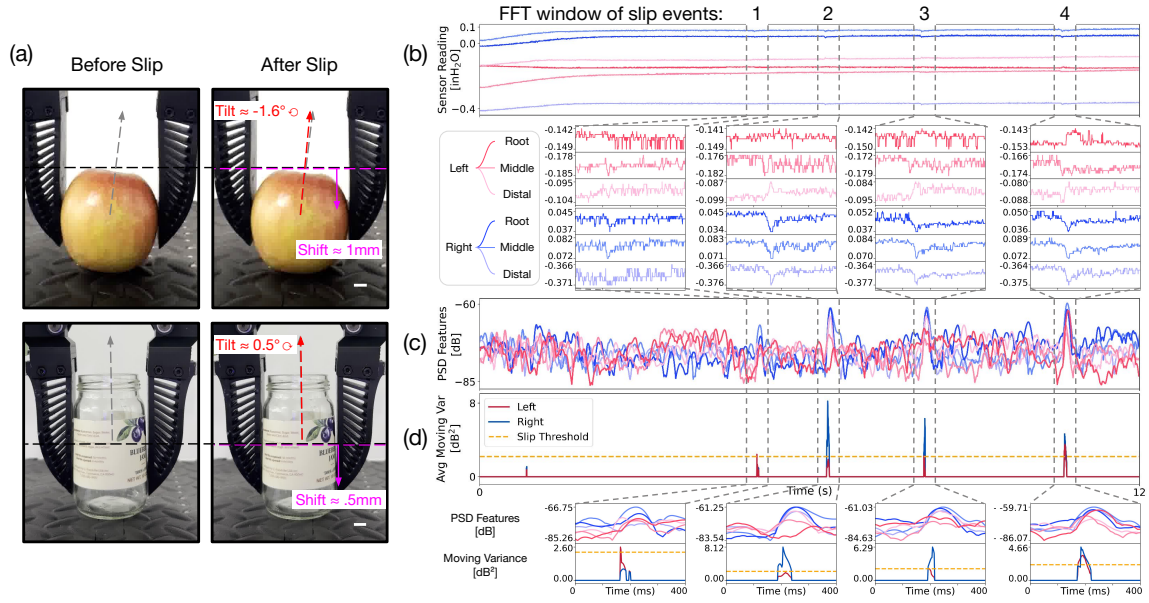


Figure 3.5: Slip detection characterization. (a) Snapshots of object displacements caused by stick-slip events. The time difference between the two snapshots of both objects is 66.7msec. The snapshots of apple correspond to Slip Event 1. Scale bars represent 1cm. (b–d) Filtered sensor readings, PSD features, and average moving variance of one representative trial of lifting an apple. There are four slip events during this trial. At the onset of each slip event, a sharp increase in the PSD feature is observed, resulting in a corresponding spike in the average moving variance, as further illustrated in the zoomed-in views.

1.42 N for the jar). We therefore focus on detecting slip at the leading finger, which captures the onset of gross slip.

We find that our slip detection algorithm can efficiently capture transitions from incipient slip to gross slip with a fixed slip threshold $T = 2 \text{ dB}^2$. From characterization, we identify two main types of slip events:

- **Leading-Finger Stick-Slip:** Stick-slip occurs at the leading finger while the trailing finger undergoes gross slip.
- **Simultaneous Stick-Slip:** Stick-slip occurs simultaneously at both fingers.

Figure 3.5 includes both types of slip events. Figure 3.5 (b) shows the filtered sensor signals used as input to the slip detection algorithm. The PSD features, computed

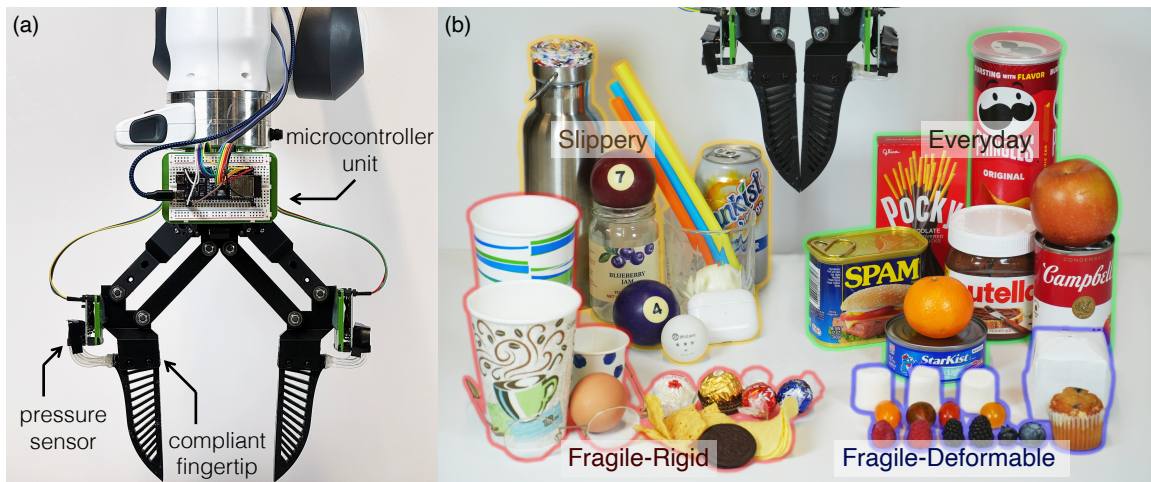


Figure 3.6: Experimental setup. (a) FORTE is mounted on a Franka Emika *Panda* robot arm for tabletop grasping tests. Scale bar represents 5cm. (b) Objects used for evaluation. Fragile objects are selected to evaluate force estimation accuracy and gripper compliance, while slippery items and eight everyday YCB objects are chosen to assess slip detection performance.

using Equation 3.3 from FFT windows of size 400, are shown in Figure 3.5 (c), while Figure 3.5 (d) presents the finger-wise average moving variance from Equation 3.5. Comparing Figures 3.5 (c) and (d) highlights the advantage of capturing changes in spectral power rather than its magnitude. The first slip event is barely distinguishable in the PSD features, whereas the moving variance clearly identifies it. Zoomed-in views show that the detection latency from the onset of sensor change is consistently below 100 msec. Overall, FORTE enables precise and real-time slip detection.

3.5 Experiments

In this section, we evaluate FORTE against baseline methods for delicate object grasping. We construct an evaluation set of 31 object types (47 instances), grouped into four categories: Fragile-Rigid, Fragile-Deformable, Slippery, and Everyday objects, as shown in Figure 3.6. The everyday objects are selected from the YCB dataset [27]. For each grasp strategy, we conduct 10 trials per object type, with results reported in Table 3.2.

3.5.1 Experimental Setup

All experiments use the same gripper setup described in Section 3.4 and follow a standardized procedure. The gripper is initialized, and an object is manually placed between the fingers. The gripper closes until contact is detected, defined as the maximum change in any tactile channel exceeding a threshold of $\tau = 0.5\%$ of full-scale span within a sliding window of $w = 200$ samples (0.1 sec). Upon contact, the gripper applies an initial force of $f_{\text{init}} = 0.25$ N. The object is then lifted at $v_{\text{lift}} = 5$ mm/sec using Cartesian velocity control at 20 Hz. If slip is detected, the gripper command is incremented by $\Delta_{\text{cmd}} = 0.88^\circ$ to increase the grip force. The lift continues until the object is either successfully lifted or dropped.

For FORTE, force estimation runs at 100 Hz with latency below 10 msec, enabling closed-loop control. Slip detection runs at 500 Hz with a typical latency below 100 msec, based on a signal window size of 400 and an overlap factor of 0.99. To evaluate generalization, the slip detection threshold T is set using values derived from the apple and jam jar in Section 3.4.2, making most test objects unseen.

We compare FORTE against two baseline conditions to isolate the contributions of compliance and tactile sensing:

- **On-Off:** The gripper closes fully at a constant speed before lifting, without using any sensing. This baseline evaluates the performance of the compliant fingers alone.
- **Without-Slip:** The gripper follows the same procedure but disables slip detection, using a fixed force without reactive adjustment. This baseline evaluates the role of slip sensing.

3.5.2 Grasping Performance

Overall, FORTE achieves a success rate of 91.9% across all 310 trials, significantly outperforming the On-Off (60%) and Without-Slip (52.6%) baselines, as shown

in Table 3.2. The gentle initial gripping force of 0.25 N enables reliable handling of fragile objects, with 13 out of 14 objects lifted successfully. In contrast, the On-Off controller damages 8 of the 14 fragile objects and causes irreversible deformation in items such as blueberries, grapes, tomatoes, and marshmallows. However, it achieves 100% success on a raw egg and a chocolate ball, indicating that compliant fingers alone can sometimes suffice.

While the Without-Slip baseline performs on par with FORTE on Fragile objects, it fails significantly on Slippery and Everyday objects, achieving only 31.1% and 1.3% success rates. In these categories, 15 out of 17 objects cannot be reliably lifted with the initial gripping force. The Without-Slip baseline also performs worse than the On-Off baseline, which achieves 88.8% and 100% success rates. In contrast, FORTE uses adaptive grasping with slip detection to achieve 100% success on 10 out of 15 objects, with overall category success rates of 88.8% and 85%. The On-Off baseline performs comparably or better in some cases by applying excessive force to prevent slip. Further tuning of FORTE’s force controller could improve performance in these cases, potentially at the cost of damaging fragile objects.

None of the three methods successfully lifted hard-boiled eggs. FORTE and the Without-Slip baseline failed due to insufficient grip force, while the On-Off baseline gradually squeezed the egg out of the gripper. Slip was detected in all Pringles trials, but the object was dropped once when grasped sideways near the distal finger region. The delayed response to the final slip suggests a need for improved control responsiveness for future applications.

In conclusion, FORTE demonstrates strong generalization across fragile, slippery, and everyday objects by combining compliant fingers with real-time force estimation and slip detection. These results highlight its effectiveness in delicate manipulation and its potential for more responsive robotic interaction in real-world settings.

3.5.3 Slip Detection Performance

In addition to grasping performance, we evaluate the accuracy of the slip detection algorithm in real-world scenarios. For each trial, we record whether slip occurs during lifting and classify the prediction as a true positive, true negative, or false positive. Summary statistics are reported in Table 3.2. When both ground truth and predicted slip are zero, precision is defined as 1.0 to reflect correct abstention from false positives.

The slip detection algorithm achieves an overall trial-wise accuracy of 0.93 and an F1 score of 0.91. A precision of 1.0 across all 310 trials indicates reliable slip detection without unnecessary false positives, avoiding excessive grasp adjustments and contributing to a 98.6% success rate on fragile objects.

The algorithm targets stick-slip events, as described in Section 3.4.2, by measuring micro-vibrations at the contact surface caused by transitions from static to kinetic friction. This enables slip detection even without noticeable geometric changes, such as in elongated objects (e.g., bottles, cans, and jars). However, the approach is less effective for extremely smooth surfaces (e.g., a Pocky box and a boiled egg), where friction transitions produce minimal force variation. In such cases, slip occurs smoothly at lift-off, and the resulting signal changes are insufficient to generate detectable increases in PSD features. Across 20 trials involving raw eggs and apples, the algorithm produces 3 false negatives. In these cases, the objects roll out from the side of the gripper due to rotational gross slip. This behavior is caused by the small contact area at the finger edge and the objects’ curved geometry combined with an off-centered center of mass.

In addition to the two targeted slip types defined in Section 3.4.2, we identify two additional slip modes that can be captured by our algorithm:

- **Rotational-to-translational slip transition:** Slip begins as rotational gross slip and transitions into translational gross slip during lifting.

- **Texture-affected slip:** Local surface texture variations cause detectable fingertip deformation during gross slip.

A rotational-to-translational slip transition occurs in off-centered grasps of elongated objects, such as a Pringles bottle or a metal bottle. As one end is lifted, the object tilts before transitioning into translational slip. Texture-affected slip occurs when the fingertip moves across surface features, such as the ridges on a tomato soup can. Further work is needed to better capture and distinguish these slip modes.

Overall, the slip detection algorithm demonstrates robust performance across diverse and unseen objects, with responsiveness to detect and react to slippage during lifting. It captures multiple slip types within a unified analytical framework. This capability is essential for achieving delicate manipulation in real-world settings.

3.6 Conclusion

We introduced FORTE, a system that integrates force and slip sensing into compliant robotic fingers through a simple 3D-printing-based process. This design enables parallel grippers to perform delicate manipulation tasks that require precise force control, such as handling fragile objects, including raspberries and potato chips. With reliable slip detection achieving an F1 score of 0.91, the system achieves a 91.9% single-trial grasp success rate across 31 diverse objects. The embedded sensing provides high-rate tactile feedback at 2 kHz, enabling accurate force estimation and timely slip detection. By integrating sensing directly within the finger structure, FORTE offers a compact and practical solution for force-aware manipulation. These results show that embedding tactile sensing within hardware provides a general way to enhance robotic capability across tools and end-effectors without complex external sensors. Extending this idea to the handheld gripper in the previous chapter would enable contact-aware feedback while preserving a unified interface across robots. This integration enhances perception at the point of contact and supports more effective skill learning and more reliable manipulation.

Table 3.2: Grasp success rates and slip detection performance. We report grasp success rates and slip detection metrics across diverse objects for FORTE and baseline methods. Accuracy measures overall correctness, precision reflects the reliability of slip predictions, recall captures slip detection coverage, and F1 balances precision and recall.

Category	Object (weight [g])	Number of trials with slip	True positives	True negatives	False positives	On-Off	Without-Slip	FORTE
Fragile-Rigid (7 objects)	Oreo Thins (7)	0	0	10	0	$\times$	10/10	10/10
	Raw Egg (62-64)	5	3	5	0	10/10	4/10	8/10
	Tortilla Chip (3-5)	0	0	10	0	$\times$	10/10	10/10
	Potato Chip (2)	0	0	10	0	$\times$	10/10	10/10
	Chocolate Ball (10-11)	0	0	10	0	10/10	10/10	10/10
	Eye-glasses (11)	0	0	10	0	N/A	10/10	10/10
	Paper Cup (2,11,20)	0	0	10	0	$\times$	10/10	10/10
	Category success rate					33.3% (20/60)	91.4% (64/70)	97.1% (68/70)
Slip detection metrics		Accuracy 0.97	Precision 1.00	Recall 0.60	F1 score 0.75			
Fragile-Deformable (7 objects)	Origami Ball	0	0	10	0	$\times$	10/10	10/10
	Raspberry (3-4)	0	0	10	0	$\times$	10/10	10/10
	Blackberry (4-5)	0	0	10	0	$\times$	10/10	10/10
	Blueberry (1-2)	0	0	10	0	Oversqueeze	10/10	10/10
	Grape Tomato (10-15)	0	0	10	0	Oversqueeze	10/10	10/10
	Marshmallow (4-6)	0	0	10	0	Oversqueeze	10/10	10/10
	Muffin (24-26)	0	0	10	0	$\times$	10/10	10/10
	Category success rate					0% (0/70)	100% (70/70)	100% (70/70)
Slip detection metrics		Accuracy 1.00	Precision 1.00	Recall 0.00	F1 score 0.00			
Slippery (9 objects)	Jam Jar (145)	8	8	2	0	10/10	0/10	10/10
	Whiskey Cup (278)	10	10	0	0	10/10	0/10	10/10
	Ping-pong Ball (3)	0	0	10	0	10/10	10/10	10/10
	Billiard Ball (170)	10	10	0	0	10/10	0/10	10/10
	Straw (1)	0	0	10	0	10/10	10/10	10/10
	Air-pod Case (61)	2	2	8	0	10/10	8/10	10/10
	Soda (371)	10	10	0	0	10/10	0/10	10/10
	Metal Bottle (299)	10	10	0	0	10/10	0/10	10/10
	Hard-boiled Egg (52)	10	0	0	0	0/10	0/10	0/10
	Category success rate					88.8% (80/90)	31.1% (28/90)	88.8% (80/90)
Slip detection metrics		Accuracy 0.91	Precision 1.00	Recall 0.82	F1 score 0.90			
Everyday (8 objects)	Pringles (197)	10	10	0	0	10/10	0/10	9/10
	Nutella (311)	10	10	0	0	10/10	1/10	10/10
	Tuna Can (178)	10	10	0	0	10/10	0/10	10/10
	Campbell Soup (348)	10	10	0	0	10/10	0/10	10/10
	Mandarin (88-96)	10	10	0	0	10/10	0/10	10/10
	Apple (231)	10	9	0	0	10/10	0/10	9/10
	SPAM (368)	10	10	0	0	10/10	0/10	10/10
	Pocky Box (88)	10	0	0	0	10/10	0/10	0/10
	Category success rate					100% (80/80)	1.25% (1/80)	85.0% (68/80)
Slip detection metrics		Accuracy 0.88	Precision 1.00	Recall 0.86	F1 score 0.93			
Overall success rate						60% (180/300)	52.6% (163/310)	91.9% (285/310)
Overall slip detection metrics		0.93	1.00	0.84	0.91			

Chapter 4: Visuomotor Skill Learning with Model-Based Control

4.1 Introduction

Advanced robotic systems have long attempted to leverage their rich physical capabilities to perform complex tasks. Humanoid robots are a representative example. Owing to their human-like embodiment, they offer strong potential for operating in human-centric environments and performing a wide range of everyday tasks [47]. Recent advances in hardware design and control algorithms have further accelerated the development of humanoid platforms [181, 207]. Despite this progress, enabling robots to operate autonomously in real-world environments remains challenging. Robots must handle uncertainty in perception, complex contact interactions, and diverse task conditions. As a result, many current systems still rely on task-specific manual programming [103, 153, 194] or human teleoperation [92, 102, 170]. These approaches are difficult to scale across tasks and environments, making the lack of autonomy a major obstacle to the widespread deployment of humanoids in real-world settings.

Imitation learning has emerged as a flexible, data-driven approach for developing robot skills from human demonstrations of robot operation [233, 251]. In particular, deep imitation learning methods based on large neural networks have achieved promising results on simpler robot morphologies, such as tabletop manipulators and wheeled platforms [23, 148]. However, applying these methods to humanoid robots introduces two additional challenges. The first challenge stems from the fact that humanoid robots are floating-base systems that need to maintain balance while physically interacting with the environment. In contact-rich tasks, the robot’s physical interactions affect its dynamics and increase the uncertainty and complexity of controlling the robot. This problem is further exacerbated by the absence of tactile and proprioceptive sensing modalities in standard teleoperation interfaces. The second

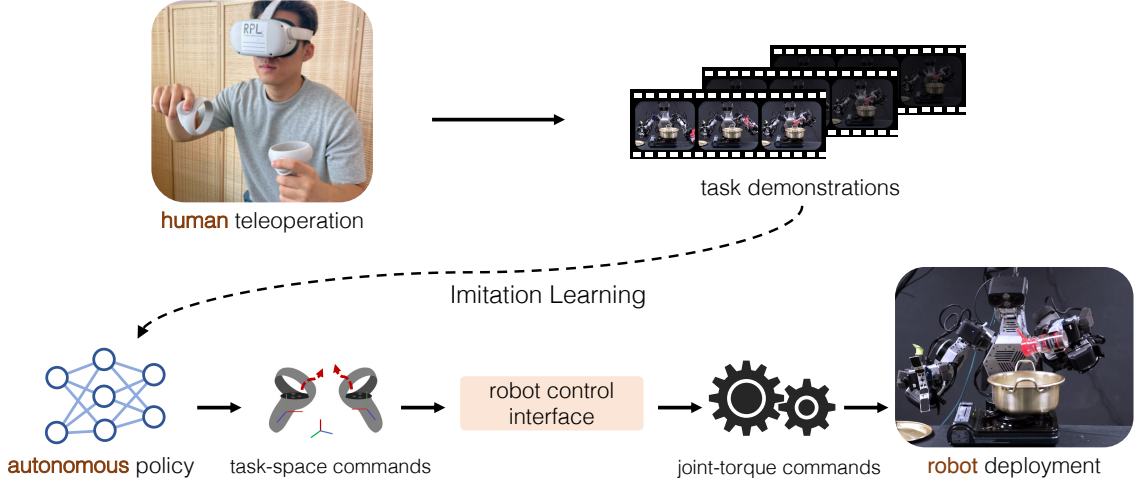


Figure 4.1: Overview of TRILL. TRILL addresses the challenge of learning humanoid loco-manipulation. We introduce a learning framework that facilitates teleoperated demonstrations with task-space commands provided by a human demonstrator. The trained policies leverage human complexity and adaptability in decision-making to generate these commands. The robot control interface then executes these target commands through joint-torque actuation while complying with robot dynamics. This synergistic combination of imitation learning and whole-body control enables successful method implementation in both simulated and real-world environments.

challenge lies in the humanoid robots’ high degree of freedom, leading to a large action space that raises the data requirements and computational demands for policy learning.

Our key idea to overcome these challenges is incorporating the whole-body control formulation [189] into our data collection system and policy learning method. Whole-body control is a comprehensive control framework that employs a minimal set of simple, low-dimensional rules to fully leverage the capabilities of floating-base robots for compliant multi-contact interaction with their environment. Utilizing this controller simplifies the process for human operators to provide task demonstrations through an intuitive virtual reality (VR) interface. Moreover, it enables our policy to predict high-level actions in the task space. These actions can subsequently be transformed into joint-level torque commands for actuation.

To this end, we introduce TRILL (Teleoperation and Imitation Learning for Loco-manipulation), a deep imitation learning framework for learning sensorimotor

policies of humanoid robots from human demonstrations. TRILL consists of three main components. The first is a VR-based teleoperation interface, offering an intuitive way for humans to provide task demonstrations. The second is a whole-body controller, which reliably converts task-space trajectories of human demonstrations into joint-torque actions. The use of whole-body control prioritizes robot stability and tracks limb trajectories to generate dynamically feasible motions. The final component is a data-efficient imitation learning algorithm for training loco-manipulation policies. Our policy predicts target setpoints for the robot’s hands and sends commands prescribing gait sequence commands, enabling sample-efficient training with high-level action abstractions. These components together enable TRILL to perform complex loco-manipulation tasks while adeptly stabilizing uncertain robot dynamics.

We evaluate our approach in simulation and real hardware settings. In two simulated environments, TRILL achieves success rates of 96% for free-space locomotion tasks, 80% for manipulation tasks, and 92% for loco-manipulation tasks. Across all tasks, our method outperforms state-of-the-art imitation learning baselines [148] by 28% in success rate. We also deploy our method to a real-world humanoid robot, DRACO 3 [16], achieving an average success rate of 85% in two contact-rich manipulation tasks. To the best of our knowledge, this work presents the first successful implementation of deep imitation learning for learning visuomotor policies of complex manipulation tasks on real-world humanoid systems.

4.2 Related Work

Loco-manipulation of Humanoid Robots Humanoid robots pose unique challenges due to discontinuous motion and the need to maintain balance, unlike wheeled mobile manipulators that operate on continuous manifolds and enable more straightforward motion planning [154, 225, 231]. Prior work has explored kinodynamic whole-body control to address these challenges [10, 28, 44, 103, 218], but these approaches are often task-specific, computationally intensive, and lack generality. Teleoperation

simplifies control through direct human input [199, 205, 209], but stable, real-time whole-body motion remains difficult to achieve, particularly for highly dynamic tasks. Recent inverse dynamics methods improve adaptability but suffer from high computational costs and numerical ill-conditioning [102, 174], limiting their practical use and constraining learning-from-demonstration pipelines. Our work leverages task-space action abstraction within a whole-body controller, instead of low-level joint retargeting, reducing computational complexity and improving robustness.

Learning from Teleoperation Demonstrations Learning from demonstration presents an effective approach to building robot behaviors with human supervision for complex and dexterous manipulation tasks [233, 254]. Teleoperated human demonstrations are especially valuable for reducing domain gaps between training and deployment. In addition, this paradigm enables relatively scalable data collection [146, 148, 215, 230]. As a result, extensive prior work [148, 156, 258] has explored its use for programming complex manipulation behaviors that are difficult to engineer manually. However, most existing approaches focus on tabletop manipulation or wheeled platforms. In contrast, collecting demonstrations for humanoid robots remains substantially more challenging, as action commands from human operators cannot be directly mapped to the robot. This difficulty arises from the complex floating-base dynamics of humanoids, along with stability constraints and uncertainties in state estimation. In this work, we develop a practical system that enables scalable collection of human demonstrations for humanoid loco-manipulation.

4.3 Method

We introduce TRILL, a deep imitation learning framework for humanoid loco-manipulation. The key to our approach is to decompose the loco-manipulation pipeline into a two-level hierarchy consisting of a high-level visuomotor policy and a low-level whole-body controller. The high-level action abstraction of the policy fa-

cilitates data-efficient learning. The low-level controller complements the high-level policy and stabilizes the robot dynamics while realizing the policy’s intended actions.

4.3.1 Problem Formulation

We model the problem of loco-manipulation as a discrete-time Markov Decision Process $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, R, \gamma, \rho_0)$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P}(\cdot|s, a)$ is the transition probability, $R(s, a, s')$ is the reward function, $\gamma \in [0, 1)$ is the discount factor, and $\rho_0(\cdot)$ is the initial state distribution. Our goal is to learn a closed-loop visuomotor policy $\pi(a_t|s_t)$ that maximizes the expected return $\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1})]$. In our context, $\mathcal{S}$ is the space of the robot’s sensory observations captured by its egocentric cameras and proprioceptive sensors, $\mathcal{A}$ is the space of the robot’s joint-torque commands, $R(s, a, s')$ is the reward function designed for the loco-manipulation task, and π is a closed-loop policy that we deploy on the robot.

To handle the complexity of visuomotor skills and train the policy π effectively, we decompose the policy π into a two-level hierarchy. At the high level is a neural network policy π_H that computes the target motion command u as the hands’ pose setpoints and the body’s locomotion. At the low level is a whole-body controller π_L that computes the robot’s joint-torque actions to realize target commands u from π_H . With this hierarchical abstraction, we can rewrite the whole policy as $\pi(a_t|s_t) = \pi_L(a_t|s_t, u_t)\pi_H(u_t|s_t)$.

TRILL utilizes this hierarchical structure, as illustrated in Figure 4.2. The high-level policy π_H generates task-space commands. We train π_H through imitation learning from human demonstrations collected through our VR teleoperation system. The low-level control policy π_L calculates motor torques to fulfill the commands set by π_H . We use the whole-body control formulation to implement the controller π_L in order to ensure the robust execution of the robot’s motions.

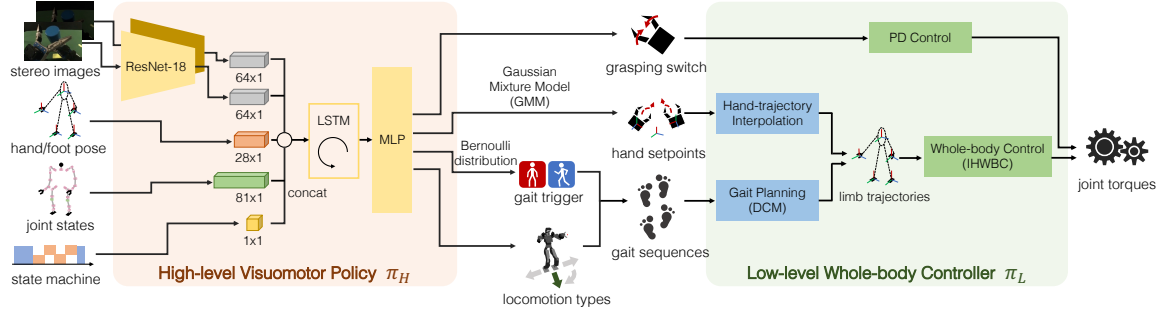


Figure 4.2: Model architecture of TRILL. The trained policies generate the target task-space command u_t at 20 Hz from the onboard stereo camera observations and the robot’s proprioceptive feedback. The robot control interface realizes the task-space commands, computes the desired joint torques a_t at 100 Hz, and sends them to the robot for actuation.

