Marche Polytechnic University

DRIM

Ph.D. Program of National Interest in Robotics and Intelligent Machines

Administrative Headquarters: Università di Genova

Human-Guided Learning and Control for Contact-Rich Robotic Manipulation: An Interoperable Toolchain

by

Albin Bajrami

Thesis submitted for the degree of *Doctor of Philosophy* (38° cycle)

April 2026

Matteo Claudio Palpacelli — Supervisor

Antonio Sgorbissa — Head of the PhD program

Borsa di dottorato cofinanziata con risorse dell'Unione europea–*NextGeneration EU*.
Piano Nazionale di Ripresa e Resilienza Missione 4, componente 1 "Potenziamento dell'offerta dei servizi di istruzione: dagli asili nido all'Università".

Department of Computer Science, Bioengineering, Robotics and Systems Engineering

"You begin for one reason, and you finish for another."

To my mother, the familiar and obvious face who carried me through difficulties, and to Eiden, the new face in whom I have seen the hope of the future.

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Albin Bajrami
April 2026

Acknowledgements

I wish to thank all the people who, in different ways, have contributed to this PhD journey.

I thank my family for their unconditional support and for always being a steady presence throughout the years. A special thank you goes to my mother, for her calls;to my elder brother, for welcoming me into his home in Australia, and for rekindling a fire of hope within me; and to Eiden, for the lightness and hope that his presence brought into my life.

I am grateful to the colleagues and friends I met in Finland, and in particular to my mentors, for their guidance, patience, and technical support.

I am grateful to Professor Matteo Claudio Palpacelli for the role he played in my research path.

Finally, I thank myself for the determination, what the Finns call *sisu*, which allowed me to keep going through turbulence and uncertainty.

Abstract

Continuous, contact-rich manipulation is increasingly relevant for industrial applications such as surface finishing and polishing, where value is created through sustained, controlled interaction between tool and workpiece. Despite its practical importance, reliable execution remains challenging because key process parameters can vary during operation (e.g., contact conditions, friction, surface properties, tool compliance and wear), making purely model-based tuning brittle. This thesis investigates whether automated, learning-based readaptation, specifically Reinforcement Learning (RL), can improve robustness under such variability, while preserving the structure and safety of classical control.

To address the problem from multiple fronts, the thesis bridges Learning from Demonstration (LfD) and Reinforcement Learning (RL) within a human-in-the-loop, deployment-oriented workflow built around NVIDIA Isaac Lab. A low-cost and open-source LfD pipeline based on multi-view RGB sensing is introduced to capture operator demonstrations, reconstruct task-relevant motion, and retarget it to the robot through constrained inverse kinematics and safety-aware filtering, producing executable trajectories and waypoints that are imported into Isaac Lab as task references. Within the same Isaac Lab framework, a modular contact-rich simulation environment is developed to study interaction dynamics and to train structured RL policies (PPO) layered on operational-space control, enabling adaptive behaviours such as impedance modulation under explicit constraints and targeted randomization of contact conditions.

Experiments on a polishing benchmark show more consistent task execution and improved robustness to changing contact conditions when RL is combined with structured control and demonstration-derived references. Finally, a PCA-based analysis is introduced to reframe policy evaluation in physical terms: by extracting dominant modes from interaction and control signals, it highlights which variables drive contact

behaviour and how they couple, providing a principled way to reason about what the agent must learn to regulate effectively beyond scalar performance metrics.

Table of contents

List of figures

List of tables

Part I

Section One

Chapter 1

Context and Purpose

> "A complex system that works is
> invariably found to have evolved
> from a simple system that worked."
>
> John Gall (Gall's Law)

Contact-rich robotic operations (e.g., polishing, wiping, tight-fit assembly) require the coordinated control of trajectory and interaction forces while operating under uncertainty in friction, surface geometry, and environmental compliance. In industrial settings, these uncertainties are the norm rather than the exception, often leading to imperfect execution, surface defects, or the need for rework.