4.3.2 Teleoperation System

We define the Cartesian hand poses in the robot’s body frame as setpoints of robot arm motions. The poses of the VR hand controllers, with respect to the VR headset, are mapped to these setpoints. In this way, the task-space commands of hand poses reduce the action space and compensate for the lack of joint sensory feedback during teleoperation. We apply trajectory interpolation to prevent aggressive or discontinuous motion trajectories to ensure safe teleoperation. Further examination of the impact of this handling of hand poses trajectories is detailed in Section 4.4.2.

We model the locomotion commands as easily executable, pre-defined gait sequences to simplify our teleoperation operation. The buttons on the VR hand controllers trigger these sequences. Concretely, we employ the Divergent Component of Motion (DCM) approach for gait planning [210]. Based on the unstable portion of linear inverted pendulum dynamics, the DCM planner calculates the divergent component of motion. Unlike other methods that involve direct control of the robot’s Center of Mass (COM), the DCM method allows for indirect COM control. It provides simplicity of the planner and a larger basin of attraction. These features are significantly beneficial in loco-manipulation, as they allow more effective control of hand positions relative to the robot’s body motion.

4.3.3 Whole-body Control

Although the morphological similarities between humans and humanoid robots make teleoperation more intuitive, the differences in kinodynamic properties can significantly complicate robots’ low-level joint control. The lack of proprioceptive feedback makes it further difficult for human demonstrators to stabilize the robot’s pose. Particularly, physical interactions (e.g., manipulating heavy objects) alter the robot’s dynamics. With off-the-shelf VR controllers, human demonstrators must be able to operate the robot safely without force feedback from objects of unknown properties.

Given these challenges, a robust low-level whole-body motion controller is critical for effectively controlling humanoid robots. A desirable motion controller should fulfill two key requirements: 1) it should provide safe stabilization of the robot’s body under external disturbances or underlying model inaccuracy, abstracting away the floating-base dynamics from human operators’ input, and 2) it should have the capability to accurately track constantly varying trajectory commands, as set by π_H , to generate appropriate locomotion and manipulation behaviors for contact-rich tasks.

Inspired by recent advances in whole-body control [10, 118, 130, 226], we have employed a model-based optimal control method to calculate motor torques. Our controller stabilizes the robot’s floating base against the uncertainties and disturbances caused by object interactions and maps closed-loop commands accurately into the robot’s trajectories. Specifically, we have implemented the Implicit-hierarchical Whole-body Control (IHWBC) [10]. The formulation of the whole-body control is given as,

$$\begin{aligned} \min_{\ddot{q}, f_r} \sum_{i=1}^n w_i \|J_i \ddot{q} + \dot{J}_i \dot{q}_m - \ddot{x}_i^d\|^2 \\ + w_{f_r} \|f_r^d - f_r\|^2 + \lambda_q \|\ddot{q}\|^2 + \lambda_{f_r} \|f_r\|^2, \end{aligned} \quad (4.1)$$

subject to

$$S_f(A\ddot{q} + b + g - J^\top f_r) = 0$$

$$U f_r \geq 0$$

$$S_r f_r \leq f_r^{\max}$$

$$\ddot{q}_{\min} \leq \ddot{q} \leq \ddot{q}_{\max}$$

$$\tau_{\min} \leq S_a(A\ddot{q} + b + g - J_c^\top f_r) \leq \tau_{\max}.$$

The notation follows Section 3 in the original work of IHWBC [10]. In this optimization formulation, i denotes the terms used for controlling limb poses within IHWBC, such as the center of the robot’s body, hands, and feet. By prioritizing body pose stabilization, the robot can tolerate minor tracking errors of its end-effectors while ensuring the body’s stability. This prioritization prevents the robot from failing due to aggressive motion trajectories. In our model, target reaction forces f_r at the non-contact points on the hands and feet are processed as virtual impedances relative to target trajectories. This approach allows the robot to exhibit adaptive, compliant manipulation during object interactions. Following Equation 4.1, the target command torque is determined as, $\tau^{\text{cmd}} = S_a(A\ddot{q}^* + b + g - J_c^\top f_r^*)$. Subsequently, the final joint torque inputs $\tau^{\text{joint}} = \tau^{\text{cmd}} + k_p(q^{\text{cmd}} - q_m) + k_d(\dot{q}^{\text{cmd}} - \dot{q}_m)$ are sent out to the robot.

4.3.4 Training of Visuomotor Policies

Our collected demonstration dataset, denoted as $\mathcal{D}$, is composed of state-action pairs $\mathcal{D} = (s_i, u_i)_{i=1}^N$. Here, N stands for the total number of data points. The observation s_i includes stereo images from the robot’s onboard camera, actual positions of the robot’s feet (14D) and hands (14D), the status of the controller’s state machine (1D), and sine and cosine representations of joint positions (54D) and joint angular velocities (27D) for the arms, legs, and neck. The demonstration commands u_i include the subsequent setpoints for the hands (14D), grasping actions for both hands (2D), and locomotion instructions (1D).

We use this dataset to train our policy π_H with a deep imitation learning algorithm. In particular, we train a behavioral cloning policy with recurrent neural networks (RNNs) [148] to capture the temporal dependencies of the robot’s movements (see Figure 4.2). To handle the inherently noisy and multimodal nature of human demonstrations, π_H utilizes a Gaussian Mixture Model (GMM) [224] for policy output. We sample hand pose setpoints from the GMM and generate trajectories to these setpoints via interpolation.

The locomotion of the humanoid robot is discretized by abstracting the robot’s walking modalities as pre-defined gait sequences. It simplifies the teleoperation system and allows the demonstrator to operate the robot’s locomotion with button clicks. To improve the data efficiency of learning locomotion actions, we introduce a hierarchical decomposition of the action spaces into two components: *gait trigger* and *locomotion types*. The *gait trigger*, sampled from a Bernoulli distribution whose probability distribution is dictated by the outputs of the high-level policy, determines if the gait is activated. Once the gait is activated, the values of the *locomotion types* determine the final gait sequence types and produce the corresponding gait commands. Such factorization of the action space enables more sample-efficient training for locomotion, especially given the relatively sparse locomotion commands in the demonstration dataset.

4.4 Experiments

In this section, we demonstrate the feasibility and effectiveness of TRILL for data collection and policy deployment on a humanoid robot, both in simulated and real-world settings. Furthermore, taking advantage of the scalability and ease of simulation environments, we use them to investigate the following research questions: 1) What factors impact humans’ teleoperation performance and the ease of use of our VR-based interface? 2) How do different choices of observation and action spaces affect policy performances? 3) How does the policy performance scale with respect

to the demonstration dataset sizes?

4.4.1 Experimental Setup

We evaluate our method in a variety of locomotion and manipulation tasks. A task is considered successful if the robot accomplishes the designated goals within a specific time limit without falling or experiencing undesirable collisions with objects. If the robot accomplishes the objectives but has undesired collisions with objects or only partially completes the tasks, we consider the task partially successful. To thoroughly evaluate our model’s performance, we construct diverse task environments for simulation and real-world deployment with the DRACO 3 humanoid robot [16].

Simulation Setup In simulation, we design two realistic benchmark environments to study the robot’s loco-manipulation abilities: *Door* and *Workbench* in MuJoCo [212], as shown in Figure 4.3. In each environment, we evaluate the robot’s ability to successfully perform subtasks involving free-space locomotion, manipulation, and loco-manipulation.

In the *Door* environment, the robot performs the following three subtasks.

- ***Walking to the door (locomotion)***: The robot walks toward a door and reaches its handle through free-space locomotion.
- ***Unlocking the latch (manipulation)***: The robot stands still and rotates the knob to unlock the door.
- ***Pushing the door (loco-manipulation)***: While walking, the robot pushes the door, thereby combining locomotion and manipulation. This task involves coordinated arm-base interaction, where the robot has to consider the mechanical constraints imposed by the door’s mechanism.

Similarly, in the *Workbench* environment, the robot performs the following three subtasks:

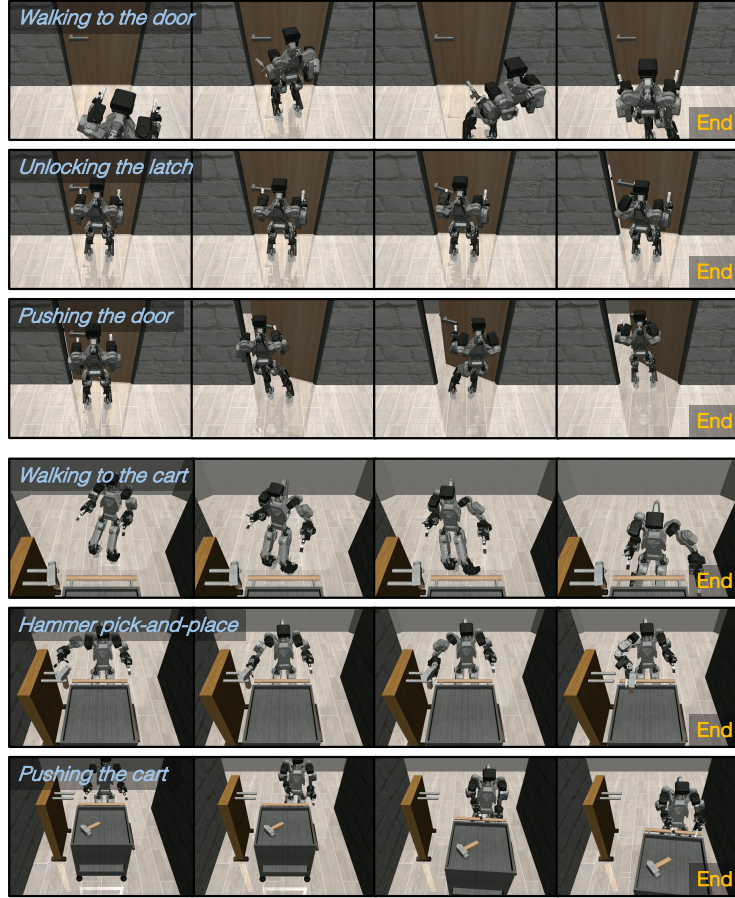


Figure 4.3: Timelapse of deploying TRILL in simulation. We present the deployment of policies trained through our method across three distinct tasks: free-space locomotion, manipulation, and loco-manipulation, in two unique simulation environment domains.

- ***Walking to the cart* (locomotion):** The robot approaches the cart. It requires precise positioning of the robot’s body and its hands to avoid collision with objects.
- ***Hammer pick-and-place* (manipulation):** The robot lifts the hammer from the hanger and accurately places it onto the cart. In contrast to the *Unlocking the latch* task in the *Door* environment, this task demands precise dexterous skills and thus poses a greater challenge for manipulation.
- ***Pushing the cart* (loco-manipulation):** While walking towards the goal,

the robot pushes a heavy cart. This task requires arm-base coordination, where the robot has to handle the planar motion constraints of the cart.

Real-World Setup Learning manipulation tasks on real-world robots presents additional challenges. This is primarily due to 1) the inherent uncertainty that arises from control and communication latency with real hardware systems, 2) the complex dynamics of robot actuators and the interactions of robots with objects in the real world, and 3) significant errors in estimating the robot’s base state, hindering the precision of manipulation behaviors. To demonstrate the efficacy of TRILL given the floating-base dynamics of real humanoid robots, we have designed two intricate manipulation tasks.

- ***Tool pick-and-place:*** The robot is tasked with accurately locating and grasping a tool before placing it into a box. This task is designed to evaluate the robot’s precision in handling manipulation tasks.
- ***Removing a spray cap:*** The robot is required to grasp a spray can with one hand and then remove its cap with the other hand. This task is designed to assess the robot’s proficiency in bimanual manipulation.

Data Collection We adopt a VR device (Meta’s Oculus Quest 2 [6]) for robot teleoperation. A stereo camera is mounted on the robot’s head, with the stereo images used as visual observations for both the human operator and the visuomotor policy. The operator’s hand poses are measured by the IMU on the Oculus hand controllers. Our experiments in simulation and with the real-robot system train policies on datasets of 250 demonstration trajectories for each task.

4.4.2 Quantitative Evaluation in Simulation

Deployment Results Table 4.1 reports quantitative evaluations of our simulated tasks. We compare our model’s performance against the following baselines.

Table 4.1: Quantitative results in simulation. We report full and partial success rates (%) on 25 trials of our TRILL policies against baselines. The numbers in parentheses are partial success rates.

	BC [59]	BC-RNN [148]	TRILL
<i>Walking to the door</i>	0 (0)	40 (80)	100 (100)
<i>Unlocking the latch</i>	48 (52)	80 (96)	92 (100)
<i>Pushing the door</i>	0 (0)	56 (64)	96 (96)
<i>Walking to the cart</i>	0 (0)	52 (88)	92 (96)
<i>Hammer pick-and-place</i>	0 (16)	60 (80)	68 (76)
<i>Pushing the cart</i>	0 (0)	80 (88)	88 (88)

- **BC:** a behavioral cloning baseline that learns a reactive policy conditioned on current observations. The model architecture is based on the design of Finn et al. [59], which uses convolutional image encoders.
- **BC-RNN:** the state-of-the-art method that employs a temporal sequence of past observations using recurrent neural networks introduced in Mandlekar et al. [148] for learning manipulation skills from teleoperated demonstrations. In contrast to the hierarchical action space used in TRILL, this baseline uses flat GMM outputs for both manipulation and locomotion commands.

As shown in Table 4.1, TRILL outperforms the two baseline methods across all tasks in full success rates and five out of the six in partial success rates. The performance gaps are more significant in free-space locomotion and loco-manipulation tasks. The BC-RNN baseline shows consistently better performances than BC, highlighting the importance of modeling the temporal dependencies for humanoid loco-manipulation skills. While these two baselines can learn policies that effectively produce hand trajectories for manipulation in some cases, they struggle to learn from the sparse locomotion commands in human demonstrations. Our model, which suitably formulates locomotion actions in separation with manipulation actions, can train visuomotor policies more efficiently than these baselines.

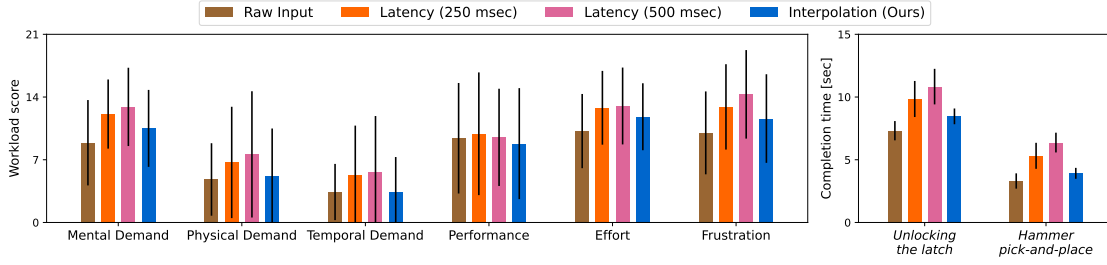


Figure 4.4: NASA TLX evaluation and completion times using the teleoperation system: We evaluate our teleoperation system using participants’ self-reports from the TLX (left) and completion times on 25 trials (right) for two manipulation tasks in simulation. Lower scores indicate better performance in both categories. We further examine the effect of varying teleoperation settings, specifically control latency and trajectory handling, on demonstrator performance. In our interface, the added workload from trajectory interpolation has a minimal effect on teleoperation performance, allowing our method to produce high-quality demonstration data without compromising robot safety.

Teleoperation Interface Evaluation We evaluate the intuitiveness of our VR-based teleoperation system. We also explore how the following factors influence user experience and demonstrator performance: 1) control delay and 2) trajectory interpolation used in our teleoperation system. To assess these factors, we recorded 25 episodes of human demonstration for each of the two simulated manipulation tasks. We utilize the NASA Task Load Index (TLX) [74] to quantitatively analyze workload and compare task completion time as the performance metrics.

- **Raw Input:** This baseline directly applies teleoperation commands without any trajectory interpolation. It is impractical in real-robot settings, as non-smooth trajectories caused by communication delays or unsafe teleoperation actions may cause damage. Nonetheless, this baseline allows us to methodically examine how our final setup’s trajectory interpolation, as described in Section 4.3.4, affects performance.
- **Latency (250 msec):** This baseline is a variant of our final setup where a 250 msec latency is artificially injected in order to study the impact of control delays on demonstrators’ performance.

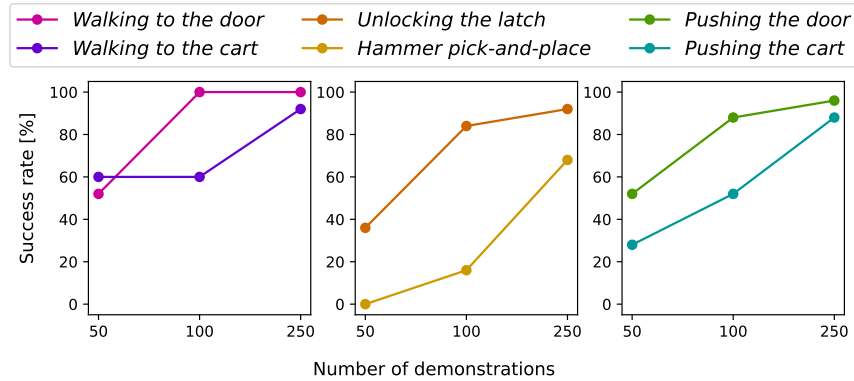


Figure 4.5: Simulation evaluation across different dataset sizes. We show the percentage changes in success rates on 25 trials, which were obtained from policies trained on datasets of different sizes. We observed that trained policies perform better across different tasks with an increase in demonstrations.

- **Latency (500 msec):** Similarly to the Latency (250 msec) baseline, it features a significantly larger control delay that could potentially hinder the demonstrators’ ability to perform tasks.

As indicated in Figure 4.4, we observe that as control latency rises, there is a corresponding increase in workload and a decrease in performance. It is also worth noting that this increase in workload burden impacts the demonstrator, consequently affecting the quality of the demonstration.

In our teleoperation setup, we implemented trajectory interpolation rather than directly using raw hand pose setpoints for limb trajectories at the whole-body controller. This was done to ensure safer robot operation, as well as smoother and less aggressive changes in command setpoints. When employing trajectory interpolation, there is only a minor impact on demonstration performance when compared to the Raw Input baseline. It suggests that the additional workload imposed by trajectory interpolation does not significantly affect teleoperation performance. As a result, our teleoperation approach can produce high-quality demonstration data while maintaining safe robot operation.

Varying Dataset Sizes We also conducted a systematic investigation into the effect of the dataset size on trained policies by varying the dataset size between 50 and 250 demonstrations. The results of this investigation are presented in Figure 4.5. Our findings indicate that an increase in the number of demonstrations generally leads to improved performance of trained policies across various tasks regardless of their difficulty.

Ablation Studies We conducted a study on the effect of observation and action space designs. In order to perform ablation studies, we compared the performance of model variants that employ different observation and action space designs, as described below.

- ***Monoscopic:*** In this model variant, only the right side of camera observation is used as input to the policy. This is intended to examine whether depth information derived from stereo images is essential for the robot’s performance.
- ***No-Joint-State:*** This variant excludes joint-state feedback in its observations. It is intended to examine the role of joint sensor feedback in learning locomotion and manipulation skills.
- ***Deterministic:*** Instead of sampling output hand setpoints from a GMM, this variant adopts a deterministic action output. It examines the importance of modeling multi-modality in action distributions and how it helps stabilize humanoid robots over the errors induced by their floating dynamics and uncertainty.
- ***Categorical:*** This variant employs a categorical action head to predict output locomotion commands. It is intended to assess the efficiency of our choice of action space tailored for humanoid locomotion.

Table 4.2: Ablation studies of TRILL in simulation. We use full and partial success rates (%) on 25 trials of trained policies as our evaluation metric. The numbers in parentheses are partial success rates. In addition, we present a comparison of the success rate of our final model, with changes indicated by color. The red color signifies a decrease, while the blue color denotes an increase.

	<i>Monoscopic</i>	<i>No-Joint-State</i>	<i>Deterministic</i>	<i>Categorical</i>
<i>Walking to the door</i>	100 (100)	92 ▼ ₈ (96 ▼ ₄)	92 ▼ ₈ (100)	52 ▼ ₄₈ (84 ▼ ₁₆)
<i>Unlocking the latch</i>	56 ▼ ₃₆ (84 ▼ ₁₆)	64 ▼ ₂₈ (84 ▼ ₁₆)	40 ▼ ₅₂ (72 ▼ ₂₈)	92 (96 ▼ ₄)
<i>Pushing the door</i>	16 ▼ ₈₀ (32 ▼ ₆₄)	24 ▼ ₇₂ (64 ▼ ₃₂)	24 ▼ ₇₂ (28 ▼ ₆₈)	80 ▼ ₁₆ (80 ▼ ₁₆)
<i>Walking to the cart</i>	100 ▲ ₈ (100 ▲ ₄)	88 ▼ ₄ (100 ▲ ₄)	36 ▼ ₅₆ (80 ▼ ₁₆)	92 (96)
<i>Hammer pick-and-place</i>	28 ▼ ₄₀ (48 ▼ ₂₈)	12 ▼ ₅₆ (20 ▼ ₅₆)	0 ▼ ₆₈ (0 ▼ ₇₆)	36 ▼ ₃₂ (64 ▼ ₁₂)
<i>Pushing the cart</i>	24 ▼ ₆₄ (44 ▼ ₄₄)	52 ▼ ₃₆ (60 ▼ ₂₈)	4 ▼ ₈₄ (4 ▼ ₈₄)	32 ▼ ₅₆ (40 ▼ ₄₈)

We report the average success rate and partial success rate of each model variant in Table 4.2. Our findings highlight the vital role of stereoscopic visual observations in tasks requiring precise hand movements. We hypothesize that this is because stereoscopic visual observations provide key spatial information to resolve depth ambiguity. Although joint state sensor modalities do not show as large an impact as stereoscopic vision, they remain important for successful manipulation. The *Deterministic* variant yields the most significant decline in success rates across manipulation and loco-manipulation tasks. It indicates that deterministic actions cannot effectively capture the action distribution’s multi-modality given the complex robot dynamics. Lastly, the *Categorical* variant showed the most considerable performance reduction in free-space locomotion and loco-manipulation tasks. It implies that separating locomotion actions into *gait trigger* and *locomotion types* improves learning efficiency for locomotion actions.

4.4.3 Real-Robot Deployment

Teleoperation Interface Evaluation We present the tracking errors of our teleoperation interface. These errors primarily result from phase lags, tracking issues, and workspace limitations imposed for safety reasons. Hand position errors account for 0.082 ± 0.032 m and hand orientation errors account for $22.2 \pm 10.9^\circ$ across human

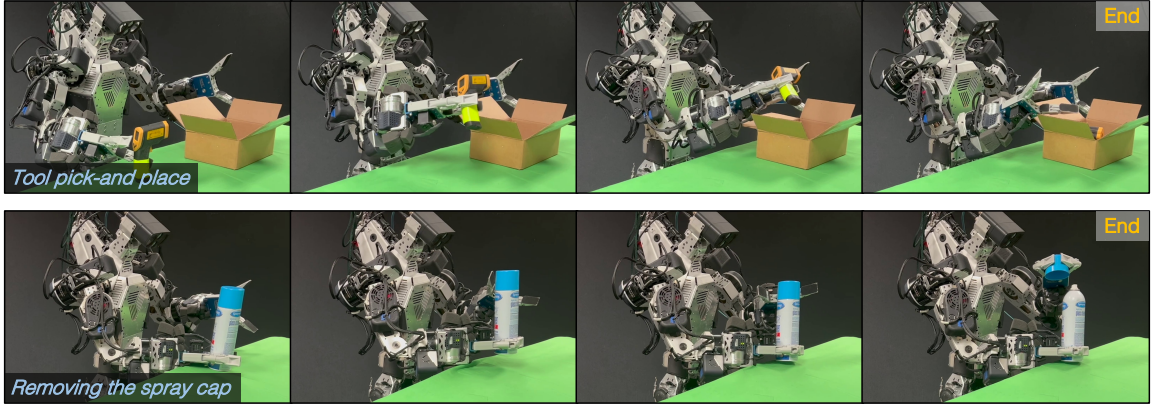


Figure 4.6: Timelapse of deploying TRILL in the real-robot system. We deploy our method to a real humanoid robot to perform two bimanual manipulation tasks. The trained policies perform the *Tool pick-and-place* task successfully in 8 out of 10 trials and the *Removing the spray cap* task in 9 out of 10 trials, consecutively, without the need for rebooting the robot.

demonstrations. Note that real-robot systems, unlike idealized simulation environments, are subject to control latency. We previously discussed the effect of this latency on demonstrator performance in the simulation settings. On average, the control latency with the real system amounts to 122 ± 190 msec. Moreover, error distribution in teleoperation and control latency underlines the importance of data collection and policy deployment within the same settings. The embodiment-specific error and latency distributions pose significant challenges when transferring policies across different hardware platforms, from demonstration to deployment.

Deployment of the Trained Policies Finally, we deploy our policies trained on real-world demonstrations to physical hardware. We aim to validate our policies’ robustness against variations due to controller limitations, sensor inaccuracies, and floating-base dynamics. We focus on the bimanual manipulation tasks described in Section 4.4.1. During evaluation, the robot performed each task 10 times in a row without rebooting. TRILL succeeded in 8 out of 10 trials in the *Tool pick-and-place* task and 9 out of 10 trials in the *Removing the spray cap* task, respectively (see Figure 4.6).

4.5 Conclusion

We introduced TRILL, an effective imitation learning framework for teaching humanoid robots loco-manipulation skills. Through the use of whole-body control, our hierarchical approach robustly executes learned task-space commands while maintaining the dynamical stability of humanoid robots. The control-driven abstraction provided by whole-body control formulates complex humanoid loco-manipulation within a compact domain, where high-dimensional, dynamically constrained motions are represented through structured task-space commands. This reduces the complexity of learning while preserving physical feasibility. Building on this control abstraction, we developed an intuitive VR teleoperation system that enables human operators to provide demonstration trajectories with ease. We also introduced a data-efficient deep imitation learning algorithm to train high-level visuomotor policies within this compact action space. Our experiments have validated the effectiveness of our learning framework in physical simulation and on real-world hardware. In the future, we aim to expand our framework to solve long-horizon tasks and further improve the generalization ability of the learned skills.

Chapter 5: Visuomotor Skill Learning with Learning-Based Control

5.1 Introduction

Legged locomotion in natural environments is an essential capability for enabling real-world robotic applications, such as autonomous inspection, disaster response, and last-mile delivery. Although recent years have witnessed significant advances in low-cost and reliable quadruped hardware, enabling robots to traverse complex environments remains challenging. Robots must simultaneously perceive their surroundings through onboard sensing and execute agile locomotion to navigate safely and effectively.

Aligned with Chapter 4, model-based control architectures provide an effective abstraction for controlling quadrupedal locomotion by leveraging analytical models of robot dynamics and contact interactions. However, to maintain computational tractability, these methods often rely on simplifying assumptions about the robot and its environment. While such assumptions enable efficient optimization, they can introduce model mismatch and become brittle when robots operate in complex or rapidly changing environments. In particular, when robots must perform agile motions in dynamic settings, such as navigating through environments with moving obstacles or human crowds, these assumptions may no longer hold.

Recent work has therefore explored data-driven approaches to improve the robustness and versatility of quadrupedal locomotion. Reinforcement learning, in particular, has demonstrated agile behaviors, such as balancing, running, jumping, and robust walking under environmental uncertainty [22, 129, 234]. Motivated by these successes, pioneering studies have explored incorporating onboard sensory observations into locomotion policies by learning end-to-end visuomotor controllers that map sensory inputs directly to motor commands in simulation [91, 238]. In practice,

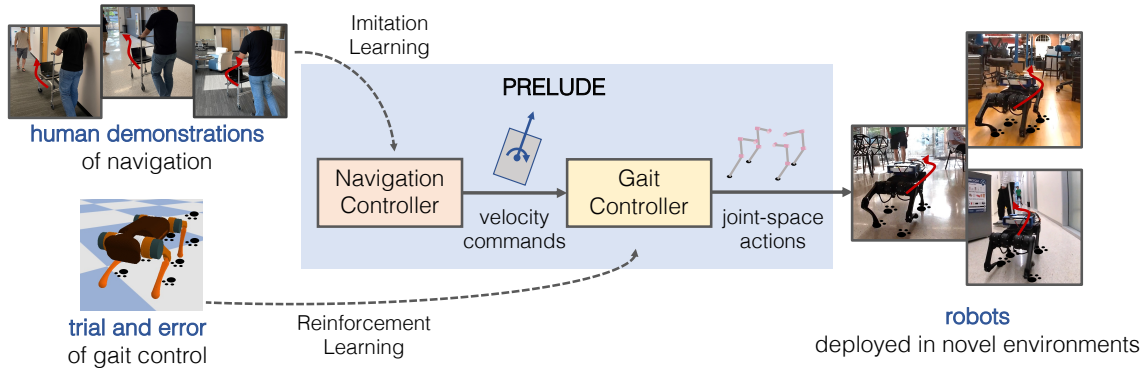


Figure 5.1: Overview of PRELUDE. PRELUDE tackles the problem of perceptive locomotion in dynamic environments. We introduce a control hierarchy where the high-level controller, trained with imitation learning, sets navigation commands and the low-level gait controller, trained with reinforcement learning, realizes the target commands through joint-space actuation. This combination enables us to effectively deploy the entire hierarchy on quadrupedal robots in real-world environments.

however, these approaches face significant challenges. Training such policies requires large amounts of data, and the sample complexity of reinforcement learning increases substantially in cluttered and dynamic environments. Moreover, the reality gap in rendering RGB images, combined with the difficulty of simulating realistic human motion, makes these approaches difficult to deploy reliably in real-world settings.

The goal of this work is to develop a perceptive legged locomotion framework that operates in real-world environments with humans and clutter. Our key idea is to introduce a control abstraction, realized through a low-level gait generation policy that executes target commands, which decomposes locomotion into high-level navigation and low-level execution. This decomposition allows us to leverage the complementary strengths of two learning mechanisms. First, we train the high-level navigation visuomotor policy using imitation learning. By learning from human demonstrations, the policy acquires complex behaviors for traversing crowds and dense clutter, while real-world data mitigates the reality gap. Second, we train the low-level gait generation policy using goal-conditioned reinforcement learning, enabling the discovery of versatile gait behaviors that are difficult to design with conventional control architectures.

We name our method PRELUDE (Perceptive Locomotion Under Dynamic Environments). It consists of a high-level navigation controller and a low-level gait controller (see Figure 5.1). Both controllers are implemented as deep neural networks. The navigation controller, trained with imitation learning, takes in egocentric RGB-D data and generates a desired motion command. To collect data for imitation learning, we design an intuitive steerable cart that can be driven around to collect steering actions in natural environments. A key advantage of this design is that a non-expert human can easily supply demonstrations at scale by pushing the cart around. The navigation controller, trained on data collected on the cart, is able to leverage rich human supervision to perform complex spatiotemporal reasoning and generate dynamic robot motions. Meanwhile, the gait controller, trained with reinforcement learning, learns to achieve variable velocity commands as input. During deployment, it takes as input the target motion command given by the high-level navigation controller and realizes the desired motion.

We evaluate our approach in simulation and with real hardware. In simulation, we demonstrate that our method outperforms state-of-the-art reinforcement learning (RL) methods by 35.4%. Further, we illustrate that our RL locomotion controller exhibits superior performance over conventional model-based methods. The hierarchical design of our method allows us to train the locomotion controller in large-scale simulation and readily transfer it to the real world. Meanwhile, the navigation controller can be trained directly from real-world human demonstration data. By combining these two facets, we can effortlessly deploy the real hardware in a realistic environment.

5.2 Related Work

Quadrupedal Locomotion The control of legged robots has been studied for decades. One of the main challenges is high computational requirements for optimizing whole-body dynamics. In the last decade, the centroidal model, which considers

the robot as a lumped mass to reduce the computational burden, has been widely used by optimization-based methods [21, 29, 113, 229]. However, such model-based approaches are constrained by computational limits and model mismatch. In response to these limits, the integration of model-based methods and data-driven methods has been actively studied [43, 180, 234, 239]. These methods leverage knowledge of robot dynamics encapsulated in model-based methods to improve training efficiency. However, the boost in training efficiency is hindered by the computational costs of the underlying controller. Alternatively, there has been some recent success when training end-to-end policies for locomotion [62, 122]. These policies boast fast computation speeds when inferring actions and have been shown to outperform model-based methods. However, these approaches focus on improving legged robots’ agility rather than how legged locomotion is applied to real tasks.

Perceptive Locomotion of Quadrupedal Robots Legged robots with exteroceptive sensors have been studied for decades. Several model-based approaches exploit exteroceptive sensory modules to reconstruct the shape of surrounding terrains [19, 24, 65, 96, 114, 145, 261]. However, they rely on accurate terrain reconstruction, and the locomotion performance is limited by the precision and speed of reconstruction. Another family of work has employed neural network models to transform raw visual observations into navigation commands [84, 163, 203]. However, they do not account for robot dynamics and instead use a pre-defined gait controller, which is not optimized for the target tasks. There is a growing interest in end-to-end learning of perceptive locomotion [91, 238]. These methods require meticulous simulation training and sim-to-real transfer, which largely limits their applicability to (quasi-)static environments. Inspired by recent advances, our approach uses deep learning models to tackle this problem. Rather than seeking an end-to-end algorithm, we design our method to harness the complementary strengths of imitation and reinforcement learning, allowing the learned model to be deployed to real hardware with minimal domain gaps.

Hierarchical Frameworks for Locomotion Hierarchical frameworks have been used in both model-based and data-driven locomotion. The main advantage of hierarchical frameworks is their ability to break down complicated systems into multiple modules, such that each module can be designed with domain-specific knowledge [21, 29, 113]. In particular, hierarchical approaches to motion generation offer several advantages when using data-driven methods. First, low-level control can be reused with various types of high-level gait planners and vice versa. Second, the overall system can be separated into modules, and individual modules are easier to train and analyze [94, 171, 214]. Our approach also adopts a hierarchical structure. We demonstrate how it gives rise to a practical learning algorithm for perceptive locomotion in dynamic environments.

5.3 Method

We introduce PRELUDE, a hierarchical learning framework for quadrupedal locomotion and navigation in dynamic environments using onboard vision sensing. The key to our approach is to decompose the perceptive locomotion pipeline into a two-level hierarchy consisting of a high-level navigation controller and a low-level gait controller. In this section, we describe our problem formulation and then introduce the hybrid learning scheme used to train the controllers at the two levels of the hierarchy: 1) imitation learning with human supervision for training the navigation controller and 2) reinforcement learning for training the locomotion controller.

5.3.1 Problem Formulation

We model the problem of perceptive legged locomotion as a discrete-time Markov Decision Process $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, R, \gamma, \rho_0)$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P}(\cdot|s, a)$ is the stochastic transition probability, $R(s, a, s')$ is the reward function, $\gamma \in [0, 1)$ is the discount factor, and $\rho_0(\cdot)$ is the initial state distribution. Our goal is to learn a closed-loop visuomotor policy $\pi(a_t|s_t)$ that maximizes

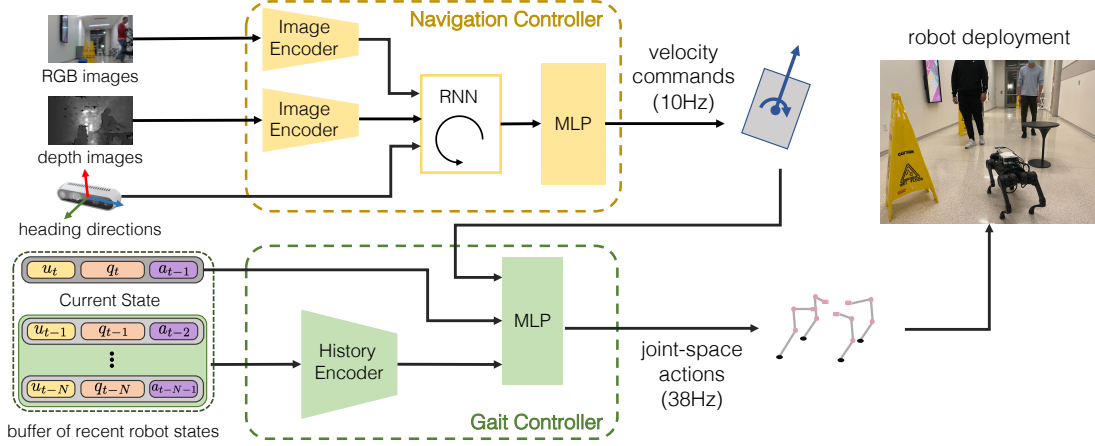


Figure 5.2: Model architecture of PRELUDE. The high-level navigation policy generates the target velocity command u_t at 10 Hz from the onboard RGB-D camera observation and robot heading. The target velocity command is used as input to the low-level gait controller along with the buffer $\mathcal{B}_t$ of velocity commands u_t , recent robot states q_t and previous joint-space actions a_{t-1} . The low-level gait policy predicts the joint-space actions as the desired joint positions at 38 Hz and sends them to the quadrupedal robot for actuation.

the expected return $\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1})]$. In our context, $\mathcal{S}$ is the space of the robot’s sensor data captured by its egocentric camera and proprioceptive sensors, $\mathcal{A}$ is the space of the robot’s joint-space commands, $R(s, a, s')$ is the reward function designed for the locomotion task, and π is a closed-loop policy that we deploy on the robot.

To handle the complexity of perceptive locomotion and train the policy π effectively, we decompose the policy π into a two-level hierarchy. At the high level is a navigation policy π_H that computes the target motion command $u \in \mathbb{R}^2$ as the target linear and angular velocities. At the low level is a gait policy π_L that computes the robot’s joint-space actions to realize target command u from π_H . With this hierarchical abstraction, we can rewrite the whole policy as, $\pi(a_t|s_t) = \pi_L(a_t|s_t, u_t)\pi_H(u_t|s_t)$.

5.3.2 Hierarchical Perceptive Locomotion Policy

PRELUDE consists of a high-level policy π_H for predicting navigation commands and a low-level π_L for generating locomotion gaits (see Figure 5.2). Here we

describe the scheme we used to train both policies and their seamless deployment both in simulation and on the real-robot system.

Imitating Navigation Demonstrations in Dynamic Environments When surrounded by static obstacles and people moving in and out of the scene, the robot must anticipate future motion and subsequently generate high-quality motion plans. By leveraging imitation learning, PRELUDE avoids the difficulty of training an RL policy that must simulate full human walking dynamics and also attempt effective navigation. Instead, we propose to learn the policy π_H for navigation using imitation learning. Through imitating human navigation behaviors, the robot can acquire complex spatial reasoning and motion prediction capabilities in dynamic environments. Meanwhile, this form of high-level demonstration is easy to collect on a hardware platform of a much simpler morphology than the quadruped. We describe our hardware platform of the steerable cart in Section 5.4.1.

The collected demonstration dataset $\mathcal{D}$ consists of state-action pairs $\mathcal{D} = \{(s_i, u_i)\}_{i=1}^N$, where the states are the camera observations from a viewpoint similar to that of the robot’s onboard egocentric camera plus the camera orientation while the actions are the linear and angular velocities estimated by odometry, and N is the total number of data points. With this dataset, we train the navigation policy π_H using a supervised imitation learning algorithm. Specifically, we design a Behavioral Cloning model with recurrent neural networks (RNNs) [148] to capture the temporal information of moving objects in the environments. Furthermore, demonstrations collected from human operators are noisy and multimodal in nature. To capture the multimodality of action distributions while preventing mode collapsing, the policy uses a Gaussian Mixture Model (GMM) to parameterize a distribution of actions [224].

In practice, π_H is implemented as a deep neural network. At each time step, its input consists of the egocentric RGB-D observation and the heading direction of the body. It outputs the target velocity command, consisting of two scalar values for

forward-linear velocity and angular velocity. As the RGB images provide rich task-relevant semantic information and the depth images provide geometry information of surrounding environments, we process RGB images and depth images separately through individual *image encoders*. The target velocity command serves as a suitable representation for specifying high-level navigation behaviors, as it can define highly reactive motions in dynamic environments compared to other common representations, such as position-based commands.

Learning Robust Locomotion Gaits A robust gait controller is key for locomotion in real dynamic environments. To this end, a desirable gait controller should meet the following two requirements: 1) it can track constantly varying velocity commands set by π_H to generate agile walking behaviors for dynamic collision avoidance; 2) it can be effortlessly deployed to real hardware in the presence of dynamics uncertainty. Motivated by recent successes in learning gait controllers in simulation and sim-to-real transfer [43, 122, 129], we use large-scale reinforcement learning in simulation to train our gait controller.

Concretely, PRELUDE learns the policy using the distributed Proximal Policy Optimization (PPO) algorithm [188] in parallel copies of simulated environments. We randomize the target velocity commands (as input to the gait policy π_L) to learn a gait policy to track rapidly changing commands. The policy is optimized over the reward function designed for 1) minimizing the tracking command error and energy, 2) stabilizing the robot body, and 3) encouraging stable foot contacts.

In addition, we introduce two key designs to ensure the learned policy to transfer directly to physical hardware: First, we apply aggressive *domain randomization* [211] to physical parameters in simulation to improve the robustness of our controller while bypassing the complicated system identification process on the real robot. We randomize mass, inertia, friction coefficients of robot joints, and external force perturbation. Second, we use a *buffer of recent robot states* as input to the

policy π_L rather than raw visual observations. For each time t , the input buffer $\mathcal{B}_t$ contains $T + 1$ tuples $\mathcal{B}_t = \{(u_i, q_i, a_{i-1})\}_{i=t-T, \dots, t}$, where u_i is the navigation command at time i , q_i is the proprioceptive state at time i , and a_{i-1} is the joint-space action output of π_L from the previous time step $i - 1$. This design resonates with recent work, RMA [122], which demonstrates that the robot’s recent state-action history can serve as a robust proxy for estimating extrinsics. For perceptive locomotion, using the buffer of robot states as input eliminates the need to handle the reality gap of visual observations from camera sensors. As shown in Figure 5.2, we process the most recent tuple directly while encoding all the other tuples using a *history encoder* implemented by 1D temporal convolution layers. In this way, the policy can weight more on the most recent state tuple, and the temporal convolution operations capture the correlations among the information from different time steps in recent history.

5.4 Experiments

We design our experiments to answer the following three questions: 1) How does PRELUDE’s hierarchical design perform compared to existing methods for perceptive locomotion? 2) Is our learned low-level gait controller more robust than conventional model-based controllers in motion generation? 3) Can PRELUDE be deployed on the quadruped robot in real-world environments?

5.4.1 Experimental Setup

We validate our proposed method for perceptive locomotion on a track with static obstacles and people who may be moving around. A successful episode is one where the robot walks from one end to the other without falling or colliding with obstacles, walls, or people. We design environments of varying difficulty to analyze our model’s performance in simulation and in real hardware with the Unitree A1 quadrupedal robot [9].

Simulation Setup We develop a simulated corridor environment in Bullet Physics [41] with 3 m width and 50 m length. We use this environment to systematically compare our approach to baselines under various conditions. We initialize the scenes with different numbers of static obstacles and walking humans, then categorize the conditions into three difficulty levels:

- **Level 1:** corridor environments with sparsely placed obstacles (average 5) and zero or one person moving.
- **Level 2:** cluttered environments with on average 15 static obstacles and zero or one person. Due to the increase in objects, the robot is more prone to collision.
- **Level 3:** challenging environments with on average 15 obstacles and 4 humans. The crowd raises the uncertainty, so the robot must anticipate the human motions.

All the static obstacles are initialized with 3D assets from the ShapeNet [31] and Google Scanned Objects [56] that cover everyday objects. Human motion is governed by a Gaussian Process-based motion generation method. Further, we add random external perturbations to robots to mimic the environmental uncertainties that tend to exist in real-world scenarios.

Real-World Setup We consider our real-world deployment environment as 15 m tracks with different numbers of static obstacles and walking humans, then categorize the conditions into four types with two annotations:

- **Static (Sparse):** open space with on average 10 large pieces of furniture with enough space to bypass them.
- **Static (Complex):** a narrow office area with more than 10 obstacles where the robot may pass through tight spaces.

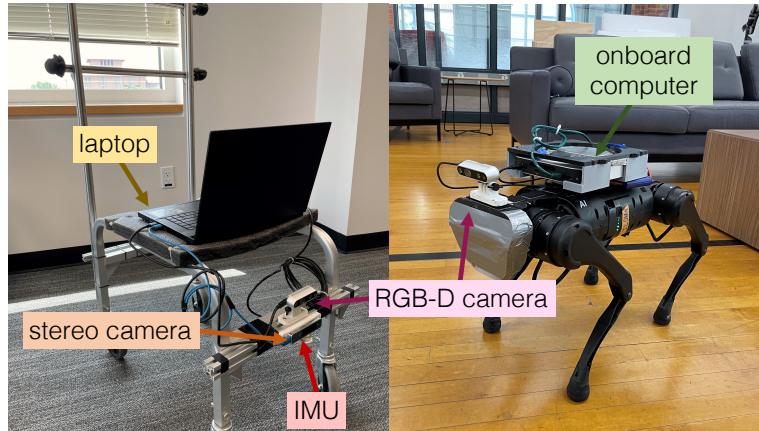


Figure 5.3: Hardware platforms of PRELUDE. A steerable cart designed for collecting human demonstrations of navigation (left) and the Unitree A1 robot with egocentric RGB-D camera mounted at cart height (right). It ensures the navigation policy trained on demonstration data can be directly deployed on the robot.

- **Dynamic (Sparse):** open space with on average 5 humans and 5 static obstacles. The robot has enough space to bypass humans but needs to anticipate humans’ movements.
- **Dynamic (Complex):** a narrow corridor with 1.6 m in length with on average 4 humans and less than 5 obstacles, the most challenging environments where the robot has to pass through narrow space and rapidly react to humans.

Between each trial, obstacles were shuffled around the area and swapped for others. People are instructed to walk in natural ways.

The deployment environments are different from where human demonstration data are collected, with unseen backgrounds, new objects, unexpected human behaviors, and changing light conditions. We mount a RealSense D435 [7] camera on the forehead of the robot for egocentric RGB-D observations. The gait controller predicts desired joint positions at 38 Hz, which are transmitted to the internal Unitree SDK for actuation. The controller is deployed on an onboard Apple Mac mini M1 computer for real-time inference.

Table 5.1: Comparison of PRELUDE and baselines in simulation. We report the average length traversed in meters (total track length: 50 m) and success rate in percentage as the evaluation metric.

Method	<i>Level 1</i>	<i>Level 2</i>	<i>Level 3</i>
DWA [61]	42.2 ± 12.8 (66%)	34.2 ± 18.2 (52%)	30.2 ± 18.3 (40%)
Hierarchical RL [171]	36.4 ± 14.8 (44%)	24.7 ± 17.6 (26%)	28.4 ± 18.9 (38%)
PRELUDE (BC)	32.0 ± 16.0 (32%)	28.1 ± 15.5 (24%)	24.1 ± 14.9 (16%)
PRELUDE (MPC [52])	42.3 ± 14.3 (74%)	38.9 ± 15.4 (60%)	34.0 ± 18.1 (50%)
PRELUDE	44.3 ± 12.4 (80%)	41.6 ± 15.2 (74%)	35.3 ± 18.2 (56%)

Data Collection To collect demonstration data using the quadruped would require a human expert to control the robot in complex environments over long periods of time. This approach is hard to scale due to the mechanical limits of the hardware and the difficulty of control. To ease data collection, we design a steerable cart platform (see Figure 5.3). This cart platform is designed in a non-holonomic morphology, permitting only the forward linear velocity and the angular velocity. Human operators can easily maneuver this cart among static and dynamic obstacles to generate expert data. We mount an Intel RealSense D435 camera on the cart to record RGB-D observations. We further obtain the cart’s linear velocity and yaw-direction angular velocity from the front-mounted Intel RealSense T265 stereo camera. The heading direction can be estimated by the VectorNav VN100 IMU sensor installed on the cart. This way, we eliminate the need for manual labeling of human actions. We collect 250 real-world trajectories within three hours of wall-clock time. To mimic such a steering maneuver in simulation, we use the 3Dconnexion SpaceMouse to provide velocity commands. We collect 80 demonstration trajectories for each difficulty level. For both domains, our datasets consist of the following information: 1) observation data, including RGB-D images and the heading orientation; 2) the forward linear and angular velocity of the cart.

5.4.2 Quantitative Evaluation

Simulation Evaluation We compare PRELUDE against the following baselines:

- **Dynamic Window Approach (DWA):** This baseline uses an online collision avoidance method that computes a cost map from obstacle positions and plans navigation commands based on this map [61]. It uses the navigation commands with our learned gait controller to actuate the robot. This baseline accesses ground-truth positions of all obstacles in the field of view.
- **Hierarchical RL:** This baseline trains the high-level navigation controller with reinforcement learning (RL) using the same distributed PPO algorithm. It uses the navigation command as the action space and invokes our learned gait controller. The reward is defined as the progress of moving forward at each step. This strategy has been explored in prior work [157, 171, 231], which uses low-level action primitives to level up the action abstraction for RL.
- **PRELUDE (MPC):** This baseline uses a conventional Model Predictive Control (MPC) baseline [52] instead of our learned gait controller. We assume the MPC baseline accesses ground-truth body velocity.
- **PRELUDE (BC):** This baseline is a variant of our final model, which does not include the recurrent neural networks in the navigation policy architecture. All the other components remain the same.

To quantitatively evaluate each model, we use the success rate of completing the track and the travel distance, averaged across all test trials. The distance metric allows us to measure partial progress made by the baselines when they struggle to complete the entire task. To evaluate all the models consistently and control the environment setup, we generate 50 randomized scenes for each difficulty level and use the same set of 50 scenes for evaluating each model. We present the average length traversed and success rate in Table 5.1.

Table 5.1 suggests that PRELUDE achieves the overall best performance in all three difficulty levels. Results of the DWA baseline imply that all three levels of

environments require non-trivial collision avoidance behaviors. Meanwhile, the Hierarchical RL baseline shows that decoupling gait control and navigation allowed the robot to traverse in simple and static environments. However, RL alone struggles to discover sophisticated navigation strategies, especially for handling dynamic obstacles. Results of PRELUDE (BC) indicate that the lack of temporal modeling leads to inferior performances, especially in *Level 3*, where our final model traverses a 44% longer average distance than the BC baseline. We hypothesize that the temporal information, harnessed by our recurrent policies, plays a critical role in reasoning about moving people and anticipating their future motions.

Results of PRELUDE (MPC) indicate that our learned navigation controller is compatible with different types of gait controllers. Even though PRELUDE (MPC) takes advantage of the robot’s dynamic models for MPC, our RL-based gait achieved higher performances overall. These results imply that our learned gait controller exhibits high robustness for agile navigation in dynamic environments.

Analysis of Gait Controllers We have shown PRELUDE outperforms baseline approaches in the perceptive locomotion task and further analyze the quality of the learned gait controller. We send the robot velocity commands computed from the setpoints of a pre-defined position trajectory. At each time step, the velocity command is computed by a simple PD controller based on the next setpoint and the robot’s current position. We set desired velocity commands with a PD controller because unit testing on the locomotion controllers is performed in a closed-loop manner to mimic the integration with the navigation controller. Furthermore, realistic navigation tasks require fast convergence speed and stability with respect to velocity error.

We design trajectories with diverse shapes and curvatures to evaluate the agility of our learned gait controller against the MPC baseline [52]. To evaluate that our gait policy handles aggressive changes in linear velocity, we design the below linear trajectories of different setpoint distances:

Table 5.2: Evaluation of gait controllers. We report the tracking errors in meters (lower the better) and success rate in percentage (higher the better) on 20 trials of tracking each pre-defined trajectory.

	MPC [52]	Gait Controller (Ours)
<i>x</i> -axis Linear	0.26 ± 0.03 (50%)	0.20 ± 0.02 (100%)
<i>x</i> -axis Sine	0.28 ± 0.05 (60%)	0.24 ± 0.02 (95%)
<i>x</i> -axis Step	0.29 ± 0.04 (45%)	0.26 ± 0.04 (90%)
Sine Trajectory	0.27 ± 0.04 (55%)	0.23 ± 0.03 (90%)
Zig-zag Trajectory	0.26 ± 0.03 (55%)	0.24 ± 0.02 (85%)

- ***x*-axis Linear:** a linear trajectory of setpoints with constant distances of 0.7 m per second.
- ***x*-axis Sine:** a linear trajectory of setpoints with distances varying as sine functions in the range of 0.4 to 1.0 m per second.
- ***x*-axis Step:** a linear trajectory of setpoints with distances varying as step functions with values of 0.5, 0.7, and 1.0 m per second.

To show that our gait policy handles large changes in the yaw rate, we designed the additional trajectories below:

- **Sine Trajectory:** a sine-like trajectory of setpoints with constant distances. The yaw direction of the trajectory changes in the range of 0.0 and 1.4 rad.
- **Zig-zag Trajectory:** a zig-zag trajectory of setpoints with constant distances where the direction of the path is given as step functions, as -0.4 to 0.4 rad.

We report two evaluation metrics: 1) the quality of trajectory tracking, computed as the average tracking error between the ground-truth trajectories and the robot’s realized trajectories, and 2) the robustness of the controller, computed by evaluating the success rate in tracking the trajectories below the error limit of 1 m without failing within 30 seconds. Table 5.2 presents the results. Our gait controller, learned with reinforcement learning, realizes the trajectories more accurately and robustly than



Figure 5.4: Real-robot deployment in novel environments. PRELUDE can be deployed in unseen human-centered environments with static and dynamic obstacles. It exhibits robust locomotion behaviors with onboard visual perception.

the MPC baseline. In particular, we observe that the MPC method regulates body acceleration and changes the body motion very smoothly when the changes to the velocity command are small. However, when commands change more aggressively, simplifications of inertia terms to convexify the MPC cause failure of the robot. In contrast, we observe that our RL-based controller can safely track aggressively changing commands. These attributes are suitable for performing the given task of navigation amongst dynamic obstacles where changes need to be rapid. Furthermore, our controller has a significantly lower lateral drift than MPC when tracking velocity commands.

Real-Robot Evaluation Finally, we demonstrate that PRELUDE can be deployed on the real robot. We compare it with our self-baseline PRELUDE (A1 Default Gait), a variant of our final model, using the robot’s default model-based controller instead. The comparison shows the effectiveness of PRELUDE in real-world deployment. In particular, 1) our navigation controller can be deployed with other types of gait controllers, and 2) our learned gait controller outperforms model-based controllers for locomotion in cluttered and dynamic environments.

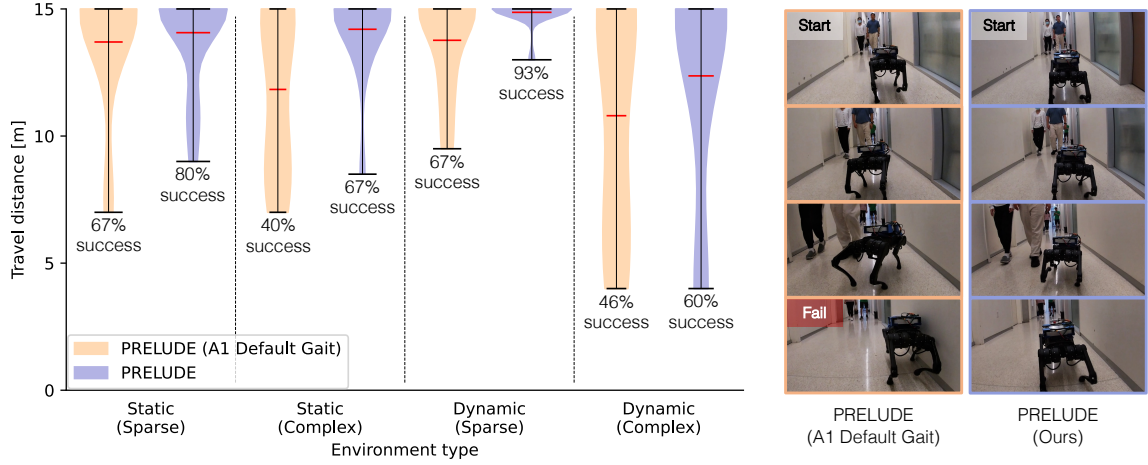


Figure 5.5: Real-robot experiments. (Left) We perform real-world trials where the robot traverses 15m-length tracks in different configurations. The plot shows distributions of travel distances in meters. The black and red lines indicate the ranges and the means of traverse length, respectively. (Right) We observed that PRELUDE (A1 Default Gait) drifts aggressively after a high-speed turning and collides into the wall, while PRELUDE turns rapidly to bypass the walking crowd and completes the trial successfully.