Classical control approaches, such as operational space control (OSC) and impedance control (IC), provide robust and reproducible behaviors by explicitly shaping the robot–environment interaction. However, their performance strongly depends on manually tuned parameters, which limits their ability to adapt to variations in surfaces, tools, and boundary conditions. As a consequence, achieving consistent performance across heterogeneous tasks and operating conditions remains challenging.

In this context, Reinforcement Learning (RL) has emerged as a promising paradigm to adapt contact strategies and control parameters in a data-driven manner. Rather than replacing classical controllers, learning-based approaches can be leveraged to modulate or augment them, potentially improving robustness and adaptability. Despite this potential, the transfer of learned policies from simulation to real robotic systems, remains a major open challenge, particularly for contact-rich manipulation.

1.1 Overarching Research Question and Thesis Perspective

This thesis addresses the overarching research question of how contact-rich manipulation skills can be taught, adapted, and reliably executed on real robotic systems under uncertainty and limited sensing, while preserving robustness, interpretability, and deployability.

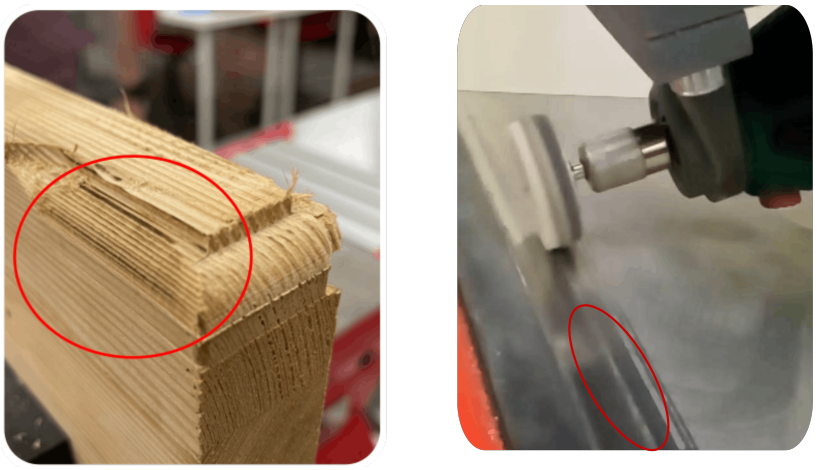
The originality of this thesis does not lie in a single standalone learning algorithm, but in the design of an interoperable and deployment-oriented methodology for contact-rich manipulation. In contrast to approaches that treat demonstration learning, contact adaptation, controller design, and behavioral analysis as largely separate problems, this work integrates them into a unified learning-from-interaction pipeline.

The central thesis claim is that contact-rich manipulation can be made more robust and practically deployable by combining three elements within the same framework: (i) human-guided task specification through LfD, (ii) structured RL for execution adaptation under uncertainty, and (iii) classical control backbones with explicit safety and stability constraints. In this perspective, learning does not replace the controller; rather, it is embedded within the control structure in order to preserve physical interpretability, constrained behavior, and real-world applicability.

Accordingly, this thesis adopts a system-level viewpoint that combines control, learning, perception, and diagnostics into a coherent and reproducible toolchain. From this standpoint, the main contribution is methodological and integrative: not only to study whether these components work in isolation, but to show how they can be connected in a practical framework for contact-rich robotic manipulation.

1.2 Motivation from Real-World Manufacturing Scenarios

The interest in learning-based control for surface-finishing operations is motivated by practical manufacturing scenarios. During polishing or contour milling, execution may be imperfect due to material heterogeneity, surface defects, or unforeseen disturbances. In such cases, it is necessary to define appropriate system behavior: whether the robot should reprocess the component, under which conditions rework should occur,



(a) (b)

Figure 1.1 Manufacturing defects observed during processing: (a) defect occurring during wood contour milling (burn marks); (b) insufficient polishing of an aluminum component.

and whether corrective actions should be applied immediately or deferred until batch completion.

Analogous issues arise, for example, in contour milling of wooden components in the presence of knots or material irregularities, as illustrated in Fig. 1.1. Here, the robot must decide whether to repeat the operation unchanged or adapt process parameters. In current industrial practice, these decisions are typically taken by a skilled operator based on experience and visual inspection. Moreover, the training and deployment pipeline should remain practical even when target surfaces are complex and accurate CAD models are unavailable.