To quantitatively evaluate each model, we report the success rate of completing the track and the distance that the robot traverses before a collision or reaching the end of the track, averaged across all test trials, as shown in Figure 5.5. To evaluate the models consistently and control the environment setup, we generate 15 randomized scenes for each type of scenario. The result shows that both PRELUDE and PRELUDE (A1 Default Gait) can navigate robustly in static and dynamic environments. It implies that our navigation controller works well with different gait controllers on real robots. Furthermore, PRELUDE achieves overall longer travel distances and higher success rates considering all four types of environments. We note that the default gait controller suffers from drifts at high-speed turning, often resulting in collisions with obstacles on the side. We hypothesize that model errors from the simplified dynamic model and inaccurate state estimation caused these issues. Our learned gait controller exhibits less drift when turning at high rates, and the robot shows higher agility in cluttered and dynamic environments.

5.5 Conclusion

We introduced PRELUDE, an effective method for learning perceptive locomotion controllers for quadrupedal robots to traverse real-world dynamic environments. Our method combines the complementary strengths of imitation learning and reinforcement learning through a hierarchical design that decomposes the locomotion problem into high-level navigation and low-level gait generation. This provides a control abstraction that enables effective behavior generation even when model-based control architectures are not available. With this low-level gait controller, imitation learning can be applied to train higher-level navigation policies without requiring direct robot teleoperation. To collect human navigation demonstrations in complex environments, we designed a steerable cart platform that allows humans to naturally demonstrate navigation behaviors. The collected dataset is then used to train the high-level navigation policy. Our work focuses on flat and indoor environments where human steering actions can be conveniently collected with the wheeled platform. For future work, we would like to extend our wheeled carts to more sophisticated mechanical designs to collect human datasets for traversing rough terrains in outdoor environments.

Chapter 6: Capturing Task Intent from Demonstration through Object-Centric Representation

6.1 Introduction

Deploying robots for everyday tasks requires reliable autonomy in natural environments. With advances in hardware and growing commercial availability, diverse platforms, including humanoid robots, are increasingly being deployed in human living and working spaces. Despite this progress, achieving robust autonomy in unstructured settings remains a fundamental challenge. Recent work has explored deep imitation learning for complex manipulation [14, 150, 191], as discussed in previous chapters. However, these approaches typically rely on large-scale demonstrations, requiring substantial human effort and remain tightly coupled to specific robot embodiments. This limits scalability and transfer across different systems.

In contrast, humans can acquire new skills from only a few observations, despite embodiment differences between the demonstrator and the learner. Enabling robots to learn directly from visual demonstrations would significantly reduce supervision and move toward training robotic foundation models from large-scale human video data. In this work, we study humanoid manipulation from human demonstrations in a one-shot setting, where a robot learns a skill from a single video [71, 79, 260], enabling rapid skill acquisition with minimal supervision. However, learning from limited visual data presents key challenges: demonstrations lack action labels and do not directly correspond to the robot’s embodiment, requiring the system to infer task intent and generalize to new situations.

Prior work on video imitation infers robot actions by reconstructing future object motion trajectories [79, 260]. While effective for single-arm manipulators, these approaches become computationally prohibitive for humanoid robots due to their

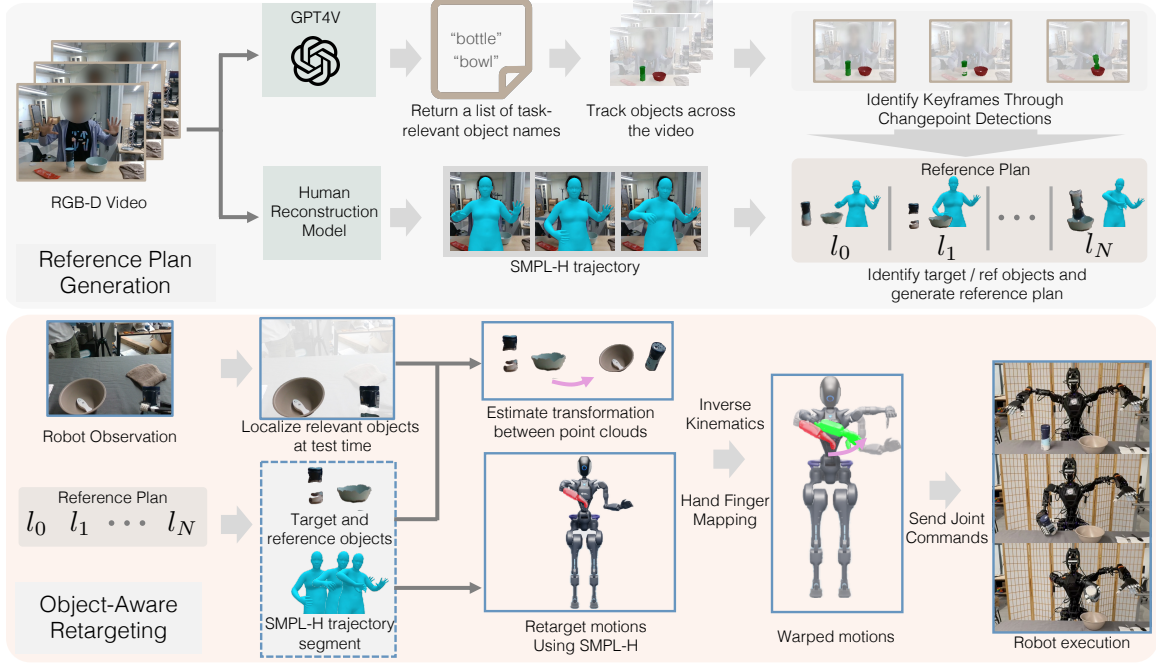


Figure 6.1: Overview of OKAMI. OKAMI is a two-stage method that enables a humanoid robot to imitate a manipulation task from a single human video. In the first stage, OKAMI generates a reference plan using GPT-4V [161] and large vision models. In the second stage, OKAMI follows this plan by retargeting human motions to the humanoid with object awareness. The retargeted motions are converted into a sequence of robot joint commands for execution.

high degrees of freedom and joint redundancy [13]. At the same time, the structural similarity between the human body and humanoid robots enables direct retargeting of human motion [46, 67]. However, existing retargeting methods primarily focus on free-space body motion [35, 39, 50, 86, 155] and lack awareness of object interactions required for manipulation. To address this limitation, we introduce object-aware retargeting, which incorporates object context into the retargeting process, enabling humanoid motions that capture task-relevant interactions and adapt to varying object configurations.

To this end, we introduce OKAMI (Object-Aware Kinematic Retargeting for Humanoid Imitation), an object-aware retargeting method that enables a bimanual humanoid with dexterous hands to imitate manipulation behaviors from video demonstrations. OKAMI adopts a two-stage pipeline to retarget human motion for task

execution under varying initial conditions. The first stage processes the video to extract a reference manipulation plan. The second stage uses this plan to synthesize humanoid motions via retargeting that adapts to object configurations in the target environment.

OKAMI is built on two key designs. First, an open-world vision pipeline identifies task-relevant objects, reconstructs human motion from video, and localizes objects at test time, enabling adaptation to different backgrounds and novel object instances within the same category. Second, a factorized retargeting process separates body motion and hand pose. We retarget body motion in the task space from the reference plan, then warp the trajectory based on object locations. Body joint trajectories are obtained via inverse kinematics (IK), while finger joint angles are mapped directly to the dexterous hands to reproduce hand-object interactions. This object-aware retargeting enables OKAMI to generalize across diverse object layouts and cluttered scenes.

We evaluate OKAMI on human video demonstrations spanning diverse manipulation tasks with rich object interactions, including picking, placing, pushing, and pouring. The proposed object-aware retargeting achieves an average task success rate of 71.7% across all tasks, outperforming the ORION baseline [260] by 58.3%. We further validate OKAMI on humanoid hardware, demonstrating strong spatial and visual generalization in natural environments with unseen object layouts, varying backgrounds, and novel object instances. These results show that OKAMI enables humanoid robots to imitate dexterous manipulation behaviors from a single human video and deploy the learned skills in real-world settings.

6.2 Related Work

Imitation Learning for Robot Manipulation Imitation learning has significantly advanced vision-based robot manipulation with high sample efficiency [32, 53, 101, 137, 147, 148, 217, 221, 222, 243, 246, 258]. Prior works have shown that robots

can learn visuomotor policies to complete various tasks with just dozens of demonstrations, ranging from long-horizon manipulation [147, 221, 258] to dexterous manipulation [137, 222, 246]. However, collecting demonstrations often requires domain expertise and high costs, creating challenges to scale. Another line of work focuses on one-shot imitation learning [32, 53, 101, 217, 243], yet it demands excessive data collection for meta-training tasks. Recently, researchers have looked into a new problem setting of imitating from a single video demonstration [71, 79, 260], referred to as “open-world imitation from observation” [260]. Unlike prior works that abstract away embodiment-specific motions due to kinematic differences between the robot and the human, we exploit embodiment motion information owing to the kinematic similarity between humans and humanoids. Specifically, we introduce *object-aware retargeting* that adapts human motions to humanoid robots.

Motion Retargeting Motion retargeting has wide applications in computer graphics and 3D vision [67], where extensive literature studies how to adapt human motions to digital avatars [98, 142, 173]. This technique has been adopted in robotics for recreating human-like motions on humanoid or anthropomorphic robots through various retargeting methods, including optimization-based approaches [86, 120, 155, 170], geometry-based methods [136], and learning-based techniques [35, 39, 78]. However, in manipulation tasks, these retargeting methods have been used within teleoperation systems, lacking a vision pipeline for automatic adaptation to object locations. OKAMI integrates the retargeting process with open-world vision, endowing it with object awareness so that the robot can mimic human motions from video demonstrations and adapt to object locations at test time.

6.3 Method

We consider the problem of learning humanoid manipulation policies from human video demonstrations. In this setting, demonstrations are provided as third-

person video streams of humans performing tasks, and the policy is conditioned on a few such demonstrations. OKAMI addresses this problem through object-centric representations that capture task-relevant structure and enable policy learning from limited demonstrations.

6.3.1 Problem Formulation

We formulate the problem as a discrete-time Markov Decision Process defined by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, R, \gamma, \mu)$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ the action space, $\mathcal{P}(\cdot \mid s, a)$ the transition dynamics, $R(s)$ the reward function, $\gamma \in [0, 1)$ the discount factor, and μ the initial state distribution. In our context, the state $s \in \mathcal{S}$ consists of raw visual observations capturing both the robot and object states, and the action $a \in \mathcal{A}$ corresponds to motion commands for the humanoid robot. Given a human demonstration video V , the objective is to learn a policy $\pi(a_t \mid s_t, V)$ that enables the humanoid robot to reproduce the demonstrated task. We assume a sparse reward function that provides a binary signal upon task completion. The goal is to maximize the expected task success rate over initial configurations sampled from μ , while generalizing to novel object configurations, environments, and visual conditions at test time. This setting requires the policy to infer task-relevant behaviors from human demonstrations and adapt them to the humanoid embodiment, preserving task intent despite differences in morphology and scene layout.

6.3.2 Reference Plan Generation

To enable object-aware retargeting, OKAMI first generates a reference plan for the humanoid robot to follow. Plan generation requires identifying task-relevant objects and understanding how humans manipulate them.

Identifying and Localizing Task-Relevant Objects To imitate manipulation tasks from videos V , OKAMI must identify the task-relevant objects for interaction.

While prior methods rely on unsupervised approaches with simple backgrounds or require additional human annotations [26, 88, 206, 259], OKAMI uses an off-the-shelf vision-language model, GPT-4V, to identify task-relevant objects in V by leveraging its internalized commonsense knowledge. Concretely, OKAMI obtains object names by sampling RGB frames from the video demonstration V and prompting GPT-4V with their concatenation. Using these object names, OKAMI employs Grounded-SAM [140] to segment the objects in the first frame and tracks them throughout the video using a video object segmentation model, Cutie [34]. This process enables OKAMI to localize task-relevant objects in V , forming the basis for subsequent steps.

Reconstructing Human Motions To retarget human motions to a humanoid robot, OKAMI reconstructs motion trajectories from the video demonstration V . We adopt an improved version of SLAHMR [240], an iterative optimization method for reconstructing human motion sequences. While SLAHMR assumes flat hands, our extension optimizes the hand poses of the SMPL-H model [179], initialized using hand pose estimates from HaMeR [167]. This modification enables joint optimization of body and hand poses from monocular video. The output is a sequence of SMPL-H parameters capturing full-body and hand motions, which OKAMI uses to retarget human movements to humanoids, as presented in Section 6.3.3. Moreover, the SMPL-H model represents human poses across demographic variations, facilitating the mapping from human demonstrators to the humanoid.

Generating a Plan from Video Having identified task-relevant objects and reconstructed human motions, OKAMI generates a reference plan from the video demonstration V for the robot to execute each subgoal. OKAMI identifies subgoals by performing temporal segmentation on V . We first track keypoints using CoTracker [106] and detect changes in keypoint velocities to determine keyframes, which correspond to subgoal states. For each subgoal, we identify a target object (in motion due to manipulation) and a reference object (providing a spatial frame for

the target object’s movement through contact or non-contact relations). The target object is determined based on the average keypoint velocity per object, while the reference object is identified using geometric heuristics or semantic relations predicted by GPT-4V.

Given the subgoals and associated objects, we construct a reference plan $l_0, l_1, \dots, l_N$, where each step l_i corresponds to a keyframe and includes the point clouds of the target object o_{target} , the reference object $o_{\text{reference}}$, and the SMPL-H trajectory segment $\tau_{t_i:t_{i+1}}^{\text{SMPL}}$. If no reference object is required (e.g., during grasping), $o_{\text{reference}}$ is null. Point clouds are obtained by back-projecting segmented objects from RGB images using depth images [256].

6.3.3 Object-Aware Retargeting

Given a reference plan from the video demonstration V , OKAMI enables a humanoid robot to imitate the task in V . The robot executes each step l_i by localizing task-relevant objects and retargeting the corresponding SMPL-H trajectory segment to the humanoid. The retargeted trajectories are then converted into joint commands via IK. This process repeats until all steps are completed, and success is evaluated based on task-specific criteria.

Localizing Objects at Test Time. To execute the plan in the test environment, OKAMI localizes task-relevant objects from the robot’s observations and extracts their 3D point clouds to track object states. By focusing on task-relevant objects, OKAMI generalizes across diverse visual conditions, including varying backgrounds and novel instances of task-relevant objects.

Retargeting Human Motions to the Humanoid. A key aspect of object awareness is adapting motions to new object locations. After localizing task-relevant objects, we employ a factorized retargeting process that synthesizes arm and hand motions separately. OKAMI first adapts arm motions to align the hands within an

object-centric coordinate frame. It then retargets finger configurations to mimic how the demonstrator interacts with the objects.

Concretely, we map human body motions to the humanoid’s task space, scaling and adjusting trajectories to account for differences in size and proportion. OKAMI then warps the retargeted trajectory so that the robot’s arms reach the new object locations. We consider two cases for trajectory warping: when the relational state between the target and reference objects remains unchanged, and when it changes. In the former, the trajectory is warped based on the target object location; in the latter, it is warped based on the reference object location.

After warping, we use IK to compute a sequence of arm joint configurations, balancing position and orientation objectives to maintain natural postures. In parallel, we retarget human hand poses to the robot’s finger joints, enabling fine-grained manipulation. This process yields a full-body joint trajectory for execution. Since arm motion retargeting is affine, it naturally scales across demonstrators with different body proportions. By adapting arm trajectories to object locations and retargeting hand poses independently, OKAMI generalizes across diverse spatial configurations.

6.4 Experiments

Our experiments are designed to answer the following research question: 1) Is OKAMI effective for a humanoid robot to imitate diverse manipulation tasks from single videos of human demonstrations? 2) Is it critical in OKAMI to retarget the body motions of demonstrators to the humanoid robot instead of only retargeting based on object locations? 3) Can OKAMI retain its performances consistently on videos demonstrated by humans of diverse demographics?

6.4.1 Experimental Setup

We use a Fourier *GR1* robot [5] as our hardware platform, equipped with two 6-DoF Inspire dexterous hands and a RealSense D435i camera [7] for video recording

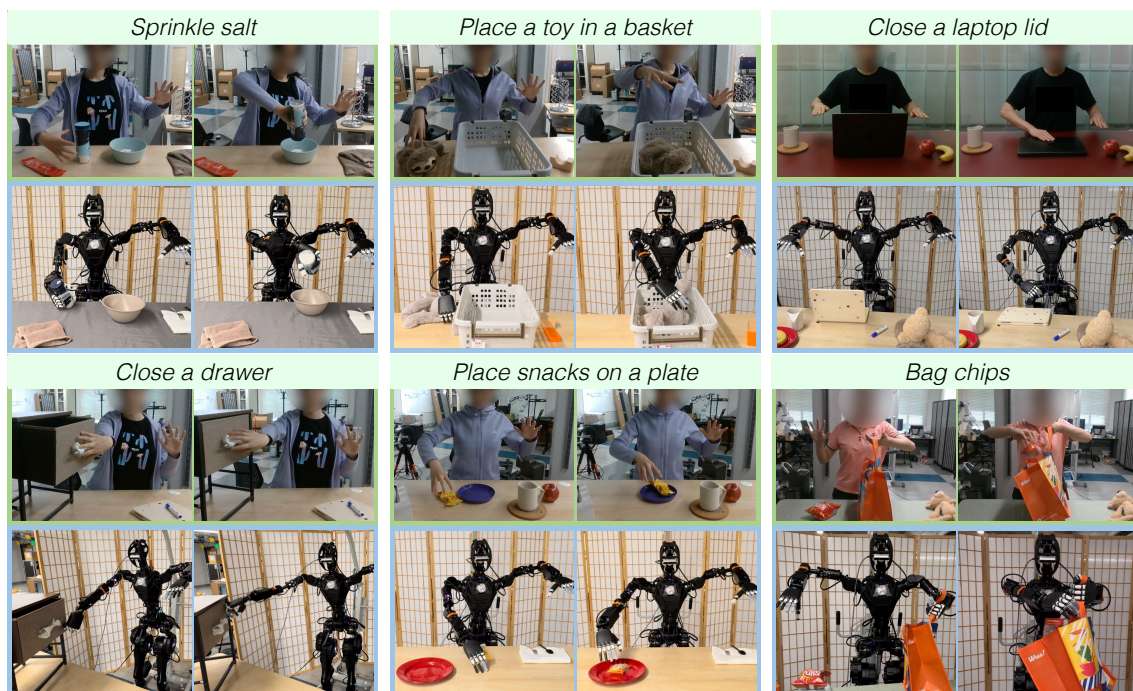


Figure 6.2: Timelapse of human demonstrations and robot rollouts. For each task, the top row shows the initial and final frames of a human demonstration, and the bottom row shows the corresponding robot rollout.

and test-time observation. We implement a joint position controller operating at 400 Hz. To avoid jerky motions, joint position commands are computed at 40 Hz and interpolated to 400 Hz trajectories.

We design six tasks that cover a diverse range of manipulation behaviors in the experiments:

1. ***Sprinkle salt:*** dispensing salt into a bowl;
2. ***Place a toy in a basket:*** placing a deformable toy inside a basket;
3. ***Close a laptop lid:*** rotating a laptop lid to a closed configuration;
4. ***Close a drawer:*** pushing a drawer until fully closed;
5. ***Place snacks on a plate:*** placing a bag of snacks on a plate;

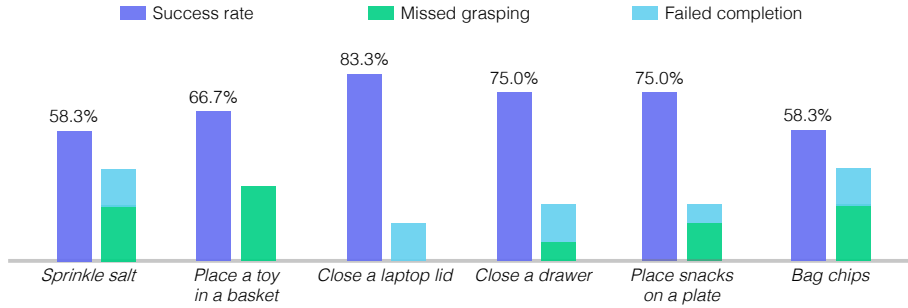


Figure 6.3: Task performance and failure analysis of OKAMI. We report evaluation across six tasks, showing success rates and failure breakdowns by mode.

6. ***Bag chips***: inserting a chip bag into an open deformable shopping bag.

Place a toy in a basket and *Place snacks on a plate* involve pick-and-place behaviors with everyday objects; *Sprinkle salt* captures pouring behavior; *Close a drawer* and *Close a laptop lid* require interaction with articulated objects, a common scenario in daily environments; and *Bag chips* involves dexterous, bimanual manipulation with multiple subgoals. While our primary focus is on real-robot experiments, we also implement *Sprinkle salt* and *Close a drawer* in simulation for reproducibility.

Throughout the experiments, we run 12 trials per task. Object locations are randomly initialized within the intersection of the robot camera’s field of view and the reachable workspace of the humanoid arms. Tasks are performed on a tabletop with multiple objects, including both task-relevant and distractor objects. We further evaluate object generalization on *Place snacks on a plate*, *Place a toy in a basket*, and *Sprinkle salt* by replacing the plate, snack bag, plush toy, and bowl with different instances of the same category.

6.4.2 Quantitative Evaluation

To demonstrate the effectiveness of OKAMI, we evaluate its policies across all tasks, covering diverse behaviors such as pick-and-place, pouring, and interaction with articulated objects. The results are presented in Figure 6.3. In our experiments, object locations are randomly initialized, requiring the robot to adapt to varying

spatial configurations. These results demonstrate the ability of OKAMI to generalize across different visual and spatial conditions.

6.4.3 Impact of Body Motion Retargeting

To evaluate whether retargeting human body motion is critical in OKAMI, we compare it against ORION on two representative tasks, *Place snacks on a plate* and *Close a laptop lid*. We use ORION [260] as a baseline. Since ORION was originally designed for parallel-jaw grippers, it is not directly applicable in our setting. We therefore adapt it with minimal modifications. We estimate palm trajectories from SMPL-H motion and warp them based on object locations. The warped trajectories are then used for IK to compute robot joint configurations. In this comparison, the key difference is that ORION does not condition on human body pose, whereas OKAMI does.

On real-robot tasks, OKAMI achieves success rates of 75.0% and 83.3% on *Place snacks on a plate* and *Close a laptop lid*, respectively, while ORION achieves 0.0% and 41.2%. We further evaluate both methods in simulation on *Sprinkle salt* and *Close a drawer*. In these settings, OKAMI achieves 82.0% and 84.0% success rates, whereas ORION achieves 0.0% and 10.0%. Most failures of ORION arise from unreliable grasping and manipulation behaviors. For example, in *Place snacks on a plate*, ORION attempts side grasps instead of the top-down grasp observed in the demonstration. It also fails to produce sufficient wrist rotation for tasks such as pouring. These failures stem from the lack of embodiment-aware information, as ORION does not incorporate human body motion. Overall, the superior performance of OKAMI highlights the importance of retargeting human body motion when learning from video demonstrations.

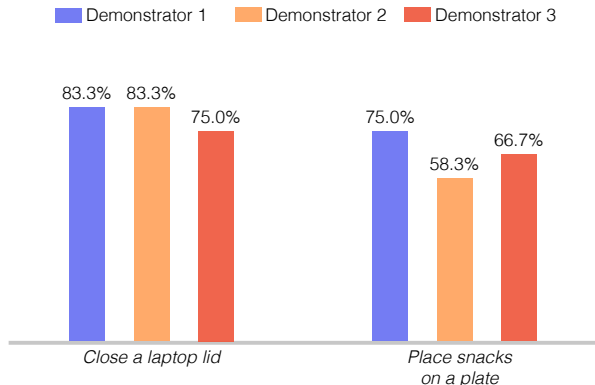


Figure 6.4: Task performance of OKAMI under demonstrator variability. We report evaluation across different demonstrators, showing performance variation. Demonstrator 1 provides the primary data used in Section 6.4.2.

6.4.4 Robustness to Demonstrator Variability

To evaluate whether OKAMI maintains consistent performance across videos from human demonstrators with diverse demographics, we conduct a controlled study using recordings from multiple individuals. As in the previous experiment, we evaluate OKAMI on the *Place snacks on a plate* and *Close a laptop lid* tasks. Results are shown in Figure 6.4. For *Close a laptop lid*, performance shows no statistically significant variation across demonstrators. For *Place snacks on a plate*, performance remains above 50% across all cases, although the worst-performing policy is 16.7% lower than the best. Further inspection reveals that one demonstrator performs the task significantly faster, leading to noisier motion estimates during human reconstruction. Overall, OKAMI maintains reasonably strong performance across diverse demonstrators, though improving robustness to variations in motion speed remains an area for future work.

6.5 Conclusion

We introduced OKAMI, a framework that enables a humanoid robot to imitate manipulation behaviors from a single RGB-D video demonstration of a human.

At its core is an object-aware retargeting strategy that maps human motions onto the humanoid while adapting them to target object locations. As a result, even when demonstrations are closely tied to a specific embodiment, OKAMI captures task intent and enables adaptive skill transfer across embodiments. The framework consists of two stages. The first generates a reference manipulation plan from the video demonstration. The second performs motion retargeting, transferring arm motions in the task space while retargeting finger motions in the joint configuration space. This design allows the robot to reproduce the demonstrated behavior while respecting embodiment-specific constraints. Experimental results demonstrate the effectiveness of OKAMI, showing consistent generalization of the learned skills across diverse object configurations.

Chapter 7: Embodiment-Aware Trajectory Generation through Configuration-Space Representation

7.1 Introduction

In the previous chapter, task intent was captured from demonstrations through object-centric representations. In particular, extracting keyframes from demonstrations provides an effective way to adapt behaviors across different robots by focusing on the essential task structure. However, differences in robot morphology can still limit behavior transfer. Variations in kinematic structure change the reachable state space of the robot, while differences in geometry and joint limits alter the set of feasible trajectories. As a result, representations that rely solely on object states in the workspace are insufficient for realizing tasks across diverse embodiments. To address this limitation, this chapter introduces a representation that models not only object states in three-dimensional space but also feasibility in an embodiment-specific configuration space (C-space). By explicitly accounting for the constraints imposed by the robot’s morphology, the representation enables the generation of executable trajectories for different robotic systems.

Motion planning provides a principled framework for handling such embodiment-specific constraints. The goal of motion planning is to find a smooth and collision-free path in a high-dimensional C-space. Classical motion planning algorithms typically use either sampling-based methods [107, 125] or optimization-based methods [177, 187] to address motion planning across various domains. However, in high-dimensional C-spaces with narrow passages, sampling-based methods incur high computational costs due to large search spaces and a small volume of solutions. While optimization-based methods can serve as an alternative, such methods are sensitive to initialization and may become stuck in local optima, often failing to find a feasible

path. Consequently, both approaches have limitations when dealing with complex motion planning problems under restricted computational resources. These limitations are critical for building robot autonomy, because it requires generating high-quality trajectories in real time.

Recent works leverage generative models to directly learn trajectory distributions instead [30, 87, 95]. By casting motion planning as sampling from a learned distribution, these models can efficiently generate trajectories within a consistent computational budget. However, they often struggle to generalize to new, complex C-spaces, resulting in high collision rates in the generated trajectories because most of these approaches use the workspace as input to neural networks instead of the C-space. Instead, we propose representing the environment in terms of key configurations [112], a sparse set of task-related configurations from prior motion planning data. The resulting model no longer needs to learn a generalizable mapping between workspace and C-space obstacle representations, improving generalization and reducing training complexity.

Another challenge is designing a training objective for generative models tailored to motion planning. Existing diffusion models for motion planning use DDPM-based losses [30, 82, 95] that focus on reconstruction quality [183]. However, this reconstruction objective does not account for underlying task constraints, resulting in degraded performance on tasks that require precise outputs and complex constraints [66]. To overcome this, we incorporate TrajOpt-inspired motion-planning costs [187] directly into the training pipeline of a diffusion model, minimizing trajectory-optimization costs associated with collision avoidance and trajectory smoothness. As a result, the model learns to generate smooth and collision-free trajectories. Further, to ensure that generated trajectories satisfy hard constraints such as collision avoidance, we feed the diffusion model’s outputs as initial solutions for subsequent trajectory optimization.

We name our framework PRESTO (Planning with Environment Representation,

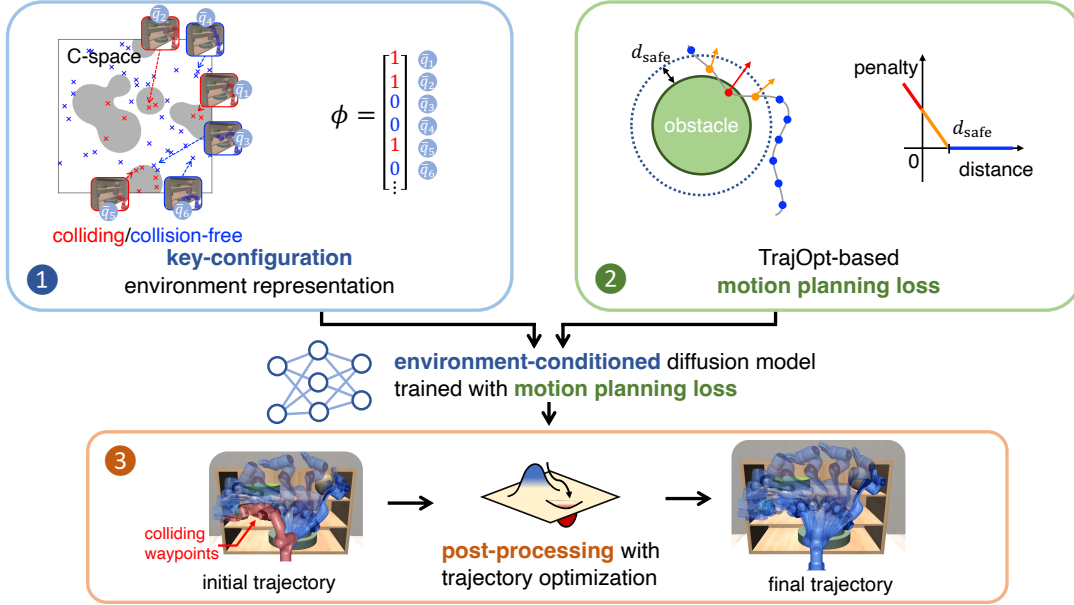


Figure 7.1: Overview of PRESTO. PRESTO aims to generate collision-free trajectories in complex, unseen C-spaces. First, we approximate these C-spaces using key configurations from prior data and generate trajectories based on this representation. A conditional diffusion model, trained with a motion planning loss, provides initial solutions that are subsequently refined through trajectory optimization.

(Sampling, and Trajectory Optimization). Figure 7.1 summarizes our framework: 1) an environment representation based on key configurations (blue block); 2) a training pipeline for a diffusion model that directly integrates motion-planning costs (green block); and 3) a diffusion-based sampling-and-optimization framework for motion planning, where the diffusion model provides initial trajectories for trajectory optimization (orange block). We evaluate PRESTO in simulated environments where a robot operates in a fixed scene populated with randomly shaped and arranged objects. The results show that the synergy between the high-quality trajectory priors generated by our diffusion model and the trajectory optimization post-processing efficiently generates collision-free trajectories in narrow passages within a limited computational budget.

7.2 Related Work

Several approaches learn to guide motion planning by complementing motion planners with learned models for collision checking [45, 48, 89, 109, 152, 255] or for sampling promising configurations [12, 90, 93, 111, 123, 124, 242, 249].

A body of work has focused on learning collision checkers to expedite motion planning. One body of work progressively learns collision models in a given environment as Gaussian mixtures [89] or kernel perceptrons [48, 255], while others use deep neural network models trained on large offline datasets. Kew et al. [109] train a neural network to estimate the clearance of robot configurations in a fixed environment. Danielczuk et al. [45] generalize collision predictions to diverse environments by training a neural network to estimate collision states from object and scene point clouds. Recently, Murali et al. [152] extend neural collision detectors to a partially observable setting. Our approach can integrate with these collision-checking methods to further enhance planning speed.

Another line of work focuses on sampling promising configurations. Some methods derive explicit distributions via kernel density estimation [12, 93], while others use neural networks, in which sequence-based models such as LSTMs [124] or a rejection-sampling policy [249] learn the distribution of collision-free configurations based on the history of collision states at previously sampled configurations. Alternatively, recent works propose using generative models to govern the sampling process. Qureshi et al. [176] map point-cloud inputs to the next sampling configuration, stochastically applying dropout in the interim layers to generate diverse samples; Yu et al. [242] use Graph Neural Networks to identify promising regions of a roadmap; and Ichter et al. [90] along with Kumar et al. [123] employ Conditional Variational Autoencoders (CVAEs) to sample task-related configurations from latent spaces. While these methods generate individual or sets of joint configurations, recent works propose using diffusion models to learn trajectory sampling distributions to accelerate motion planning.

Diffuser [95] first explored diffusion models for generating trajectories across various start and goal configurations in a fixed environment. Subsequent work aims to generalize to unseen environments using test-time guidance [30, 182], but struggles with large environmental variations due to significant mismatches in learned trajectory distributions. Recent studies [11] address this issue by conditioning diffusion models on environmental conditions. Notably, Huang et al. [87] and Xian et al. [232] condition on point-cloud representations to sample motion plans across diverse environments. However, these models still face challenges in generalizing to a new C-space because they rely on workspace inputs, even though motion planning occurs in the C-space.

Prior work explores alternative representations for motion planning, focusing on C-space grounded approaches [97, 112]. These methods approximate complex C-spaces from past problems to find collision-free trajectories in unseen environments. In particular, our work incorporates a key-configuration representation inspired by Kim et al. [112], which utilizes collision states at task-related key configurations to enhance model generalization across varying C-spaces.

7.3 Method

PRESTO comprises three key components, illustrated in Figure 7.1. First, we generate a set of key configurations and their collision states from the motion planning dataset. Based on the resulting representation, we train a conditional diffusion model that incorporates motion-planning costs to guide the model toward smooth and collision-free trajectories. Finally, we feed the trajectories generated by the diffusion model to trajectory optimization. Each component of our framework is detailed in the following sections.

7.3.1 Problem Formulation

Let $\mathcal{C}$ be a d -dimensional C-space, which is divided into two subspaces: $\mathcal{C}_o$ representing C-space obstacles, and $\mathcal{C}_f = \mathcal{C} \setminus \mathcal{C}_o$ representing the collision-free C-space. We denote the robot’s configuration as a d -dimensional vector $q \in \mathcal{C}$. A trajectory is represented as a sequence of waypoint configurations $\tau = (q_0, q_1, \dots, q_T)$. Given the start configuration q_s and goal configuration q_g , the objective of motion planning is to find a collision-free path $\tau \in \mathcal{C}_f$ from q_s to q_g .

In this work, we assume that we are provided with a dataset $\mathcal{D} = \{\mathcal{D}_m\}_{m=1}^M$ obtained from solving M past planning problems, where each data point $\mathcal{D}_m = \{q_s, q_g, \tau, \mathcal{G}\}$ consists of a start configuration q_s , a goal configuration q_g , a trajectory τ , and an environment geometry $\mathcal{G}$. We assume consistent environment fixtures but varying object shapes and locations across problems. We use an optimization-based planner to compute ground-truth trajectories for training data. Our goal is to develop a generative model that provides a good initial solution for trajectory optimization, even in environments with unseen obstacles and their arrangements.

7.3.2 Background

Denoising Diffusion Probabilistic Models The diffusion model involves a forward diffusion process, which starts from a perfect solution (in our case, a collision-free trajectory) and gradually injects noise until reaching an isotropic Gaussian distribution, and a reverse diffusion process, which learns to generate data through gradual denoising. Let $i \in \{0, 1, \dots, N\}$ denote a denoising step. The data generation process using a diffusion model involves iteratively denoising noise τ_i until reaching a denoised sample τ_0 , where τ is a sequence of robot configurations in this work. The reverse diffusion process is given by:

$$p_\theta(\tau_{0:T}) = p(\tau_T) \prod_{i=1}^N p_\theta(\tau_{i-1} | \tau_i),$$

where $p_\theta(\tau_{i-1}|\tau_i) = \mathcal{N}(\tau_{i-1}; \mu_\theta(\tau_i, i), \Sigma_\theta(\tau_i, i))$, as proven by Ho et al. [82] under some mild conditions. Ho et al. [82] show that $\mu_\theta(\tau_i, i)$ can be modeled as a function $\epsilon_\theta(\tau_i, i)$, which predicts a noise value from a sample at the i -th denoising step. In addition, they show that the training objective can be simplified $\min_\theta \|\epsilon_\theta(\tau_i, i) - \epsilon_i\|^2$, and the diffusion model can be trained by minimizing this term.

Conditional Diffusion Models We convert the unconditional form of $p_\theta(\tau_{i-1}|\tau_i)$ to be conditioned on our environment representation. We adopt the methods proposed by Dhariwal and Nichol [51], whose objective is to generate an image conditioned on the class. Specifically, they demonstrate that the term of the conditional reverse diffusion process can be factorized as follows: $p_{\theta, \phi}(\tau_{i-1}|\tau_i, \phi) \propto p_\theta(\tau_{i-1}|\tau_i)p_\phi(\phi|\tau_{i-1})$, where ϕ is the conditioning variable, which in our case will be an environment representation. Under some mild conditions, they show that after applying the logarithm, we get $\log(p_\theta(\tau_{i-1}|\tau_i) p_\phi(\phi|\tau_{i-1})) \approx \log p(u) + \text{const}$, where $u \sim \mathcal{N}(\tau_{i-1}; \mu_\theta(\tau_i, i) + g\Sigma_\theta(\tau_i, i), \Sigma_\theta(\tau_i, i))$ and $g = \nabla_{\tau_{i-1}} \log p_\phi(\phi|\tau_{i-1})|_{\tau_{i-1}=\mu_\theta(\tau_i, i)}$. Notice that the mean parameter of the Gaussian term in u includes an additional component, $g\Sigma_\theta(\tau_i, i)$, which is not a part of the mean parameter of $p_\theta(\tau_{i-1}|\tau_i)$. This mathematical derivation implies that the diffusion model can generate a sample conditioned on ϕ by the gradients g . Based on this, we particularly use a classifier-free guidance diffusion model to implement our motion generation. It is known that conditional diffusion steps can be run by incorporating the probabilities from both a conditional and an unconditional diffusion model [81], as $\nabla \log p_\phi(\phi|\tau) = \nabla \log p_\phi(\tau|\phi) - \nabla \log p_\phi(\tau)$. Therefore, by plugging into the conditional probabilities, conditional guidance can be trained together with the diffusion models, eliminating the need for a separate classifier.

7.3.3 Environment Representation

We represent the environment as an approximation of its C-space using a collection of key configurations selected from the dataset. We denote the set of key configurations as $\{\bar{q}^k\}_{k=1}^K$, where the number of key configurations K determines

Algorithm 7.1 Key-Configuration Selection

Require: C-space/Workspace separation distance $d_q^{\min}, d_x^{\min}$,
Collision proportion bound c , Motion plan dataset $\mathcal{D}$,
Number of key configurations K

Ensure: Key configurations $\{\bar{q}^k\}_{k=1}^K$

```

// Initialization
1:  $\{\bar{q}\} \leftarrow \emptyset$  ▷ Initialize the key configuration buffer
// Sampling and selecting key configurations
2: while  $|\{\bar{q}\}| < K$  do
3:    $\{\tau, q_s, q_g, \mathcal{G}\} \sim \mathcal{D}$  ▷ Sample a motion plan instance
4:    $q \sim \tau$  ▷ Sample a configuration
5:    $d_q = \text{MinCspaceDistance}(\{\bar{q}\} \cup \{q\})$ 
6:    $d_x = \text{MinWorkspaceDistance}(\{\bar{q}\} \cup \{q\})$ 
7:    $p_c = \frac{1}{M} \sum_{m=1}^M \text{EnvCollision}(q, n)$ 
8:   if  $d_q > d_q^{\min}$  and  $d_x > d_x^{\min}$  and  $p_c \in (c, 1 - c)$  then
9:      $\{\bar{q}\} \leftarrow \{\bar{q}\} \cup \{q\}$ 
10:  end if
11: end while
12: return  $\{\bar{q}\}$ 

```

the resolution of the environment’s C-space approximation. A larger K increases the resolution of the C-space approximation, but it also raises the computational overhead at query time. For each motion planning problem, we compute the environment representation $\phi \in \{0, 1\}^K$ as a binary vector that specifies the collision states of each key configuration. In our setups, we use $K = 1025$.

Algorithm 7.1 describes our procedure for generating key configurations, which is a modified version of the algorithm originally proposed by Kim et al. [111]. It takes the dataset $\mathcal{D}$ and the hyperparameters $d_q^{\min}, d_x^{\min}, c, K$. Here, $d_q^{\min}$ denotes the minimum C-space distance between key configurations, $d_x^{\min}$ represents the minimum workspace distance for end-effector tips, and c is the bound on the proportion of environments where a key configuration is occupied.

We initialize the key configuration set as an empty set (line 1) and sample new configurations until the target size is reached (lines 2-11). At each step, a configuration is uniformly sampled from $\mathcal{D}$ (lines 3-4) and filtered based on three conditions (lines 5-10). To avoid duplicates, we ensure the new configuration is sufficiently distant from existing ones in both C-space and workspace (lines 5-6), while also limiting the proportion of collision states across different environments to prioritize informative configurations (line 7). By limiting the proportion of collision states, we filter

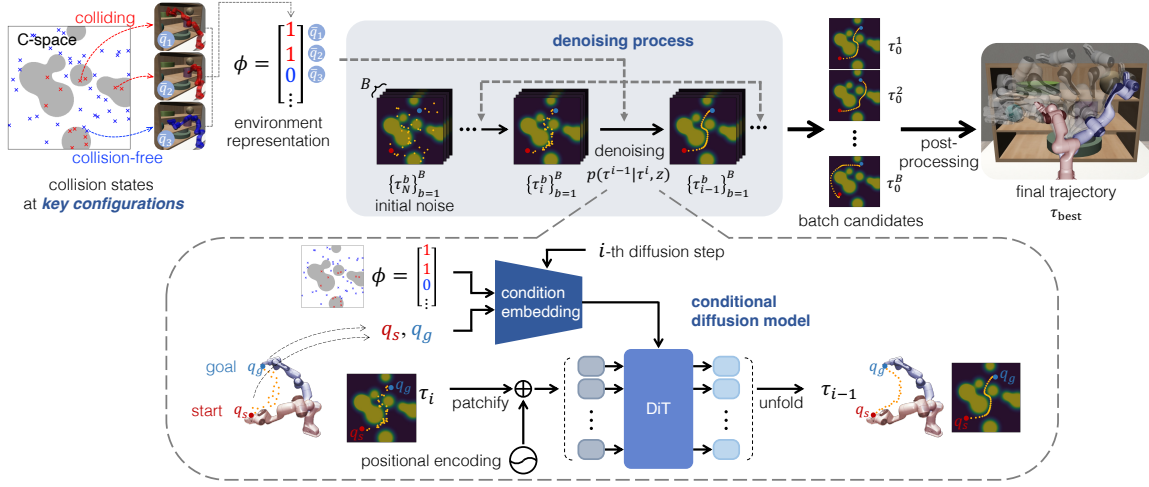


Figure 7.2: Trajectory generation pipeline of PRESTO. We obtain the environment representation for an unseen problem by checking the collision states at the key configurations used during training. Using the trained conditional diffusion model, we generate multiple trajectories conditioned on this representation and then select the least-colliding trajectory after post-processing.

out configurations that never result in collisions, as they provide no meaningful information about the environment. If all criteria are met, the configuration is added to the buffer (lines 8-10). This process generates key configurations that effectively capture task-relevant C-space regions for motion planning.

7.3.4 Training Conditional Diffusion Models

Model Architecture Figure 7.2 (bottom) illustrates our model architecture, which is based on the Diffusion Transformer (DiT) [168]. Our model takes as inputs the current diffusion step i , the noisy trajectory at the i -th step τ_i , the environment representation ϕ , and the start and goal joint configurations q_s and q_g . We use v-prediction [184] during inference to enhance sample quality [183].

To process the inputs, the trajectory τ_i is first patchified and tokenized by an MLP, as in DiT [168]. The diffusion step i and the start and goal configurations q_s and q_g are mapped to high-dimensional frequency embeddings [168] to capture small changes. The embedded trajectory patches are given as input tokens to the transformer, while i , ϕ , q_s , and q_g are incorporated as conditioning inputs to align

sampled trajectories with the current scene and endpoint constraints.

The DiT comprises six transformer blocks to process the trajectories, where each block incorporates Adaptive Layer Normalization (AdaLN) [168] that transforms the output features based on the conditioning inputs. In AdaLN, a separate MLP maps conditioning inputs to the transform parameters as $\gamma, g, b = \text{MLP}(i, \phi, q_s, q_g)$, applied to the output x as $\hat{x} = x + \gamma \odot ((1 + g) \odot \text{LN}(x) + b)$, where $\text{LN}(x)$ denotes layer normalization and $\odot$ denotes element-wise multiplication. This allows the model to adjust its output based on the current scene and the diffusion iteration.

Training with Motion-Planning Costs Our training objective includes three terms: *Diffusion Loss*, *Collision Loss*, and *Smoothing Loss*. While *Diffusion Loss* implements the standard reconstruction objective used in diffusion models, we add *Collision Loss* and *Smoothing Loss* to encourage the model to learn the motion planning constraints.

- ***Diffusion Loss***: This term $\mathcal{L}_{\text{diffusion}}$ represents the standard loss function used for training conditional diffusion models based on the DDPM framework.
- ***Collision Loss***: Inspired by TrajOpt [187], this term encourages the model to generate trajectories that maintain a safe distance from objects and other links. It is defined as

$$\mathcal{L}_{\text{coll}} = \sum_{i,j} |d_{\text{safe}} - \text{sd}(\mathcal{A}_i, \mathcal{O}_j)|^+ + \sum_{i \neq j} |d_{\text{safe}} - \text{sd}(\mathcal{A}_i, \mathcal{A}_j)|^+,$$

where $|\cdot|^+ = \max(\cdot, 0)$ and $\text{sd}(\cdot) = \text{dist}(\cdot) - \text{penetration}(\cdot)$ with $d_{\text{safe}} = 0.01$ m as the safety margin of the hinge loss. This loss is summed over each robot link $\mathcal{A}_i$ and object $\mathcal{O}_j$ based on the swept volume of the trajectory. The first term accounts for the collision between the i -th link $\mathcal{A}_i$ and the j -th obstacle $\mathcal{O}_j$, while the second term denotes self-collision among different links $\mathcal{A}_i$ and $\mathcal{A}_j$, where $i \neq j$.

- **Smoothing Loss:** This term penalizes the L2-norm between adjacent configurations, defined as $\mathcal{L}_{\text{smooth}} = \sum_t \|q_t - q_{t-1}\|^2$. It regulates the distances between consecutive configurations to encourage shorter and smoother trajectories for the robot.

We use a weighted sum of the loss terms to train our model: $\mathcal{L} = w_1 \mathcal{L}_{\text{diffusion}} + w_2 \mathcal{L}_{\text{coll}} + w_3 \mathcal{L}_{\text{smooth}}$, with $w_1 = 1.0$, $w_2 = 0.05$, and $w_3 = 0.005$. While the model is not highly sensitive to these values, w_2 is kept small for stable training.

7.3.5 Trajectory Generation

Figure 7.2 (top) provides an overview of our trajectory generation process. The inputs q_s, q_g, ϕ specify the motion planning problem, where q_s and q_g are the start and goal configurations, and ϕ is the environment representation from the key configurations’ collision states. During denoising, we use batch sampling to leverage GPU parallelization, enhancing the likelihood of finding collision-free trajectories among the diffusion model’s stochastic outputs. Our sampling follows the Denoising Diffusion Implicit Model [201], which accelerates the process by using fewer denoising iterations during inference than during training. We then apply trajectory optimization to post-process the sampled trajectories and select the one with the lowest collision cost.

We detail our trajectory generation in Algorithm 7.2. First, we compute the environment representation ϕ by checking the collision states of the key configurations $\bar{q}$ (line 1). Next, we initialize the denoising process with a batch of random noise τ_N sampled from an isotropic Gaussian distribution (line 2). At each of the N iterations, we predict a denoised trajectory τ_{i-1} from τ_i , conditioned on ϕ and the current step i (line 5), and then apply endpoint constraints to ensure connectivity between the start q_s and the goal q_g , as in Diffuser [95] (line 6). The resulting trajectories τ_{seed} provide initialization for post-processing (line 9). In our experiments, we denoise $B = 4$ trajectories over $N = 64$ iterations.

Algorithm 7.2 PRESTO Trajectory Generation

Require: Start/Goal configurations $\{q_s, q_g\}$, Environment $\mathcal{G}$,
Key configurations $\{\bar{q}^k\}_{k=1}^K$, Diffusion model $\mu_\theta(\cdot)$,
Noise schedule $\{\sigma_i\}_{i=1}^N$
Ensure: Output trajectory τ_{best}
 // Computing environment representation
1: $\phi = \text{CheckCollision}(\mathcal{G}, \{\bar{q}^k\}_{k=1}^K)$
 // Batched denoising with reverse process
2: $\{\tau_N^b \sim \mathcal{N}(\mathbf{0}, \mathbf{I})\}_{b=1}^B$ ▷ Sample a batch of initial trajectories
3: **parallel for** $b = 1, \dots, B$ **do**
4: **for** $i = N, \dots, 1$ **do**
5: $\tau_{i-1}^b \sim \mathcal{N}(\mu_\theta(\tau_i^b, \phi, i), \sigma_i)$
6: $q_0 \leftarrow q_s, q_T \leftarrow q_g$ ▷ Apply endpoint constraints q_s, q_g
7: **end for**
8: **end parallel**
9: $\tau_{\text{seed}} = \{\tau_0^b\}_{b=1}^B$ ▷ Batch-sized sampled trajectories
 // Post-processing
10: $\tau_{\text{opt}} = \text{TrajectoryOptimization}(\tau_{\text{seed}})$
11: $\tau_{\text{best}} = \text{BestTrajectory}(\tau_{\text{opt}})$
12: **return** τ_{best}

In the post-processing phase, we refine the sampled trajectories using a fixed number of trajectory optimization iterations [187] to address potential collisions (line 10). To accelerate this process, we employ cuRobo [208], which can batch-process the trajectories while eliminating data exchange across devices. Afterward, we select the trajectory with the fewest collisions (line 11).

7.3.6 Incorporating Guidance During Sampling

In an unconditional diffusion model, test-time guidance [95, 250] constrains trajectories to specific environments and start/goal configurations. While prior works [30, 87] rely on test-time guidance for collision avoidance and endpoint constraints, we use only conditional diffusion models and trajectory optimization for strict constraint satisfaction. Here, we also include an ablation study on the complementary use of guidance steps during sampling to enhance motion planning performance.

Algorithm 7.3 describes our procedure for computing cost gradients. As in MPD [30], the guidance function includes collision and smoothness costs as described in Section 7.3.4. Collision costs are computed using cuRobo [208], while smoothness costs employ a Gaussian Process prior [17, 30]. To ensure stability during guidance,

Algorithm 7.3 Guidance Step with Cost Gradients

Require: Trajectory τ , Smoothing Kernel $\mathcal{K}$, Environment $\mathcal{G}$

Ensure: Output trajectory τ_{guide}

```
1:  $\tilde{\tau} = \text{Convolve}(\text{UnNormalize}(\tau), \mathcal{K})$   
   // Computing costs  
2:  $c_{\text{coll}} = \min(\text{CollisionCost}(\tilde{\tau}, \mathcal{G}), d_{\text{max}})$   
3:  $c_{\text{smooth}} = \text{GPCost}(\tau)$   
4:  $c_{\text{guide}} = k_{\text{smooth}}c_{\text{smooth}} + k_{\text{coll}}c_{\text{coll}}$   
   // Applying guidance gradients  
5:  $g_{\text{guide}} = \text{Clamp}(\nabla_{\tau}(c_{\text{guide}}), g_{\text{max}})$   
6:  $\tau_{\text{guide}} = \tau - g_{\text{guide}}$   
7: return  $\tau_{\text{guide}}$ 
```

we smooth the sampled trajectory with a Gaussian kernel ($\sigma = 4.0$) before computing costs, allowing collision-cost gradients to affect neighboring points (line 1). We then compute the clamped collision cost ($d_{\text{max}} = 0.1$) and the smoothness cost, summing them as $k_{\text{smooth}}c_{\text{smooth}} + k_{\text{coll}}c_{\text{coll}}$, with $k_{\text{smooth}} = 1e - 9$ and $k_{\text{coll}} = 1e - 2$ (line 2-4). Gradients are computed with PyTorch [166], clamped ($g_{\text{max}} = 1.0$) to prevent erratic updates, zeroed at endpoints, and directly added to the trajectory (line 5-6).

7.4 Experiments

We design our experiments to address the following research questions: 1) Do diffusion models using key-configuration representations generalize effectively to unseen environments? 2) Does incorporating motion-planning costs into diffusion model training improve the model’s ability to capture task constraints such as collision avoidance? 3) By seeding trajectories, does our method outperform pure planners in computational efficiency and learning-based approaches in trajectory quality?

7.4.1 Experimental Setup

We evaluate our method on a motion planning task in which the Franka Emika *Panda* robot arm [73] traverses a 3-tier shelf with various objects in simulation, as presented in Figure 7.3 (top).

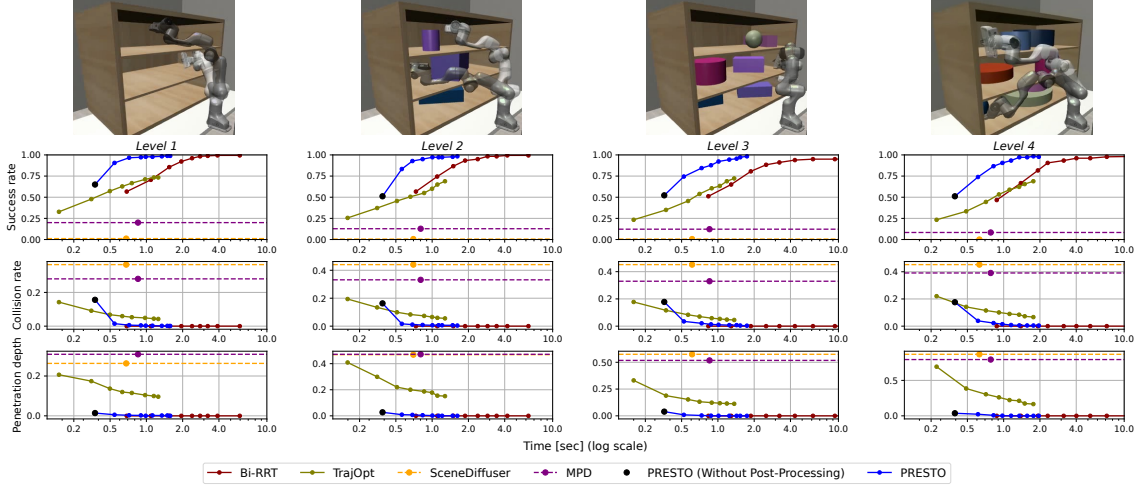


Figure 7.3: Main results. We report the success rate (%), the collision rate (%), and the penetration depth (m) across 180 problems. **(Top)** The evaluation environments feature consistent 3-tier shelf fixtures with randomized object positions that vary across levels. **(Bottom)** We show PRESTO’s performance changes across domains and computational budgets compared to the baselines.