1.3 Frameworks and Software Artifacts

In line with the proposed perspective, this thesis presents and develops two open-source software frameworks (published online on GitHub (Bajrami, 2026a,b)) with the objective of providing ready-to-use tools while also serving as experimental platforms for the investigation of learning-based contact control under realistic constraints.

Specifically, the following components are developed:

- a Residual RL and OSC framework based on waypoint-defined reference trajectories (Sec. 4.4.2), in which a learning policy modulates incremental updates of OSC parameters (e.g., ΔK_p, $\Delta\zeta$) and, optionally, joint torques;
- an RL and Learning from Demonstration (LfD) pipeline (Sec. 4.5) for skill replication from human demonstrations.

Although the LfD pipeline was originally developed in a different application context (Sec. 3.3), it is included here as a practical and low-cost solution to the skill acquisition problem addressed in this thesis. Its results (Sec. 3.7) are directly reusable within the contact-rich manipulation framework investigated here.

1.4 Objectives and Research Questions

The main objective of this thesis is to develop new protocols and algorithms for an interactive human–robot teaching process for contact-rich manipulation, addressing the problem from multiple but complementary perspectives. On the one hand, a human-centred approach is adopted, in which the operator directly teaches the task to be executed. On the other hand, learning-based control strategies are investigated to adapt execution under uncertainty. In addition, the integration of low-cost, general-purpose RGB-D sensing into this process is explored.

The work is structured around four complementary pillars: (i) learning-based modulation of contact controllers, (ii) human-guided skill acquisition under limited sensing, (iii) interpretability and reproducibility of learned behaviors, and (iv) robustness and readiness for real-world deployment. The corresponding research questions are formulated as follows:

1. *RL + OSC for contact:* to what extent can a reinforcement learning policy embedded within an Operational Space Control (OSC) structure stabilize the contact transient response and improve robustness under domain randomization (e.g., varying static friction), while improving force regulation along the interaction normal compared to pure OSC baselines?
2. *Interpretability via PCA:* is it possible to define a Principal Component Analysis (PCA)-based behavioral fingerprint of an RL-augmented OSC controller, useful for early diagnosis and reproducible comparisons across different training runs?

3. *Low-cost LfD with 2×RGB:* can a pipeline based on two low-cost RGB-D cameras, using only their RGB streams, produce sufficiently reliable trajectories and waypoints for robotic execution?

4. *Robustness:* which randomizations and which safety layers (hard constraints on forces, velocities, and limits) are necessary to achieve robust training and prepare deployment on real hardware?

1.5 Thesis Contributions

The following contributions jointly address the overarching research question by covering complementary aspects of skill acquisition, execution, analysis, and deployment for contact-rich manipulation. Their novelty lies not only in the individual components, but also in their integration into a coherent and reusable framework:

C1 End-to-end LfD pipeline with 2×RGB on the robot: a complete architecture integrating 2D detection on dual RGB streams, 3D triangulation with calibration, transformation into the robot frame, retargeting (IK), and command streaming, including explicit safety constraints and real-time requirements. Unlike more instrumented or marker-based demonstration systems, the proposed pipeline targets low-cost sensing and direct generation of robot-executable trajectories and waypoints.

C2 Trainable and constrained LfD+RL formulation: demonstrations specify the *what* (trajectory and waypoints), while RL optimizes the *how* (execution), using a discrete action set and hard constraints to prevent inadmissible actions and promote stability. In contrast to end-to-end policy learning approaches, the proposed formulation preserves a structured separation between task specification and control adaptation.

C3 Experimental setup for contact and comparison baselines: implementation and evaluation of OSC-based configurations for contact tasks (e.g., polishing), including randomized conditions and baseline OSC-parameter optimization procedures with force- and stability-oriented metrics. Unlike nominal evaluations carried out under fixed contact conditions, the proposed setup explicitly studies robustness under changing interaction parameters.

C4 Diagnostic analysis and interpretability: introduction of a PCA-based behavioral fingerprint for RL-augmented OSC controllers, supporting run-to-run comparison and diagnostic analysis beyond task-level metrics. In contrast to evaluations based only on reward or success rate, this contribution provides a physically grounded view of how interaction and control variables jointly characterize learned behavior.