Training Setup Our training domain consists of 5,000 environments, where 1-6 objects (cuboid, cylinder, sphere) are placed in random poses within each shelf slot. For each scene, we first sample collision-free initial and target joint positions within the workspace and then generate motion plans via cuRobo [208]. We collect a total of 50,000 environment-trajectory pairs (trajectory length $T = 1000$) annotated with key-configuration labels as in Algorithm 7.1.

Evaluation Setup While the evaluation domain is generated using a similar procedure, we partition the dataset into four difficulty levels, each consisting of 180 problems. This helps assess the performance of PRESTO and the baselines on unseen scenes of varying complexity.

- **Level 1:** The shelf is empty, as shown in Figure 7.3 (top left). Although the environment remains consistent, the task is challenging due to the shelf’s non-convex workspace and the need to connect random start and goal configurations.

- **Level 2-3:** Each slot contains one object for *Level 2* and two objects for *Level 3*, adding complexity to the C-space and requiring environment-conditional collision-free trajectory generation, as shown in Figure 7.3 (top center).
- **Level 4:** Each slot contains 3-4 objects, as shown in Figure 7.3 (top right). These environments are the most challenging due to narrow passages between obstacles, increased C-space complexity, and slower collision-checking and distance calculations among many objects.

We use the following metrics to evaluate the trajectories generated by PRESTO and other motion planning methods:

- **Success rate:** the percentage of collision-free trajectories within the batch. Higher is better.
- **Collision rate:** the average fraction of colliding segments in each trajectory. This metric reflects the likelihood that each joint configuration is collision-free, even if collisions occur elsewhere in the trajectory. Lower is better.
- **Penetration depth:** the average maximum penetration depth in each trajectory. This metric quantifies the severity of collision, measuring worst-case deviation from a collision-free trajectory. Lower is better.

To systematically evaluate how performance changes across different computational budgets, we vary the number of optimization iterations during post-processing. The Bi-RRT baseline was evaluated on an AMD Ryzen 9 5900X and all other baselines were evaluated on an NVIDIA A5000.

7.4.2 Quantitative Evaluation

To answer the research questions, we compare our model’s performance against the following baselines, representative of each approach, as presented in Figure 7.3 (bottom).

- **Bi-RRT:** a pure planner based on LaValle et al. [125], in particular, RRT-connect [119]. Bi-RRT searches bidirectionally by growing random trees from both start and goal configurations to find collision-free paths in an unknown C-space. Although it is probabilistically complete, it empirically struggles with slow searches in narrow passages. Since its success rate depends on the provided time, we evaluate its performance across different search timeouts.
- **TrajOpt:** a pure planner from Schulman et al. [187] that optimizes trajectories using a hinge penalty for collisions and configuration distances. The optimization scheme in TrajOpt is the same as in PRESTO, without the initial seed from our diffusion model. We use a GPU-accelerated implementation from cuRobo [208] and control the computational budget by adjusting optimization iterations.
- **SceneDiffuser:** a baseline from Huang et al. [87] uses a conditional diffusion model for trajectory planning through sampling. Unlike PRESTO, this baseline conditions on point-cloud inputs encoded by Point Transformer [252] instead of a C-space representation. We use the author’s original network implementation, trained on our dataset.
- **Motion Planning Diffusion (MPD):** a baseline from Carvalho et al. [30] uses a diffusion model for trajectory planning through sampling. Unlike PRESTO, MPD employs unconditional diffusion models and relies on sampling guidance for trajectory constraints such as collision avoidance or connecting start and goal configurations. We adapt their network to our setup and training dataset.

Comparison to Pure Learning Algorithms We consider pure learning algorithms, SceneDiffuser and MPD, which use neither a key-configuration environment representation nor a motion-planning objective for training diffusion models. We evaluate all methods *without* trajectory optimization post-processing, as indicated by

the large black, yellow, and purple dots in Figure 7.3 for PRESTO, SceneDiffuser, and MPD. PRESTO consistently outperforms both across all levels. For example, in *Level 3*, PRESTO achieves a 52.2% success rate, while SceneDiffuser and MPD achieve only 0.6% and 12.2%, respectively.

Comparison to Pure Planners Compared to Bi-RRT, PRESTO uses diffusion-learned trajectory priors to generate collision-free trajectories more efficiently, especially in narrow passages. In *Level 4*, PRESTO achieves a 90% success rate in 1.0 second, compared to 2.3 seconds for Bi-RRT. Furthermore, as environment complexity increases, Bi-RRT struggles to find valid segments, widening the success gap from 97.8% vs. 70.6% in *Level 1* to 90.6% vs. 46.7% in *Level 4* under a 1-second computational budget. Next, we consider TrajOpt, an optimization-based method. Despite PRESTO’s computational overhead for running the diffusion model, its high-quality initial trajectories lead to faster convergence in complex domains. For example, in *Level 2*, PRESTO achieves a 97.2% success rate in 1.0 second despite an initial overhead of 0.2 seconds, while TrajOpt remains below 60% in the same time. The success rate gap with a 1-second computational budget grows from 26.7% in *Level 1* to 37.3% in *Level 4*, demonstrating PRESTO’s effectiveness in complex environments.

7.4.3 Ablation Studies

To analyze the impact of each component, we conduct ablation studies using variants of our method, with results shown in Figure 7.4.

- ***Point-Cloud Conditioning:*** To assess the effectiveness of key-configuration representations for generalization, we train a variant of PRESTO conditioned on an equivalent number of workspace point clouds instead of key configurations. We use a patch-based transformer [244] to encode the point clouds, keeping the rest of the architecture unchanged.

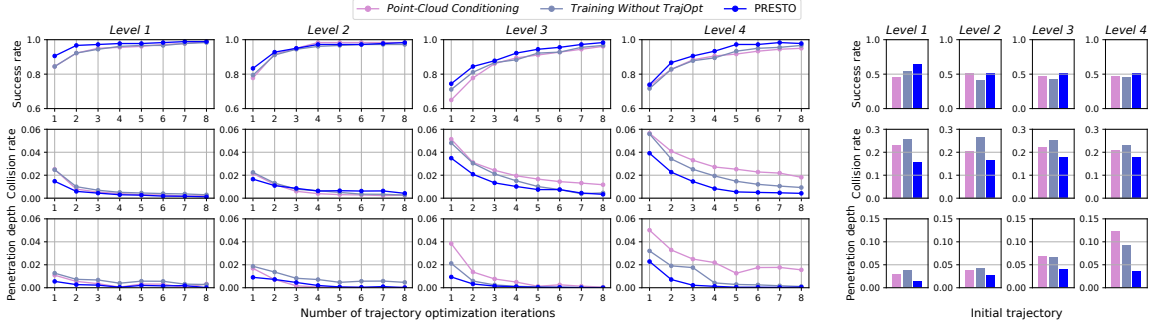


Figure 7.4: Ablation study results. We report the success rate (%), the collision rate (%), and the penetration depth (m) averaged across 180 problems for PRESTO and the self-variant baselines. **(Left)** We show performance changes with varying post-processing iterations. **(Right)** We present the performance of trajectories directly generated by the diffusion models without post-processing.

- **Training Without TrajOpt:** To examine the impact of incorporating motion-planning costs, we train a variant of PRESTO without motion-planning costs (*Collision Loss* and *Smoothing Loss*). The rest of the architecture remains unchanged.

Generalization of Key-Configuration Representation Compared to PRESTO, *Point-Cloud Conditioning* exhibits performance degradation across problem levels and post-processing iterations: collision rates and penetration depth remain higher, and worsen with increased problem complexity. For instance, in *Level 1-2*, PRESTO outperforms *Point-Cloud Conditioning* by 1.4% in success rate, 0.3% in collision rate, and 0.001 m in penetration depth. In *Level 3-4*, the gaps increase to 3.6%, 1.4%, and 0.013 m. This highlights the advantage of using C-space representations over point-cloud-based conditioning in complex scenes.

Efficacy of Training with Motion-Planning Costs Compared to PRESTO, *Training Without TrajOpt* exhibits performance degradation across all levels. Though less severe than *Point-Cloud Conditioning*, the trend is consistent. In *Level 1-2*, PRESTO outperforms *Training Without TrajOpt* by an average of 1.81% in success rate and 0.2% in collision rate. In *Level 3-4*, the performance gap increases to 2.7% in

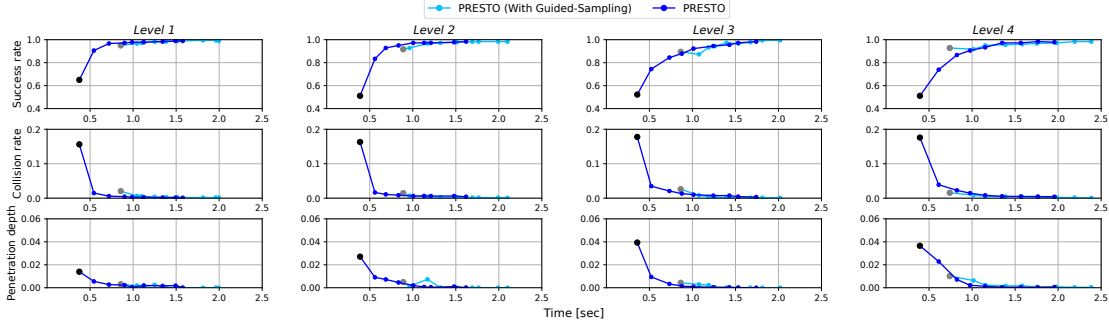


Figure 7.5: Results with guided-sampling. We report the success rate (%), the collision rate (%), and the penetration depth (m) averaged across 180 problems for PRESTO and the self-variant baselines. Black dots represent PRESTO without post-processing, while gray dots represent PRESTO with Guided Sampling, also without post-processing.

success rate and 0.7% in collision rate. This shows that incorporating motion-planning costs for training diffusion models improves trajectory quality across various domains.

Efficacy of Post-Processing We observe that applying trajectory optimization during post-processing improves performance across all levels. Additionally, we validate that the success of PRESTO is largely due to the high-quality, nearly collision-free initial trajectories obtained from our diffusion model. As shown in Figure 7.4 (right), PRESTO in *Level 4* outperforms *Point-Cloud Conditioning* by achieving a much smaller penetration depth (0.037 m vs. 0.123 m), despite similar initial success rates (51.1% vs. 47.2%), leading to faster convergence during trajectory optimization.

7.4.4 Impact of Guided-Sampling

Figure 7.5 compares our model with a variant that includes guidance steps during denoising iterations. Overall, guided-sampling enhances the quality of initial trajectories (black dots represent PRESTO without guidance, and gray dots represent PRESTO with guidance). For example, the success rate of the denoised trajectory before optimization reaches 92.8% in *Level 4*, compared to 51.1% for PRESTO without guidance. However, incorporating guidance requires evaluating cost gradients at each diffusion iteration, resulting in computational overhead. In *Level 3*, this over-

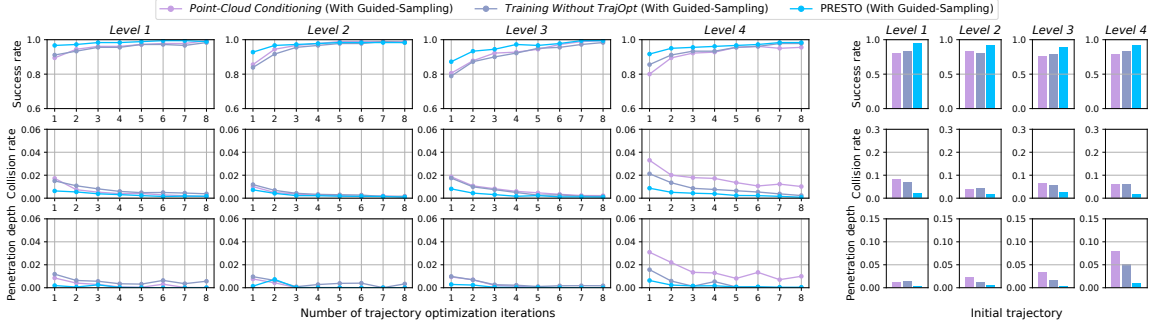


Figure 7.6: Ablation studies with guided-sampling. We report the success rate (%), the collision rate (%), and the penetration depth (m) averaged across 180 problems for PRESTO and the self-variant baselines with guided-sampling. **(Left)** We show performance changes with varying post-processing iterations. **(Right)** We present the performance of trajectories directly generated by the diffusion model without post-processing.

head accumulates to an average of 0.38 seconds, indicating that the additional cost of guidance steps may occasionally degrade performance within a given time frame. Despite this, guidance generally improves performance across *Level 1-4* in all three metrics: success rate, collision rate, and penetration depth, given the same number of trajectory optimization iterations.

We also report the effects of guidance on variants of our model in Figure 7.6. Consistent with the results in Figure 7.5, performance across all baselines improves when guidance steps are applied compared to the original results in Figure 7.4. Notably, the success rate gap between PRESTO and its variants widens with the application of guidance. For instance, across *Level 1-4*, the gap between PRESTO and the closest baseline (*Point-Cloud Conditioning*) increases from 2.4% to 4.0%. As PRESTO generates higher-quality initial trajectories with smaller penetration depths, spurious collisions are resolved with just a few guidance steps, leading to greater success rate gains compared to the ablations of PRESTO.

7.4.5 Computation Time Details

We provide computation time details for PRESTO and present them in Figure 7.7. The key-configuration query incurs an average overhead of 0.658 milliseconds,

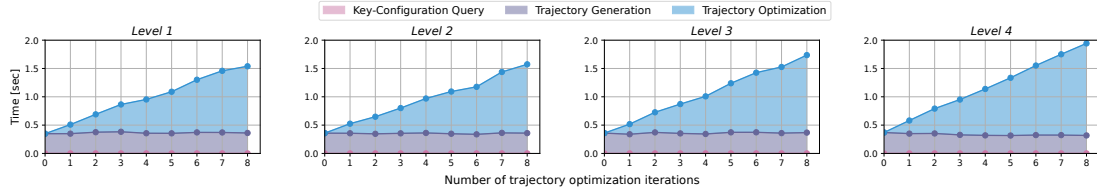


Figure 7.7: Breakdown of computation time for PRESTO. We report the average computation time (sec) across 180 problems for PRESTO and provide a breakdown of the time taken for each step: the key-configuration query for environment representation, trajectory generation using the diffusion model, and trajectory optimization during post-processing.

which is negligible compared to trajectory generation in the diffusion model, averaging 0.345 seconds, and trajectory optimization during post-processing, which scales linearly with the number of iterations. The performance improvements of PRESTO, leveraging an environment representation based on key configurations, over the *Point-Cloud Conditioning* baseline in Section 7.4.3 demonstrate its effectiveness in enhancing representation quality without compromising computational speed. This makes it particularly well-suited for scenarios with limited computational resources.

7.5 Conclusion

We present PRESTO, a learning-guided motion planning framework that integrates diffusion-based trajectory sampling with post-processing trajectory optimization. By incorporating environment representations for embodiment-specific C-spaces based on key configurations, together with a motion-planning training objective, the framework efficiently generates collision-free trajectories in previously unseen environments. Simulated experiments demonstrate the efficacy of our framework compared to both diffusion-based planning approaches and conventional motion planning methods. In this work, we assumed known environment geometry for ground-truth collision states at key configurations. Future work could extend the framework to handle partial observability of unseen geometries.

Chapter 8: Conclusion

8.1 Summary

This dissertation presents research on developing skills that generalize across diverse robotic systems by leveraging structural priors inherent in robot hardware, control, and representations. Rather than relying solely on data-driven learning, the results show that incorporating structured knowledge from robotic systems enables more efficient and reliable skill acquisition. This approach enables embodiment-aware skill learning, allowing skills to transfer across robots while maintaining physical feasibility and leading to robust and generalizable behaviors in practical scenarios. In particular, skills can be learned without requiring robots directly in the loop while still capturing meaningful tool use and object interactions, and can be transferred across robots with different morphologies. Overall, these findings highlight the importance of leveraging structural priors to build robot autonomy that is both scalable and interpretable, moving beyond purely black-box learning methods.

To recapitulate, this dissertation introduces a framework that integrates skill learning with structural priors across three components: first, hardware interfaces that enable consistent physical interactions across robots (Chapters 2 and 3); second, control abstractions that support reliable execution of complex behaviors (Chapters 4 and 5); and third, embodiment-aware representations that enable skills to adapt across different robot morphologies (Chapters 6 and 7). Together, these contributions provide a foundation for developing scalable and transferable robot skills, advancing the broader goal of building generalist robot autonomy.

8.2 Future Research

Looking ahead, achieving generalist robot autonomy requires robots to accumulate knowledge across different embodiments, transfer that knowledge to new

systems, and compose it into behaviors that generalize across tasks and platforms. A central question is how skills can be reused and composed when both task structure and robot embodiment vary. Rather than treating skill composition as purely an algorithmic problem, it should be viewed as fundamentally constrained by embodiment. The structure of the robot, its sensing and actuation capabilities, and the abstractions used to represent actions shape which behaviors can be learned, transferred, and composed. To move toward this goal, future research will study richer forms of skill composition by building on and extending the structural priors explored in this dissertation. In particular, it will investigate how control abstractions, hardware design, and representations jointly determine which skill compositions are feasible and reusable across robots. Grounded in the integration of robot-specific structural priors for developing generalizable skills, this research aims to advance the next generation of generalist robot autonomy. This agenda focuses on three complementary goals:

1. Designing hardware, control, and representations to enable broader and more versatile skills,
2. Developing principles for their co-design to support skill acquisition, transfer, and composition, and
3. Applying these insights to develop robot foundation models that enable robots to accumulate, share, and refine knowledge collectively.

8.2.1 Designing Hardware for Task Diversity

As shown in Part I, end-effector hardware can serve as a unifying interface for manipulation, enabling skill transfer across diverse robot embodiments. However, existing platforms still limit the range of transferable skills. Future research will investigate which aspects of physical interaction should be shared across robots and which must remain embodiment-specific, including grasp taxonomies and sensor placement strategies that support systematic skill reuse across platforms.

To support this goal, scalable hardware platforms will be developed to broaden the range of transferable manipulation skills. Current designs present a trade-off: dexterous hands achieve high capability but are difficult to reproduce, whereas scalable designs often lack the dexterity and sensing required for complex tasks. To bridge this gap, universal end-effectors, such as multi-finger hands [105] with integrated tactile sensing [196], will be developed to combine dexterity with reproducibility. In addition, the roles of grasp taxonomies and sensor placement in manipulation learning will be systematically studied. These platforms aim to extend sensorimotor capabilities and provide accessible, high-capacity systems for dexterous manipulation research.

8.2.2 Devising Principles of Control for Skill Learning

While earlier parts of this dissertation demonstrate the benefits of modular advances in hardware, control, and representation learning, their full potential emerges when these components are designed jointly. For complex tasks, control structure plays a central role in determining the behaviors a robot can express, shaping the space of feasible actions and their alignment with task requirements.

Future research will study the theoretical and empirical principles underlying control design for skill learning. In particular, it will investigate how control and learning can be co-designed so that learning focuses on task-relevant variation rather than embodiment-specific dynamics. A key direction is to understand how perception, control, and actuation interact to produce transferable sensorimotor capabilities. For example, force-domain action representations, such as impedance control in end-effector space, enable safe and adaptive manipulation, while actions derived from optimal control handle robot-specific constraints and achieve consistent performance across diverse actuation systems [128, 191, 193]. By formalizing these principles, this research aims to establish a foundation for control design that guides the co-development of hardware and algorithms within a unified framework.

8.2.3 Scaling Knowledge Sharing with Embodiment Awareness

The ultimate goal of embodiment-aware learning is to enable knowledge sharing across heterogeneous robots. Recent advances in robot foundation models [18, 20, 160] combine large vision-language models with robot-specific action layers and show promising results for scalable robot learning. However, these action layers remain tightly coupled to individual robots. As a result, transferring models across systems often requires extensive fine-tuning, limiting scalability and potentially causing retroactive forgetting.

Future research will incorporate explicit embodiment awareness into robot foundation models to establish a standardized action interface across diverse robots. By grounding action representations in embodiment-aware structure, robots can reuse shared knowledge while adapting to their own hardware constraints. This direction aims to enable robot foundation models that accumulate and generalize knowledge across embodiments, supporting collective learning across robot platforms.

8.3 Final Remarks

My long-term research goal is to develop a system-level pipeline for building general-purpose robots that can collectively accumulate, share, and refine reusable knowledge across diverse robotic systems. Such systems should evolve alongside the robots themselves, allowing knowledge to transfer and adapt as robot hardware, sensing, and capabilities continue to advance. Human intelligence has evolved through the collective accumulation of knowledge across individuals and generations. Similarly, future robot systems should be able to build upon shared experience, enabling capabilities to grow beyond what any single robot or system could learn alone. As these capabilities scale across systems and deployments, they enable reliable and adaptive robot behavior across a wide range of real-world domains and, in aggregate, drive the evolution of increasingly capable and general robot systems. In this sense, the development of robot autonomy has the potential not only to transform how robots operate

in real-world environments, but also to reshape how we approach entire industries and how we think about intelligence itself. My ultimate goal in developing general-purpose robot autonomy is to augment human physical and intellectual capabilities. Rather than replacing human effort, I envision robots as partners that support and extend human activities. By overcoming physical limitations across many industries, robotic systems can enable humans to focus more on broader intellectual and philosophical pursuits that drive human progress.

Works Cited

- [1] 1X NEO. <https://www.1x.tech/neo>.
- [2] Agility Robotics Digit. <https://agilityrobotics.com/robots>.
- [3] Boston Dynamics Spot. <https://bostondynamics.com/products/spot>.
- [4] Figure 03. <https://www.figure.ai/figure>.
- [5] GR1 - Fourier Intelligence. <https://fourierintelligence.com/gr1>.
- [6] Meta Quest 2: Immersive all-in-one vr headset. <https://www.meta.com/quest/products/quest-2>.
- [7] Intel RealSense SDK. <https://github.com/IntelRealSense/librealsense>.
- [8] Unitree B1. <https://shop.unitree.com/products/unitree-b1>, .
- [9] Unitree A1. <https://www.unitree.com/products/a1>, .
- [10] Junhyeok Ahn, Steven Jens Jorgensen, Seung Hyeon Bang, and Luis Sentis. Versatile locomotion planning and control for humanoid robots. *Frontiers in Robotics and AI*, 2021.
- [11] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision-making? In *International Conference on Learning Representations*, 2023.
- [12] Oktay Arslan and Panagiotis Tsiotras. Machine learning guided exploration for sampling-based motion planning algorithms. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015.

- [13] Tamim Asfour and Rüdiger Dillmann. Human-like motion of a humanoid robot arm based on a closed-form solution of the inverse kinematics problem. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003.
- [14] Tamim Asfour, Pedram Azad, Florian Gyarfas, and Rüdiger Dillmann. Imitation learning of dual-arm manipulation tasks in humanoid robots. *International Journal of Humanoid Robotics*, 2008.
- [15] Ko Ayusawa and Eiichi Yoshida. Motion retargeting for humanoid robots based on simultaneous morphing parameter identification and motion optimization. *IEEE Transactions on Robotics*, 2017.
- [16] Seung Hyeon Bang, Carlos Gonzalez, Junhyeok Ahn, Nicholas Paine, and Luis Sentis. Control and evaluation of a humanoid robot with rolling contact knees. *Frontiers in Robotics and AI*, 2023.
- [17] Tim D. Barfoot, Chi Hay Tong, and Simo Särkkä. Batch continuous-time trajectory estimation as exactly sparse gaussian process regression. In *Robotics: Science and Systems*, 2014.
- [18] Jose Barreiros, Andrew Beaulieu, Aditya Bhat, Rick Cory, Eric Cousineau, Hongkai Dai, Ching-Hsin Fang, Kunimatsu Hashimoto, Muhammad Zubair Irshad, Masha Itkina, Naveen Kuppaswamy, Kuan-Hui Lee, Katherine Liu, Dale McConachie, Ian McMahon, Haruki Nishimura, Calder Phillips-Grafflin, Charles Richter, Paarth Shah, Krishnan Srinivasan, Blake Wulfe, Chen Xu, Mengchao Zhang, Alex Alspach, Maya Angeles, Kushal Arora, Vitor Campagnolo Guizilini, Alejandro Castro, Dian Chen, Ting-Sheng Chu, Sam Creasey, Sean Curtis, Richard Denitto, Emma Dixon, Eric Dusel, Matthew Ferreira, Aimee Goncalves, Grant Gould, Damrong Guoy, Swati Gupta, Xuchen Han, Kyle Hatch, Brendan Hathaway, Allison Henry, Hillel Hochsztein, Phoebe Horgan, Shun Iwase, Donovan Jackson, Siddharth Karamcheti, Sedrick Keh, Joseph Masterjohn, Jean Mercat, Patrick Miller, Paul Mitiguy, Tony Nguyen, Jeremy

- Nimmer, Yuki Noguchi, Reko Ong, Aykut Onol, Owen Pfannenstiehl, Richard Poyner, Leticia Priebe Mendes Rocha, Gordon Richardson, Christopher Rodriguez, Derick Seale, Michael Sherman, Mariah Smith-Jones, David Tago, Pavel Tokmakov, Matthew Tran, Basile Van Hoorick, Igor Vasiljevic, Sergey Zakharov, Mark Zolotas, Rares Ambrus, Kerri Fetzer-Borelli, Benjamin Burchfiel, Hadas Kress-Gazit, Siyuan Feng, Stacie Ford, and Russ Tedrake. A careful examination of large behavior models for multitask dexterous manipulation. *arXiv preprint arXiv:2507.05331*, 2025.
- [19] Dominik Belter, Przemysław Łabecki, Peter Fankhauser, and Roland Siegwart. RGB-D terrain perception and dense mapping for legged robots. *International Journal of Applied Mathematics and Computer Science*, 2016.
- [20] 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. *arXiv preprint arXiv:2410.24164*, 2024.
- [21] Gerardo Bledt and Sangbae Kim. Implementing regularized predictive control for simultaneous real-time footstep and ground reaction force optimization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019.
- [22] Gerardo Bledt, Patrick M Wensing, and Sangbae Kim. Policy-regularized model predictive control to stabilize diverse quadrupedal gaits for the MIT cheetah. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.
- [23] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex 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. *arXiv preprint arXiv:2212.06817*, 2022.
- [24] Russell Buchanan, Lorenz Wellhausen, Marko Bjelonic, Tirthankar Bandyopadhyay, Navinda Kottege, and Marco Hutter. Perceptive whole body planning for multi-legged robots in confined spaces. *Journal of Field Robotics*, 2021.
- [25] JD Byerlee. The mechanics of stick-slip. *Tectonophysics*, 1970.
- [26] Sergi Caelles, Jordi Pont-Tuset, Federico Perazzi, Alberto Montes, Kevis-Kokitsi Maninis, and Luc Van Gool. The 2019 DAVIS challenge on VOS: Unsupervised multi-object segmentation. *arXiv preprint arXiv:1905.00737*, 2019.
- [27] Berk Calli, Arjun Singh, James Bruce, Aaron Walsman, Kurt Konolige, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. Yale-CMU-Berkeley dataset for robotic manipulation research. *The International Journal of Robotics Research*, 2017.
- [28] Duncan Calvert, Bhavyansh Mishra, Stephen McCrory, Sylvain Bertrand, Robert Griffin, and Jerry Pratt. A fast, autonomous, bipedal walking behavior over rapid regions. In *IEEE-RAS International Conference on Humanoid Robots*, 2022.
- [29] Jan Carius, Farbod Farshidian, and Marco Hutter. MPC-Net: A first principles guided policy search. *IEEE Robotics and Automation Letters*, 2020.

- [30] Joao Carvalho, An T Le, Mark Baierl, Dorothea Koert, and Jan Peters. Motion planning diffusion: Learning and planning of robot motions with diffusion models. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2023.
- [31] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.
- [32] Matthew Chang and Saurabh Gupta. One-shot visual imitation via attributed waypoints and demonstration augmentation. In *IEEE International Conference on Robotics and Automation*, 2023.
- [33] Wei Chen, Heba Khamis, Ingvars Birznieks, Nathan F Lepora, and Stephen J Redmond. Tactile sensors for friction estimation and incipient slip detection—toward dexterous robotic manipulation: A review. *IEEE Sensors Journal*, 2018.
- [34] Ho Kei Cheng, Seoung Wug Oh, Brian Price, Joon-Young Lee, and Alexander Schwing. Putting the object back into video object segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.
- [35] Xuxin Cheng, Yandong Ji, Junming Chen, Ruihan Yang, Ge Yang, and Xiaolong Wang. Expressive whole-body control for humanoid robots. In *Robotics: Science and Systems*, 2024.
- [36] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion Policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems*, 2023.

- [37] Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal Manipulation Interface: In-the-wild robot teaching without in-the-wild robots. In *Robotics: Science and Systems*, 2024.
- [38] Hojung Choi, Mark R Cutkosky, and Andrew A Stanley. Integrated pneumatic sensing and actuation for soft haptic devices. *IEEE Robotics and Automation Letters*, 2023.
- [39] Sungjoon Choi, Matthew KXJ Pan, and Joohyung Kim. Nonparametric motion retargeting for humanoid robots on shared latent space. In *Robotics: Science and Systems*, 2020.
- [40] Open X-Embodiment Collaboration, Abby O’Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, Albert Tung, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, An-chit Gupta, Andrew Wang, Andrey Kolobov, Anikait Singh, Animesh Garg, Aniruddha Kembhavi, Annie Xie, Anthony Brohan, Antonin Raffin, Archit Sharma, Arefeh Yavary, Arhan Jain, Ashwin Balakrishna, Ayzaan Wahid, Ben Burgess-Limerick, Beomjoon Kim, Bernhard Schölkopf, Blake Wulfe, Brian Ichter, Cewu Lu, Charles Xu, Charlotte Le, Chelsea Finn, Chen Wang, Chen-feng Xu, Cheng Chi, Chenguang Huang, Christine Chan, Christopher Agia, Chuer Pan, Chuyuan Fu, Coline Devin, Danfei Xu, Daniel Morton, Danny Driess, Daphne Chen, Deepak Pathak, Dhruv Shah, Dieter Büchler, Dinesh Jayaraman, Dmitry Kalashnikov, Dorsa Sadigh, Edward Johns, Ethan Foster, Fangchen Liu, Federico Ceola, Fei Xia, Feiyu Zhao, Felipe Vieira Frujeri, Freek Stulp, Gaoyue Zhou, Gaurav S. Sukhatme, Gautam Salhotra, Ge Yan, Gilbert Feng, Giulio Schiavi, Glen Berseth, Gregory Kahn, Guangwen Yang, Guanzhi Wang, Hao Su, Hao-Shu Fang, Haochen Shi, Henghui Bao, Heni Ben Amor, Henrik I Christensen, Hiroki Furuta, Homanga Bharadhwaj, Homer

Walke, Hongjie Fang, Huy Ha, Igor Mordatch, Ilija Radosavovic, Isabel Leal, Jacky Liang, Jad Abou-Chakra, Jaehyung Kim, Jaimyn Drake, Jan Peters, Jan Schneider, Jasmine Hsu, Jay Vakil, Jeannette Bohg, Jeffrey Bingham, Jeffrey Wu, Jensen Gao, Jiaheng Hu, Jiajun Wu, Jialin Wu, Jiankai Sun, Jianlan Luo, Jiayuan Gu, Jie Tan, Jihoon Oh, Jimmy Wu, Jingpei Lu, Jingyun Yang, Jitendra Malik, João Silvério, Joey Hejna, Jonathan Booyer, Jonathan Tompson, Jonathan Yang, Jordi Salvador, Joseph J. Lim, Junhyek Han, Kaiyuan Wang, Kanishka Rao, Karl Pertsch, Karol Hausman, Keegan Go, Keerthana Gopalakrishnan, Ken Goldberg, Kendra Byrne, Kenneth Oslund, Kento Kawaharazuka, Kevin Black, Kevin Lin, Kevin Zhang, Kiana Ehsani, Kiran Lekkala, Kirsty Ellis, Krishan Rana, Krishnan Srinivasan, Kuan Fang, Kunal Pratap Singh, Kuo-Hao Zeng, Kyle Hatch, Kyle Hsu, Laurent Itti, Lawrence Yunliang Chen, Lerrel Pinto, Li Fei-Fei, Liam Tan, Linxi "Jim" Fan, Lionel Ott, Lisa Lee, Luca Weihs, Magnum Chen, Marion Lepert, Marius Memmel, Masayoshi Tomizuka, Masha Itkina, Mateo Guaman Castro, Max Spero, Maximilian Du, Michael Ahn, Michael C. Yip, Mingtong Zhang, Mingyu Ding, Minh Heo, Mohan Kumar Srirama, Mohit Sharma, Moo Jin Kim, Muhammad Zubair Irshad, Naoaki Kanazawa, Nicklas Hansen, Nicolas Heess, Nikhil J Joshi, Niko Suenderhauf, Ning Liu, Norman Di Palo, Nur Muhammad Mahi Shafiullah, Oier Mees, Oliver Kroemer, Osbert Bastani, Pannag R Sanketi, Patrick "Tree" Miller, Patrick Yin, Paul Wohlhart, Peng Xu, Peter David Fagan, Peter Mitran, Pierre Sermanet, Pieter Abbeel, Priya Sundaresan, Qiuyu Chen, Quan Vuong, Rafael Rafailov, Ran Tian, Ria Doshi, Roberto Mart'in-Mart'in, Rohan Bajjal, Rosario Scalise, Rose Hendrix, Roy Lin, Runjia Qian, Ruohan Zhang, Russell Mendonca, Rutav Shah, Ryan Hoque, Ryan Julian, Samuel Bustamante, Sean Kirmani, Sergey Levine, Shan Lin, Sherry Moore, Shikhar Bahl, Shivin Dass, Shubham Sonawani, Shubham Tulsiani, Shuran Song, Sichun Xu, Siddhant Halder, Siddharth Karamcheti, Simeon Adebola, Simon Guist, Soroush Nasiriany, Stefan Schaal, Stefan Welker, Stephen Tian, Subramanian

- Ramamoorthy, Sudeep Dasari, Suneel Belkhale, Sungjae Park, Suraj Nair, Suvir Mirchandani, Takayuki Osa, Tanmay Gupta, Tatsuya Harada, Tatsuya Matsushima, Ted Xiao, Thomas Kollar, Tianhe Yu, Tianli Ding, Todor Davchev, Tony Z. Zhao, Travis Armstrong, Trevor Darrell, Trinity Chung, Vidhi Jain, Vikash Kumar, Vincent Vanhoucke, Vitor Guizilini, Wei Zhan, Wenxuan Zhou, Wolfram Burgard, Xi Chen, Xiangyu Chen, Xiaolong Wang, Xinghao Zhu, Xinyang Geng, Xiyuan Liu, Xu Liangwei, Xuanlin Li, Yansong Pang, Yao Lu, Yecheng Jason Ma, Yejin Kim, Yevgen Chebotar, Yifan Zhou, Yifeng Zhu, Yilin Wu, Ying Xu, Yixuan Wang, Yonatan Bisk, Yongqiang Dou, Yoonyoung Cho, Youngwoon Lee, Yuchen Cui, Yue Cao, Yueh-Hua Wu, Yujin Tang, Yuke Zhu, Yunchu Zhang, Yunfan Jiang, Yunshuang Li, Yunzhu Li, Yusuke Iwasawa, Yutaka Matsuo, Zehan Ma, Zhuo Xu, Zichen Jeff Cui, Zichen Zhang, Zipeng Fu, and Zipeng Lin. Open X-Embodiment: Robotic learning datasets and RT-X models. In *IEEE International Conference on Robotics and Automation*, 2024.
- [41] Erwin Coumans and Yunfei Bai. PyBullet, a python module for physics simulation in robotics, games and machine learning, 2017.
- [42] Whitney Crooks, Gabrielle Vukasin, Maeve O’Sullivan, William Messner, and Chris Rogers. Fin Ray[®] effect inspired soft robotic gripper: From the robosoft grand challenge toward optimization. *Frontiers in Robotics and AI*, 2016.
- [43] Xingye Da, Zhaoming Xie, David Hoeller, Byron Boots, Anima Anandkumar, Yuke Zhu, Buck Babich, and Animesh Garg. Learning a contact-adaptive controller for robust, efficient legged locomotion. In *Conference on Robot Learning*, 2021.
- [44] Hongkai Dai, Andrés Valenzuela, and Russ Tedrake. Whole-body motion planning with centroidal dynamics and full kinematics. In *IEEE-RAS International Conference on Humanoid Robots*, 2014.

- [45] Michael Danielczuk, Arsalan Mousavian, Clemens Eppner, and Dieter Fox. Object rearrangement using learned implicit collision functions. In *IEEE International Conference on Robotics and Automation*, 2021.
- [46] Kouros Darvish, Yeshasvi Tirupachuri, Giulio Romualdi, Lorenzo Rapetti, Diego Ferigo, Francisco Javier Andrade Chavez, and Daniele Pucci. Whole-body geometric retargeting for humanoid robots. In *IEEE-RAS International Conference on Humanoid Robots*, 2019.
- [47] Kouros Darvish, Luigi Penco, Joao Ramos, Rafael Cisneros, Jerry Pratt, Eiichi Yoshida, Serena Ivaldi, and Daniele Pucci. Teleoperation of humanoid robots: A survey. *IEEE Transactions on Robotics*, 2023.
- [48] Nikhil Das and Michael Yip. Learning-based proxy collision detection for robot motion planning applications. *IEEE Transactions on Robotics*, 2020.
- [49] Cosimo Della Santina, Giorgio Grioli, Manuel Catalano, Alberto Brando, and Antonio Bicchi. Dexterity augmentation on a synergistic hand: The Pisa/IIT SoftHand+. In *IEEE-RAS International Conference on Humanoid Robots*, 2015.
- [50] Emel Demircan, Thor Besier, Samir Menon, and Oussama Khatib. Human motion reconstruction and synthesis of human skills. In *Advances in Robot Kinematics: Motion in Man and Machine: Motion in Man and Machine*, 2010.
- [51] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In *Advances in Neural Information Processing Systems*, 2021.
- [52] Jared Di Carlo, Patrick M Wensing, Benjamin Katz, Gerardo Bleddt, and Sangbae Kim. Dynamic locomotion in the MIT cheetah 3 through convex model-predictive control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018.

- [53] Norman Di Palo and Edward Johns. Learning multi-stage tasks with one demonstration via self-replay. In *Conference on Robot Learning*, 2022.
- [54] Wensong Diao, Xiaoli Wang, Wei Shi, Ying Cao, and Genshuo Liu. A tribo-piezoelectric coupled sensor for force and slip detection in soft grippers. *Nano Energy*, 2025.
- [55] Siyuan Dong, Daolin Ma, Elliott Donlon, and Alberto Rodriguez. Maintaining grasps within slipping bound by monitoring incipient slip. In *IEEE International Conference on Robotics and Automation*, 2019.
- [56] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas McHugh B, and Vincent Vanhoucke. Google Scanned Objects: A high-quality dataset of 3D scanned household items. In *IEEE International Conference on Robotics and Automation*, 2022.
- [57] Hongjie Fang, Hao-Shu Fang, Yiming Wang, Jieji Ren, Jingjing Chen, Ruo Zhang, Weiming Wang, and Cewu Lu. AirExo: Low-cost exoskeletons for learning whole-arm manipulation in the wild. In *IEEE International Conference on Robotics and Automation*, 2024.
- [58] Raul Fernandez, Ismael Payo, Andres S Vazquez, and Jonathan Becedas. Micro-vibration-based slip detection in tactile force sensors. *Sensors*, 2014.
- [59] Chelsea Finn, Xin Yu Tan, Yan Duan, Trevor Darrell, Sergey Levine, and Pieter Abbeel. Deep spatial autoencoders for visuomotor learning. In *IEEE International Conference on Robotics and Automation*, 2016.
- [60] Mario Daniele Fiore, Gaetano Meli, Anton Ziese, Bruno Siciliano, and Ciro Natale. A general framework for hierarchical redundancy resolution under arbitrary constraints. *IEEE Transactions on Robotics*, 2023.
- [61] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, 1997.

- [62] Zipeng Fu, Ashish Kumar, Jitendra Malik, and Deepak Pathak. Minimizing energy consumption leads to the emergence of gaits in legged robots. In *Conference on Robot Learning*, 2021.
- [63] Zipeng Fu, Qingqing Zhao, Qi Wu, Gordon Wetzstein, and Chelsea Finn. HumanPlus: Humanoid shadowing and imitation from humans. In *Conference on Robot Learning*, 2024.
- [64] Niklas Funk, Erik Helmut, Georgia Chalvatzaki, Roberto Calandra, and Jan Peters. Evetac: An event-based optical tactile sensor for robotic manipulation. *IEEE Transactions on Robotics*, 2024.
- [65] Siddhant Gangapurwala, Mathieu Geisert, Romeo Orsolino, Maurice Fallon, and Ioannis Havoutis. RLOC: Terrain-aware legged locomotion using reinforcement learning and optimal control. *IEEE Transactions on Robotics*, 2022.
- [66] Giorgio Giannone, Akash Srivastava, Ole Winther, and Faez Ahmed. Aligning optimization trajectories with diffusion models for constrained design generation. In *Advances in Neural Information Processing Systems*, 2023.
- [67] Michael Gleicher. Retargetting motion to new characters. In *ACM SIGGRAPH*, 1998.
- [68] Stacey L Gorniak, Vladimir M Zatsiorsky, and Mark L Latash. Manipulation of a fragile object. *Experimental Brain Research*, 2010.
- [69] Abhinav Grover, Philippe Nadeau, Christopher Grebe, and Jonathan Kelly. Learning to detect slip with barometric tactile sensors and a temporal convolutional neural network. In *IEEE International Conference on Robotics and Automation*, 2022.
- [70] Alexander M Gruebele, Michael A Lin, Dane Brouwer, Shenli Yuan, Andrew C Zerbe, and Mark R Cutkosky. A stretchable tactile sleeve for reaching into cluttered spaces. *IEEE Robotics and Automation Letters*, 2021.

- [71] Dingkun Guo. Learning multi-step manipulation tasks from a single human demonstration. *arXiv preprint arXiv:2312.15346*, 2023.
- [72] Huy Ha, Yihuai Gao, Zipeng Fu, Jie Tan, and Shuran Song. UMI on legs: Making manipulation policies mobile with manipulation-centric whole-body controllers. In *Conference on Robot Learning*, 2024.
- [73] Sami Haddadin, Sven Parusel, Lars Johannsmeier, Saskia Golz, Simon Gabl, Florian Walch, Mohamadreza Sabaghian, Christoph Jähne, Lukas Hausperger, and Simon Haddadin. The Franka Emika robot: A reference platform for robotics research and education. *IEEE Robotics & Automation Magazine*, 2022.
- [74] Sandra G Hart and Lowell E Staveland. Development of NASA-TLX (task load index): Results of empirical and theoretical research. *Advances in Psychology*, 1988.
- [75] Jiro Hashizume, Tae Myung Huh, Srinivasan A Suresh, and Mark R Cutkosky. Capacitive sensing for a gripper with gecko-inspired adhesive film. *IEEE Robotics and Automation Letters*, 2019.
- [76] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016.
- [77] Liang He, Qiuji Lu, Sara-Adela Abad, Nicolas Rojas, and Thrishantha Nanayakkara. Soft fingertips with tactile sensing and active deformation for robust grasping of delicate objects. *IEEE Robotics and Automation Letters*, 2020.
- [78] Tairan He, Zhengyi Luo, Wenli Xiao, Chong Zhang, Kris Kitani, Changliu Liu, and Guanya Shi. Learning human-to-humanoid real-time whole-body teleoperation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2024.

- [79] Nick Heppert, Max Argus, Tim Welschhold, Thomas Brox, and Abhinav Valada. DITTO: Demonstration imitation by trajectory transformation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2024.
- [80] 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 Kelly Fu, Bob Wei, Sangeetha Ramesh, Khem Holden, Kim Kleiven, David Rendleman, Sean Kirmani, Jeff Bingham, Jon 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*, 2023.
- [81] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *Advances in Neural Information Processing Systems*, 2021.
- [82] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.
- [83] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 1997.
- [84] David Hoeller, Lorenz Wellhausen, Farbod Farshidian, and Marco Hutter. Learning a state representation and navigation in cluttered and dynamic environments. *IEEE Robotics and Automation Letters*, 2021.
- [85] Robert D Howe and Mark R Cutkosky. Sensing skin acceleration for slip and texture perception. In *IEEE International Conference on Robotics and Automation*, 1989.

- [86] Kai Hu, Christian Ott, and Dongheui Lee. Online human walking imitation in task and joint space based on quadratic programming. In *IEEE International Conference on Robotics and Automation*, 2014.
- [87] Siyuan Huang, Zan Wang, Puhao Li, Baoxiong Jia, Tengyu Liu, Yixin Zhu, Wei Liang, and Song-Chun Zhu. Diffusion-based generation, optimization, and planning in 3d scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.
- [88] Yixuan Huang, Jialin Yuan, Chanhoo Kim, Pupul Pradhan, Bryan Chen, Li Fuxin, and Tucker Hermans. Out of sight, still in mind: Reasoning and planning about unobserved objects with video tracking enabled memory models. In *IEEE International Conference on Robotics and Automation*, 2024.
- [89] Jinwook Huh and Daniel D Lee. Learning high-dimensional mixture models for fast collision detection in rapidly-exploring random trees. In *IEEE International Conference on Robotics and Automation*, 2016.
- [90] Brian Ichter, James Harrison, and Marco Pavone. Learning sampling distributions for robot motion planning. In *IEEE International Conference on Robotics and Automation*, 2018.
- [91] Chieko Imai, Minghao Zhang, Yuchen Zhang, Marcin Kierebinski, Ruihan Yang, Yuzhe Qin, and Xiaolong Wang. Vision-guided quadrupedal locomotion in the wild with multi-modal delay randomization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2022.
- [92] Yasuhiro Ishiguro, Tasuku Makabe, Yuya Nagamatsu, Yuta Kojio, Kunio Kojima, Fumihito Sugai, Yohei Kakiuchi, Kei Okada, and Masayuki Inaba. Bilateral humanoid teleoperation system using whole-body exoskeleton cockpit TABLIS. *IEEE Robotics and Automation Letters*, 2020.

- [93] Thomas Fridolin Iversen and Lars-Peter Ellekilde. Kernel density estimation based self-learning sampling strategy for motion planning of repetitive tasks. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2016.
- [94] Deepali Jain, Atil Iscen, and Ken Caluwaerts. Hierarchical reinforcement learning for quadruped locomotion. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019.
- [95] Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, 2022.
- [96] Fabian Jenelten, Takahiro Miki, Aravind E Vijayan, Marko Bjelonic, and Marco Hutter. Perceptive locomotion in rough terrain – online foothold optimization. *IEEE Robotics and Automation Letters*, 2020.
- [97] Nikolay Jetchev and Marc Toussaint. Fast motion planning from experience: Trajectory prediction for speeding up movement generation. *Autonomous Robots*, 2013.
- [98] Biao Jiang, Xin Chen, Wen Liu, Jingyi Yu, Gang Yu, and Tao Chen. MotionGPT: Human motion as a foreign language. In *Advances in Neural Information Processing Systems*, 2024.
- [99] Roland S Johansson and J Randall Flanagan. Coding and use of tactile signals from the fingertips in object manipulation tasks. *Nature Reviews Neuroscience*, 2009.
- [100] Roland S. Johansson and Åke B. Vallbo. Tactile sensory coding in the glabrous skin of the human hand. *Trends in Neurosciences*, 1983.
- [101] Edward Johns. Coarse-to-fine imitation learning: Robot manipulation from a single demonstration. In *IEEE International Conference on Robotics and Automation*, 2021.

- [102] Steven Jens Jorgensen, Michael W. Lanighan, Sylvain S Bertrand, Andrew Watson, Joseph S Altemus, R. Scott Askew, Lyndon Bridgwater, Beau Domingue, Charlie Kendrick, Jason Lee, Mark Paterson, Jairo Sanchez, Patrick Beeson, Seth Gee, Stephen Hart, Ana Huaman Quispe, Robert Griffin, Inho Lee, Stephen McCrory, Luis Sentis, Jerry Pratt, and Joshua S Mehling. Deploying the nasa valkyrie humanoid for ied response: An initial approach and evaluation summary. In *IEEE-RAS International Conference on Humanoid Robots*, 2019.
- [103] Steven Jens Jorgensen, Mihir Vedantam, Ryan Gupta, Henry Cappel, and Luis Sentis. Finding locomanipulation plans quickly in the locomotion constrained manifold. In *IEEE International Conference on Robotics and Automation*, 2020.
- [104] Euan Judd, Bekir Aksoy, Krishna Manaswi Digumarti, Herbert Shea, and Dario Floreano. Slip anticipation for grasping deformable objects using a soft force sensor. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2022.
- [105] Dongho Kang, Aaron Kim, Mingyo Seo, Kazuto Yokoyama, Tetsuya Narita, and Luis Sentis. PLATO hand: Shaping contact behavior with fingernails for precise manipulation. *arXiv preprint arXiv:2602.05156*, 2026.
- [106] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker: It is better to track together. In *European Conference on Computer Vision*, 2024.
- [107] Lydia E Kavraki, Petr Svestka, J-C Latombe, and Mark H Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 1996.
- [108] Charles C Kemp, Aaron Edsinger, Henry M Clever, and Blaine Matulevich. The design of stretch: A compact, lightweight mobile manipulator for indoor

human environments. In *IEEE International Conference on Robotics and Automation*, 2022.

- [109] J Chase Kew, Brian Ichter, Maryam Bandari, Tsang-Wei Edward Lee, and Aleksandra Faust. Neural collision clearance estimator for batched motion planning. In *Workshop on the Algorithmic Foundations of Robotics*, 2020.
- [110] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, Peter David Fagan, Joey Hejna, Masha Itkina, Marion Lepert, Yecheng Jason Ma, Patrick Tree Miller, Jimmy Wu, Suneel Belkhale, Shivin Dass, Huy Ha, Arhan Jain, Abraham Lee, Youngwoon Lee, Marius Memmel, Sungjae Park, Ilija Radosavovic, Kaiyuan Wang, Albert Zhan, Kevin Black, Cheng Chi, Kyle Beltran Hatch, Shan Lin, Jingpei Lu, Jean Mercat, Abdul Rehman, Pannag R Sanketi, Archit Sharma, Cody Simpson, Quan Vuong, Homer Rich Walke, Blake Wulfe, Ted Xiao, Jonathan Heewon Yang, Arefeh Yavary, Tony Z. Zhao, Christopher Agia, Rohan Baijal, Mateo Guaman Castro, Daphne Chen, Qiuyu Chen, Trinity Chung, Jaimyn Drake, Ethan Paul Foster, Jensen Gao, Vitor Guizilini, David Antonio Herrera, Minh Heo, Kyle Hsu, Jiaheng Hu, Muhammad Zubair Irshad, Donovan Jackson, Charlotte Le, Yunshuang Li, Kevin Lin, Roy Lin, Zehan Ma, Abhiram Maddukuri, Suvir Mirchandani, Daniel Morton, Tony Nguyen, Abigail O’Neill, Rosario Scalise, Derick Seale, Victor Son, Stephen Tian, Emi Tran, Andrew E. Wang, Yilin Wu, Annie Xie, Jingyun Yang, Patrick Yin, Yunchu Zhang, Osbert Bastani, Glen Berseth, Jeannette Bohg, Ken Goldberg, Abhinav Gupta, Abhishek Gupta, Dinesh Jayaraman, Joseph J Lim, Jitendra Malik, Roberto Martín-Martín, Subramanian Ramamoorthy, Dorsa Sadigh, Shuran Song, Jiajun Wu, Michael C. Yip, Yuke Zhu, Thomas Kollar, Sergey Levine, and Chelsea Finn. DROID: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

- [111] Beomjoon Kim, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Guiding search in continuous state-action spaces by learning an action sampler from off-target search experience. In *AAAI Conference on Artificial Intelligence*, 2018.
- [112] Beomjoon Kim, Leslie Pack Kaelbling, and Tomás Lozano-Pérez. Adversarial actor-critic method for task and motion planning problems using planning experience. In *AAAI Conference on Artificial Intelligence*, 2019.
- [113] Donghyun Kim, Jared Di Carlo, Benjamin Katz, Gerardo Bledt, and Sangbae Kim. Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control. *arXiv preprint arXiv:1909.06586*, 2019.
- [114] Donghyun Kim, Daniel Carballo, Jared Di Carlo, Benjamin Katz, Gerardo Bledt, Bryan Lim, and Sangbae Kim. Vision aided dynamic exploration of unstructured terrain with a small-scale quadruped robot. In *IEEE International Conference on Robotics and Automation*, 2020.
- [115] Gyuwon Kim and Donghyun Hwang. BaroTac: Barometric three-axis tactile sensor with slip detection capability. *Sensors*, 2022.
- [116] Heecheol Kim, Yoshiyuki Ohmura, Akihiko Nagakubo, and Yasuo Kuniyoshi. Training robots without robots: Deep imitation learning for master-to-robot policy transfer. *IEEE Robotics and Automation Letters*, 2023.
- [117] Sunwoo Kim, Maks Sorokin, Jehee Lee, and Sehoon Ha. Human motion control of quadrupedal robots using deep reinforcement learning. *arXiv preprint arXiv:2204.13336*, 2022.
- [118] Twan Koolen, Sylvain Bertrand, Gray Thomas, Tomas De Boer, Tingfan Wu, Jesper Smith, Johannes Engelsberger, and Jerry Pratt. Design of a momentum-based control framework and application to the humanoid robot atlas. *International Journal of Humanoid Robotics*, 2016.

- [119] James J Kuffner and Steven M LaValle. RRT-Connect: An efficient approach to single-query path planning. In *IEEE International Conference on Robotics and Automation*, 2000.
- [120] Scott Kuindersma, Robin Deits, Maurice Fallon, Andrés Valenzuela, Hongkai Dai, Frank Permenter, Twan Koolen, Pat Marion, and Russ Tedrake. Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot. *Autonomous Robots*, 2016.
- [121] Abhishek Kumar, Yuto Tanaka, Anastasios Grigoriadis, Joannis Grigoriadis, Mats Trulsson, and Peter Svensson. Training-induced dynamics of accuracy and precision in human motor control. *Scientific Reports*, 2017.
- [122] Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. RMA: Rapid motor adaptation for legged robots. In *Conference on Robot Learning*, 2021.
- [123] Rahul Kumar, Aditya Mandalika, Sanjiban Choudhury, and Siddhartha Srinivasa. LEGO: Leveraging experience in roadmap generation for sampling-based planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019.
- [124] Yen-Ling Kuo, Andrei Barbu, and Boris Katz. Deep sequential models for sampling-based planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018.
- [125] Steven M LaValle and James J Kuffner. Rapidly-exploring random trees: Progress and prospects. In *Algorithmic and computational robotics*. A K Peters/CRC Press, 2001.
- [126] Byeongho Lee, Yonghyeon Lee, Seungyeon Kim, MinJun Son, and Frank C Park. Equivariant motion manifold primitives. In *Conference on Robot Learning*, 2023.