1.6 Thesis Scope and Research Approach

This thesis focuses on a methodological and software pipeline for contact-rich robotic manipulation. The pipeline combines (i) trajectory generation via LfD with 2×RGB vision and waypoint tracking, (ii) reinforcement learning in simulation under safety constraints and an OSC control structure, and (iii) behavioral analysis and diagnostics based on dimensionality-reduction tools for interpretability and comparison.

The research follows a modular and system-level approach, in which different components of the learning-from-interaction pipeline are investigated in depth. While each component addresses a specific challenge, all of them contribute to the same overarching objective of enabling robust, interpretable, and deployable contact-rich manipulation.

Throughout the Ph.D. path, multiple methodological challenges in robotics and AI were addressed. When relevant, the corresponding knowledge and tools are reported within their original application contexts and explicitly transferred to the contact-rich manipulation setting investigated here.

The overall aim of this thesis is to propose and demonstrate new methodologies for interactive human–robot operation in contact-rich scenarios, supported by working protocols, software frameworks, and qualitative and quantitative experimental results.

1.7 Limitations

- **Incomplete end-to-end validation:** the reported results should be interpreted as demonstrators or partial validations; a full end-to-end evaluation of the complete pipeline (LfD → RL → hardware deployment) has not yet been completed.

- **LfD predominantly in free space:** contact-aware reproduction, including robust management of contact transitions and detachment phases, is treated as an extension rather than as a fully consolidated component.
- **Engineering constraints and platform dependence:** some architectural choices are motivated by real-time and computational constraints, which may limit generality across different hardware and software platforms.
- **Restricted action space:** the adoption of compact action spaces and hard constraints improves stability and trainability but may limit exploration and generalization to more diverse tasks and conditions.

These limitations also define clear directions for future research and extensions of the proposed frameworks.

1.8 Thesis Organization

- **Ch. 2 – Background and Related Work.** OSC and impedance/admittance control; RL for contact-rich manipulation (residual learning, impedance modulation); low-cost vision LfD via dual-RGB keypoint triangulation and task-space tracking.
- **Ch. 3 – A Learning-from-Demonstration Framework for Human-Guided Skill Acquisition.** A preparatory low-cost vision LfD setup on a redundant robotic arm to capture human demonstrations and produce motion references.
- **Ch. 4 – Reinforcement Learning-Based Training and Analysis Pipeline for Contact-Rich Tasks.** A modular RL pipeline with two methods: Procedural Trajectory Mode (residual RL over OSC/impedance) and Trajectory by Learning from Demonstration, with task rewards and logging.
- **Ch. 5 – Results.** Presents outcomes from the pipeline: i) Procedural Trajectory Mode (incl. friction randomization); ii) Trajectory by Learning from Demonstration; iii) a PCA-based behavioural fingerprint to interpret RL-augmented OSC.
- **Ch. 6 – Conclusion.** Findings, open-source release, future work

Chapter 2

Background and Related Work

"No plan survives first contact with the enemy."

Helmuth von Moltke the Elder (1871)

In this chapter, key conceptual foundations are reviewed with respect to robotic controllers, RL applied to robotics, and LfD strategies. The objective is to contextualize the contributions of this thesis and to outline the state of the art at the intersection of robotics and AI. The chapter first introduces the main control frameworks employed in this work, then discusses how RL is commonly integrated with model-based control structures, and finally reviews LfD approaches relevant to trajectory specification and execution.

The control background is used to frame two distinct strategies that are central to this thesis. In the first strategy, RL is integrated directly with the controller in contact-rich settings by acting on the controller parameters, i.e., by modulating the control behavior within an OSC-based backbone. In the second strategy, a low-cost vision LfD pipeline is used to obtain reference motions, while RL does not modify controller parameters but instead operates at the level of task-space motion references, which are executed by a fixed-parameter tracking controller. This distinction reflects the structure of the thesis and, as introduced in Chapter 1, the fact that two distinct software codebases are presented. The chapter concludes with a synthesis of the reviewed literature, clarifying how these works inform the thesis approach and which gaps the thesis explicitly targets.