- [127] Dongheui Lee, Raffaele Soloperto, and Matteo Saveriano. Bidirectional invariant representation of rigid body motions and its application to gesture recognition and reproduction. *Autonomous Robots*, 2018.
- [128] Jaemin Lee, Mingyo Seo, Andrew Bylard, Robert Sun, and Luis Sentis. Real-time model predictive control for industrial manipulators with singularity-tolerant hierarchical task control. In *IEEE International Conference on Robotics and Automation*, 2023.
- [129] Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science Robotics*, 2020.
- [130] Yisoo Lee, Sanghyun Kim, Jaeheung Park, Nikos Tsagarakis, and Jinoh Lee. A whole-body control framework based on the operational space formulation under inequality constraints via task-oriented optimization. *IEEE Access*, 2021.
- [131] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 2016.
- [132] Jianhua Li, Siyuan Dong, and Edward Adelson. Slip detection with combined tactile and visual information. In *IEEE International Conference on Robotics and Automation*, 2018.
- [133] Jinhan Li, Yifeng Zhu, Yuqi Xie, Zhenyu Jiang, Mingyo Seo, Georgios Pavlakos, and Yuke Zhu. OKAMI: Teaching humanoid robots manipulation skills through single video imitation. In *Conference on Robot Learning*, 2024.
- [134] Tianyu Li, Jungdam Won, Alexander Clegg, Jeonghwan Kim, Akshara Rai, and Sehoon Ha. ACE: Adversarial correspondence embedding for cross morphology motion retargeting from human to nonhuman characters. In *ACM SIGGRAPH Asia*, 2023.

- [135] Yue Li, Manjun Zhao, Yadong Yan, Luanxi He, Yingyi Wang, Zuoping Xiong, Shuqi Wang, Yuanyuan Bai, Fuqin Sun, Qifeng Lu, Yu Wang, Tie Li, and Ting Zhang. Multifunctional biomimetic tactile system via a stick-slip sensing strategy for human-machine interactions. *NPJ Flexible electronics*, 2022.
- [136] Yuwei Liang, Weijie Li, Yue Wang, Rong Xiong, Yichao Mao, and Jiafan Zhang. Dynamic movement primitive based motion retargeting for dual-arm sign language motions. In *IEEE International Conference on Robotics and Automation*, 2021.
- [137] Toru Lin, Yu Zhang, Qiyang Li, Haozhi Qi, Brent Yi, Sergey Levine, and Jitendra Malik. Learning visuotactile skills with two multifingered hands. In *IEEE International Conference on Robotics and Automation*, 2025.
- [138] Huihan Liu, Rutav Shah, Shuijing Liu, Jack Pittenger, Mingyo Seo, Yuchen Cui, Yonatan Bisk, Roberto Martín-Martín, and Yuke Zhu. Casper: Inferring diverse intents for assistive teleoperation with vision language models. In *Conference on Robot Learning*, 2025.
- [139] Sandra Q Liu and Edward H Adelson. GelSight Fin Ray: Incorporating tactile sensing into a soft compliant robotic gripper. In *IEEE International Conference on Soft Robotics*, 2022.
- [140] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection. In *European Conference on Computer Vision*, 2024.
- [141] Zeyi Liu, Cheng Chi, Eric Cousineau, Naveen Kuppaswamy, Benjamin Burchfiel, and Shuran Song. ManiWAV: Learning robot manipulation from in-the-wild audio-visual data. In *Conference on Robot Learning*, 2025.

- [142] Zhengyi Luo, Jinkun Cao, Alexander Winkler, Kris Kitani, and Weipeng Xu. Perpetual humanoid control for real-time simulated avatars. In *IEEE/CVF International Conference on Computer Vision*, 2023.
- [143] Zhengyi Luo, Jinkun Cao, Josh Merel, Alexander Winkler, Jing Huang, Kris Kitani, and Weipeng Xu. Universal humanoid motion representations for physics-based control. In *International Conference on Learning Representations*, 2024.
- [144] Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. In *Conference on robot learning*, 2019.
- [145] Octavio Antonio Villarreal Magana, Victor Barasuol, Marco Camurri, Luca Franceschi, Michele Focchi, Massimiliano Pontil, Darwin G Caldwell, and Claudio Semini. Fast and continuous foothold adaptation for dynamic locomotion through CNNs. *IEEE Robotics and Automation Letters*, 2019.
- [146] Ajay Mandlekar, Jonathan Booher, Max Spero, Albert Tung, Anchit Gupta, Yuke Zhu, Animesh Garg, Silvio Savarese, and Li Fei-Fei. Scaling robot supervision to hundreds of hours with roboturk: Robotic manipulation dataset through human reasoning and dexterity. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019.
- [147] Ajay Mandlekar, Danfei Xu, Roberto Martín-Martín, Silvio Savarese, and Li Fei-Fei. Learning to generalize across long-horizon tasks from human demonstrations. In *Robotics: Science and Systems*, 2020.
- [148] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *Conference on Robot Learning*, 2021.

- [149] Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj Narang, Linxi Fan, Yuke Zhu, and Dieter Fox. MimicGen: A data generation system for scalable robot learning using human demonstrations. In *Conference on Robot Learning*, 2023.
- [150] Yutaro Matsuura, Kento Kawaharazuka, Naoki Hiraoka, Kunio Kojima, Kei Okada, and Masayuki Inaba. Development of a whole-body work imitation learning system by a biped and bi-armed humanoid. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2023.
- [151] Martin Meier, Florian Patzelt, Robert Haschke, and Helge J Ritter. Tactile convolutional networks for online slip and rotation detection. In *International Conference on Artificial Neural Networks*, 2016.
- [152] Adithyavairavan Murali, Arsalan Mousavian, Clemens Eppner, Adam Fishman, and Dieter Fox. CabiNet: Scaling neural collision detection for object rearrangement with procedural scene generation. In *IEEE International Conference on Robotics and Automation*, 2023.
- [153] Masaki Murooka, Iori Kumagai, Mitsuharu Morisawa, Fumio Kanehiro, and Abderrahmane Kheddar. Humanoid loco-manipulation planning based on graph search and reachability maps. *IEEE Robotics and Automation Letters*, 2021.
- [154] Keiji Nagatani, Tomonobu Hirayama, Akio Gofuku, and Yutaka Tanaka. Motion planning for mobile manipulator with keeping manipulability. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2002.
- [155] Shinichiro Nakaoka, Atsushi Nakazawa, Fumio Kanehiro, Kenji Kaneko, Mitsuharu Morisawa, and Katsushi Ikeuchi. Task model of lower body motion for a biped humanoid robot to imitate human dances. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005.

- [156] Soroush Nasiriany, Tian Gao, Ajay Mandlekar, and Yuke Zhu. Learning and retrieval from prior data for skill-based imitation learning. In *Conference on Robot Learning*, 2021.
- [157] Soroush Nasiriany, Huihan Liu, and Yuke Zhu. Augmenting reinforcement learning with behavior primitives for diverse manipulation tasks. In *IEEE International Conference on Robotics and Automation*, 2022.
- [158] Yuki Noguchi, Tatsuya Matsushima, Yutaka Matsuo, and Shixiang Shane Gu. Tool as embodiment for recursive manipulation. *arXiv preprint arXiv:2112.00359*, 2021.
- [159] Michael Noseworthy, Rohan Paul, Subhro Roy, Daehyung Park, and Nicholas Roy. Task-conditioned variational autoencoders for learning movement primitives. In *Conference on robot learning*, 2020.
- [160] 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. *arXiv preprint arXiv:2503.14734*, 2025.
- [161] OpenAI. GPT-4 technical report. *arXiv preprint 2303.08774*, 2023.
- [162] Amir Pagoli, Frédéric Chapelle, Juan-Antonio Corrales-Ramon, Youcef Mezouar, and Yuri Lapusta. Large-area and low-cost force/tactile capacitive sensor for soft robotic applications. *Sensors*, 2022.

- [163] Xinlei Pan, Tingnan Zhang, Brian Ichter, Aleksandra Faust, Jie Tan, and Sehoon Ha. Zero-shot imitation learning from demonstrations for legged robot visual navigation. In *IEEE International Conference on Robotics and Automation*, 2020.
- [164] Alexandros Paraschos, Christian Daniel, Jan R Peters, and Gerhard Neumann. Probabilistic movement primitives. In *Advances in Neural Information Processing Systems*, 2013.
- [165] Jyothish Pari, Nur Muhammad Shafiullah, Sridhar Pandian Arunachalam, and Lerrel Pinto. The surprising effectiveness of representation learning for visual imitation. In *Robotics: Science and Systems*, 2022.
- [166] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, 2019.
- [167] Georgios Pavlakos, Dandan Shan, Ilija Radosavovic, Angjoo Kanazawa, David Fouhey, and Jitendra Malik. Reconstructing hands in 3D with transformers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.
- [168] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *IEEE/CVF International Conference on Computer Vision*, 2023.
- [169] Luigi Penco, Brice Clément, Valerio Modugno, E Mingo Hoffman, Gabriele Nava, Daniele Pucci, Nikos G Tsagarakis, J-B Mouret, and Serena Ivaldi. Robust real-time whole-body motion retargeting from human to humanoid. In *IEEE-RAS International Conference on Humanoid Robots*, 2018.

- [170] Luigi Penco, Nicola Scianca, Valerio Modugno, Leonardo Lanari, Giuseppe Oriolo, and Serena Ivaldi. A multimode teleoperation framework for humanoid loco-manipulation: An application for the icub robot. *IEEE Robotics & Automation Magazine*, 2019.
- [171] Xue Bin Peng, Glen Berseth, KangKang Yin, and Michiel Van De Panne. DeepLoco: Dynamic locomotion skills using hierarchical deep reinforcement learning. *ACM Transactions on Graphics*, 2017.
- [172] Xue Bin Peng, Erwin Coumans, Tingnan Zhang, Tsang-Wei Edward Lee, Jie Tan, and Sergey Levine. Learning agile robotic locomotion skills by imitating animals. In *Robotics: Science and Systems*, 2020.
- [173] Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, and Angjoo Kanazawa. AMP: Adversarial motion priors for stylized physics-based character control. *ACM Transactions on Graphics*, 2021.
- [174] Amartya Purushottam, Yeongtae Jung, Christopher Xu, and Joao Ramos. Dynamic mobile manipulation via whole-body bilateral teleoperation of a wheeled humanoid. *IEEE Robotics and Automation Letters*, 2024.
- [175] FangXi Qi, Lei Xu, Yin He, Han Yan, and Hao Liu. PVDF-based flexible piezoelectric tactile sensors: Review. *Crystal Research and Technology*, 2023.
- [176] Ahmed Hussain Qureshi, Yinglong Miao, Anthony Simeonov, and Michael C Yip. Motion planning networks: Bridging the gap between learning-based and classical motion planners. *IEEE Transactions on Robotics*, 2020.
- [177] Nathan Ratliff, Matt Zucker, J Andrew Bagnell, and Siddhartha Srinivasa. CHOMP: Gradient optimization techniques for efficient motion planning. In *IEEE International Conference on Robotics and Automation*, 2009.

- [178] Joseph M Romano, Kaijen Hsiao, Günter Niemeyer, Sachin Chitta, and Katherine J Kuchenbecker. Human-inspired robotic grasp control with tactile sensing. *IEEE Transactions on Robotics*, 2011.
- [179] Javier Romero, Dimitrios Tzionas, and Michael J Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics*, 2017.
- [180] 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*, 2022.
- [181] Saeed Saeedvand, Masoumeh Jafari, Hadi S Aghdasi, and Jacky Baltes. A comprehensive survey on humanoid robot development. *The Knowledge Engineering Review*, 2019.
- [182] Kallol Saha, Vishal Mandadi, Jayaram Reddy, Ajit Srikanth, Aditya Agarwal, Bipasha Sen, Arun Singh, and Madhava Krishna. EDMP: Ensemble-of-costs-guided diffusion for motion planning. In *IEEE International Conference on Robotics and Automation*, 2024.
- [183] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photo-realistic text-to-image diffusion models with deep language understanding. In *Advances in Neural Information Processing Systems*, 2022.
- [184] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022.

- [185] Matteo Saveriano, Fares J Abu-Dakka, Aljaž Kramberger, and Luka Peternel. Dynamic movement primitives in robotics: A tutorial survey. *The International Journal of Robotics Research*, 2023.
- [186] Matthias Schöpfer, Carsten Schürmann, Michael Pardowitz, and Helge Ritter. Using a piezo-resistive tactile sensor for detection of incipient slippage. In *International Symposium on Robotics) and German Conference on Robotics*, 2010.
- [187] John Schulman, Yan Duan, Jonathan Ho, Alex Lee, Ibrahim Awwal, Henry Bradlow, Jia Pan, Sachin Patil, Ken Goldberg, and Pieter Abbeel. Motion planning with sequential convex optimization and convex collision checking. *The International Journal of Robotics Research*, 2014.
- [188] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [189] Luis Sentis and Oussama Khatib. A whole-body control framework for humanoids operating in human environments. In *IEEE International Conference on Robotics and Automation*, 2006.
- [190] Mingyo Seo, Ryan Gupta, Yifeng Zhu, Alexy Skoutnev, Luis Sentis, and Yuke Zhu. Learning to walk by steering: Perceptive quadrupedal locomotion in dynamic environments. In *IEEE International Conference on Robotics and Automation*, 2023.
- [191] Mingyo Seo, Steve Han, Kyutae Sim, Seung Hyeon Bang, Carlos Gonzalez, Luis Sentis, and Yuke Zhu. Deep imitation learning for humanoid loco-manipulation through human teleoperation. In *IEEE-RAS International Conference on Humanoid Robots*, 2023.

- [192] Mingyo Seo, Yoonyoung Cho, Yoonchang Sung, Peter Stone, Yuke Zhu, and Beomjoon Kim. PRESTO: Fast motion planning using diffusion models based on key-configuration environment representation. In *IEEE International Conference on Robotics and Automation*, 2025.
- [193] Mingyo Seo, H. Andy Park, Shenli Yuan, Yuke Zhu, and Luis Sentis. LEGATO: Cross-embodiment imitation using a grasping tool. *IEEE Robotics and Automation Letters*, 2025.
- [194] Alessandro Settimi, Danilo Caporale, Przemyslaw Kryczka, Mirko Ferrati, and Lucia Pallottino. Motion primitive based random planning for loco-manipulation tasks. In *IEEE-RAS International Conference on Humanoid Robots*, 2016.
- [195] Rutav Shah, Shuijing Liu, Qi Wang, Zhenyu Jiang, Sateesh Kumar, Mingyo Seo, Roberto Martín-Martín, and Yuke Zhu. MimicDroid: In-context learning for humanoid manipulation from human play videos. In *IEEE International Conference on Robotics and Automation*, 2026.
- [196] Siqi Shang, Mingyo Seo, Yuke Zhu, and Lillian Chin. FORTE: Tactile force and slip sensing on compliant fingers for delicate manipulation. *IEEE Robotics and Automation Letters*, 2026.
- [197] Kenneth Shaw, Shikhar Bahl, and Deepak Pathak. VideoDex: Learning dexterity from internet videos. In *Conference on Robot Learning*, 2023.
- [198] Benjamin Shih, Dylan Drotman, Caleb Christianson, Zhaoyuan Huo, Ruffin White, Henrik I Christensen, and Michael T Tolley. Custom soft robotic gripper sensor skins for haptic object visualization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.
- [199] Neo Ee Sian, Kazuhito Yokoi, Shuuji Kajita, Fumio Kanehiro, and Kazuo Tanie. Whole body teleoperation of a humanoid robot -development of a simple master

- device using joysticks-. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2002.
- [200] Laura Smith, Nikita Dhawan, Marvin Zhang, Pieter Abbeel, and Sergey Levine. AVID: Learning multi-stage tasks via pixel-level translation of human videos. In *Robotics: Science and Systems*, 2020.
- [201] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.
- [202] Shuran Song, Andy Zeng, Johnny Lee, and Thomas Funkhouser. Grasping in the wild: Learning 6DoF closed-loop grasping from low-cost demonstrations. *IEEE Robotics and Automation Letters*, 2020.
- [203] Maks Sorokin, Jie Tan, C Karen Liu, and Sehoon Ha. Learning to navigate sidewalks in outdoor environments. *IEEE Robotics and Automation Letters*, 2022.
- [204] Kaustubh Sridhar, Souradeep Dutta, Dinesh Jayaraman, James Weimer, and Insup Lee. Memory-consistent neural networks for imitation learning. In *International Conference on Learning Representations*, 2024.
- [205] Mike Stilman, Koichi Nishiwaki, and Satoshi Kagami. Humanoid teleoperation for whole body manipulation. In *IEEE International Conference on Robotics and Automation*, 2008.
- [206] Austin Stone, Ted Xiao, Yao Lu, Keerthana Gopalakrishnan, Kuang-Huei Lee, Quan Vuong, Paul Wohlhart, Sean Kirmani, Brianna Zitkovich, Fei Xia, Chelsea Finn, and Karol Hausman. Open-world object manipulation using pre-trained vision-language models. In *Conference on robot learning*, 2024.
- [207] Tomomichi Sugihara and Mitsuharu Morisawa. A survey: dynamics of humanoid robots. *Advanced Robotics*, 2020.

- [208] Balakumar Sundaralingam, Siva Kumar Sastry Hari, Adam Fishman, Caelan Garrett, Karl Van Wyk, Valts Blukis, Alexander Millane, Helen Oleynikova, Ankur Handa, Fabio Ramos, Nathan Ratliff, and Dieter Fox. cuRobo: Parallelized collision-free minimum-jerk robot motion generation. *arXiv preprint arXiv:2310.17274*, 2023.
- [209] Susumu Tachi, Kiyoshi Komoriya, Kazuya Sawada, Takashi Nishiyama, Toshiyuki Itoko, Masami Kobayashi, and Kozo Inoue. Telexistence cockpit for humanoid robot control. *Advanced Robotics*, 2003.
- [210] Toru Takenaka, Takashi Matsumoto, and Takahide Yoshiike. Real time motion generation and control for biped robot -1st report: Walking gait pattern generation-. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009.
- [211] 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*, 2018.
- [212] Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012.
- [213] Ryan L Truby, Lillian Chin, Annan Zhang, and Daniela Rus. Fluidic inner-vascularization sensorizes structures from a single build material. *Science advances*, 2022.
- [214] Vassilios Tsounis, Mitja Alge, Joonho Lee, Farbod Farshidian, and Marco Hutter. DeepGait: Planning and control of quadrupedal gaits using deep reinforcement learning. *IEEE Robotics and Automation Letters*, 2020.
- [215] Albert Tung, Josiah Wong, Ajay Mandlekar, Roberto Martín-Martín, Yuke Zhu, Li Fei-Fei, and Silvio Savarese. Learning multi-arm manipulation through

- collaborative teleoperation. In *IEEE International Conference on Robotics and Automation*, 2021.
- [216] Julen Urain, Michele Ginesi, Davide Tateo, and Jan Peters. ImitationFlow: Learning deep stable stochastic dynamic systems by normalizing flows. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020.
- [217] Eugene Valassakis, Georgios Papagiannis, Norman Di Palo, and Edward Johns. Demonstrate once, imitate immediately (DOME): Learning visual servoing for one-shot imitation learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2022.
- [218] Jean Chagas Vaz and Paul Oh. Material handling by humanoid robot while pushing carts using a walking pattern based on capture point. In *IEEE International Conference on Robotics and Automation*, 2020.
- [219] Filipe Veiga, Herke Van Hoof, Jan Peters, and Tucker Hermans. Stabilizing novel objects by learning to predict tactile slip. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015.
- [220] Vincent Wall, Gabriel Zöller, and Oliver Brock. Passive and active acoustic sensing for soft pneumatic actuators. In *IEEE International Conference on Robotics and Automation*, 2023.
- [221] Chen Wang, Linxi Fan, Jiankai Sun, Ruohan Zhang, Li Fei-Fei, Danfei Xu, Yuke Zhu, and Anima Anandkumar. MimicPlay: Long-horizon imitation learning by watching human play. In *Conference on Robot Learning*, 2023.
- [222] Chen Wang, Haochen Shi, Weizhuo Wang, Ruohan Zhang, Li Fei-Fei, and C Karen Liu. DexCap: Scalable and portable mocap data collection system for dexterous manipulation. In *Robotics: Science and Systems*, 2024.

- [223] Yanzhe Wang, Hao Wu, Haotian Guo, and Huixu Dong. GelSight FlexiRay: Breaking planar limits by harnessing large deformations for flexible,full-coverage multimodal sensing. *arXiv preprint arXiv:2411.18979*, 2024.
- [224] Ziyu Wang, Alexander Novikov, Konrad Zolna, Josh S Merel, Jost Tobias Springenberg, Scott E Reed, Bobak Shahriari, Noah Siegel, Caglar Gulcehre, Nicolas Heess, and Nando de Freitas. Critic regularized regression. In *Advances in Neural Information Processing Systems*, 2020.
- [225] Tim Welschhold, Christian Dornhege, and Wolfram Burgard. Learning mobile manipulation actions from human demonstrations. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.
- [226] Patrick M Wensing and David E Orin. Improved computation of the humanoid centroidal dynamics and application for whole-body control. *International Journal of Humanoid Robotics*, 2016.
- [227] G Westling and Roland S Johansson. Responses in glabrous skin mechanoreceptors during precision grip in humans. *Experimental Brain research*, 1987.
- [228] Heather E Wheat, Lauren M Salo, and Antony W Goodwin. Human ability to scale and discriminate forces typical of those occurring during grasp and manipulation. *Journal of Neuroscience*, 2004.
- [229] Alexander W Winkler, C Dario Bellicoso, Marco Hutter, and Jonas Buchli. Gait and trajectory optimization for legged systems through phase-based end-effector parameterization. *IEEE Robotics and Automation Letters*, 2018.
- [230] Josiah Wong, Albert Tung, Andrey Kurenkov, Ajay Mandlekar, Li Fei-Fei, Silvio Savarese, and Roberto Martín-Martín. Error-aware imitation learning from teleoperation data for mobile manipulation. In *Conference on Robot Learning*, 2022.

- [231] Fei Xia, Chengshu Li, Roberto Martín-Martín, Or Litany, Alexander Toshev, and Silvio Savarese. ReLMoGen: Leveraging motion generation in reinforcement learning for mobile manipulation. In *IEEE International Conference on Robotics and Automation*, 2021.
- [232] Zhou Xian, Nikolaos Gkanatsios, Theophile Gervet, Tsung-Wei Ke, and Katerina Fragkiadaki. Chaineddiffuser: Unifying trajectory diffusion and keypose prediction for robotic manipulation. In *Conference on Robot Learning*, 2023.
- [233] Fan Xie, Alexander Chowdhury, M De Paolis Kaluza, Linfeng Zhao, Lawson Wong, and Rose Yu. Deep imitation learning for bimanual robotic manipulation. In *Advances in Neural Information Processing Systems*, 2020.
- [234] Zhaoming Xie, Xingye Da, Buck Babich, Animesh Garg, and Michiel van de Panne. GLiDE: Generalizable quadrupedal locomotion in diverse environments with a centroidal model. *arXiv preprint arXiv:2104.09771*, 2021.
- [235] Mengda Xu, Zhenjia Xu, Cheng Chi, Manuela Veloso, and Shuran Song. XSkill: Cross-embodiment skill discovery. In *Conference on Robot Learning*, 2023.
- [236] Wenfu Xu, Heng Zhang, Han Yuan, and Bin Liang. A compliant adaptive gripper and its intrinsic force sensing method. *IEEE Transactions on Robotics*, 2021.
- [237] Yashuai Yan, Esteve Valls Mascaro, and Dongheui Lee. ImitationNet: Unsupervised human-to-robot motion retargeting via shared latent space. In *IEEE-RAS International Conference on Humanoid Robots*, 2023.
- [238] Ruihan Yang, Minghao Zhang, Nicklas Hansen, Huazhe Xu, and Xiaolong Wang. Learning vision-guided quadrupedal locomotion end-to-end with cross-modal transformers. In *International Conference on Learning Representations*, 2022.

- [239] Yuxiang Yang, Tingnan Zhang, Erwin Coumans, Jie Tan, and Byron Boots. Fast and efficient locomotion via learned gait transitions. In *Conference on Robot Learning*, 2022.
- [240] Vickie Ye, Georgios Pavlakos, Jitendra Malik, and Angjoo Kanazawa. Decoupling human and camera motion from videos in the wild. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.
- [241] Sarah Young, Dhiraj Gandhi, Shubham Tulsiani, Abhinav Gupta, Pieter Abbeel, and Lerrel Pinto. Visual imitation made easy. In *Conference on Robot Learning*, 2021.
- [242] Chenning Yu and Sicun Gao. Reducing collision checking for sampling-based motion planning using graph neural networks. In *Advances in Neural Information Processing Systems*, 2021.
- [243] Tianhe Yu, Pieter Abbeel, Sergey Levine, and Chelsea Finn. One-shot composition of vision-based skills from demonstration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019.
- [244] Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.
- [245] Kevin Zakka, Andy Zeng, Pete Florence, Jonathan Tompson, Jeannette Bohg, and Debidatta Dwibedi. XIRL: Cross-embodiment inverse reinforcement learning. In *Conference on Robot Learning*, 2022.
- [246] Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3D Diffusion Policy: Generalizable visuomotor policy learning via simple 3d representations. In *Robotics: Science and Systems*, 2024.

- [247] Annan Zhang, Ryan L Truby, Lillian Chin, Shuguang Li, and Daniela Rus. Vision-based sensing for electrically-driven soft actuators. *IEEE Robotics and Automation Letters*, 2022.
- [248] Annan Zhang, Lillian Chin, Daniel L Tong, and Daniela Rus. Embedded air channels transform soft lattices into sensorized grippers. In *IEEE International Conference on Robotics and Automation*, 2024.
- [249] Clark Zhang, Jinwook Huh, and Daniel D Lee. Learning implicit sampling distributions for motion planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018.
- [250] Mingyuan Zhang, Zhongang Cai, Liang Pan, Fangzhou Hong, Xinying Guo, Lei Yang, and Ziwei Liu. MotionDiffuse: Text-driven human motion generation with diffusion model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.
- [251] Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *IEEE International Conference on Robotics and Automation*, 2018.
- [252] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *IEEE/CVF International Conference on Computer Vision*, 2021.
- [253] Jialiang Zhao, Naveen Kuppaswamy, Siyuan Feng, Benjamin Burchfiel, and Edward Adelson. PolyTouch: A robust multi-modal tactile sensor for contact-rich manipulation using tactile-diffusion policies. In *IEEE International Conference on Robotics and Automation*, 2025.

- [254] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Robotics: Science and Systems*, 2023.
- [255] Yuheng Zhi, Nikhil Das, and Michael Yip. DiffCo: Auto-differentiable proxy collision detection with multi-class labels for safety-aware trajectory optimization. *IEEE Transactions on Robotics*, 2022.
- [256] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3d data processing. *arXiv preprint arXiv:1801.09847*, 2018.
- [257] You Zhou, Jianfeng Gao, and Tamim Asfour. Learning via-point movement primitives with inter- and extrapolation capabilities. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019.
- [258] Yifeng Zhu, Abhishek Joshi, Peter Stone, and Yuke Zhu. VIOLA: Imitation learning for vision-based manipulation with object proposal priors. In *Conference on Robot Learning*, 2022.
- [259] Yifeng Zhu, Zhenyu Jiang, Peter Stone, and Yuke Zhu. Learning generalizable manipulation policies with object-centric 3d representations. In *Conference on Robot Learning*, 2023.
- [260] Yifeng Zhu, Arisrei Lim, Peter Stone, and Yuke Zhu. Vision-based manipulation from single human video with open-world object graphs. *arXiv preprint arXiv:2405.20321*, 2024.
- [261] J Zico Kolter and Andrew Y Ng. The Stanford LittleDog: A learning and rapid replanning approach to quadruped locomotion. *The International Journal of Robotics Research*, 2011.
- [262] Gabriel Zöller, Vincent Wall, and Oliver Brock. Active acoustic contact sensing for soft pneumatic actuators. In *IEEE International Conference on Robotics and Automation*, 2020.