2.1 Modern Foundations of Robotic Controllers

The modern treatment of contact in manipulation originates from the idea that a robot should not impose position or force independently, but rather a *mechanical behavior*, a target relation between motion and interaction forces.

Impedance control formalized this perspective by prescribing a mass–spring–damper behavior at the end effector, enabling compliant motion and stable exchanges of power with the environment (Hogan (1985)). Its conceptual dual, admittance control, instead leverages force measurements to generate motion commands, allowing the realization of equivalent compliant behaviors under typical actuator and sensing constraints (Hogan (1985)). Together, these controllers established the vocabulary of compliant interaction that underpins most contact-rich tasks, from assembly and peg-in-hole operations to surface finishing and force-regulated manipulation (Keemink et al. (2018); Schumacher et al. (2019); Suomalainen et al. (2022); Zhang et al. (2025)).

A complementary line of work shaped how those behaviors are realized in task space. Hybrid position/force control separated constrained and unconstrained directions, regulating force where contact exists while tracking trajectories along the tangent directions (Raibert and Craig (1981)). Operational Space Control (OSC) then provided a principled way to express dynamics and control laws directly in task coordinates, decoupling end-effector motion and null-space objectives and making force regulation at the tool a first-class citizen of the control problem (Khatib (1987)). In practice, these ideas translate into controllers that track a reference pose while modulating stiffness and damping, switching smoothly to force objectives when contact is detected.

At a lower level of the control stack, classical Proportional - Integral - Derivative (PID) regulation remains the workhorse for stabilizing motion and rejecting disturbances, and it is still the dominant feedback structure in industrial control practice (Åström and Hägglund (2001)). The appeal of PID lies in its minimal parametrization coupled with a broad applicability, provided that practical non-idealities such as actuator saturation, measurement noise, and sampling are handled explicitly (Åström and Hägglund (1995); Li et al. (2006); Visioli (2006)).

In this thesis, two control backbones are employed depending on the task regime. For contact-rich behaviors, OSC is combined with a Cartesian impedance objective: the end-effector is regulated toward a reference pose while realizing a desired mass–spring–damper behavior in task space. Within this formulation, learning is integrated at

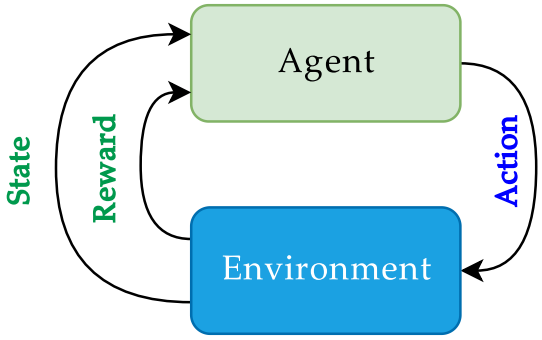


Figure 2.1 Conceptual agent–environment interaction loop in RL.

the level of interaction behavior rather than at the actuator level: the policy shapes the closed-loop compliance by modulating impedance parameters (stiffness and damping).

For free-space motion and trajectory execution, a Cartesian PD tracking controller is adopted. In this case, the policy acts at the level of task-space references by selecting incremental updates to the Cartesian setpoint, which are then tracked by the PD loop. This separation is intentional: it keeps the learning problem expressed in compact, physically meaningful task-space variables, while the underlying feedback controller enforces stability and yields a predictable closed-loop response.

2.2 Reinforcement Learning (RL) and Deep RL

RL provides a general framework for learning decision-making policies through interaction with an environment. The problem is commonly formalized as a Markov Decision Process (MDP), in which an agent receives an observation (or state), selects an action according to a policy, and receives a scalar reward that encodes the task objective (Fig. 2.1) (Kaelbling et al. (1996); Sutton and Barto (2018)). Learning aims at maximizing the expected return, typically defined as the discounted sum of future rewards, by improving the policy or the associated value functions (Sutton and Barto (2018)).

Deep Reinforcement Learning (DRL) extends RL by using deep function approximators, typically implemented as deep neural networks, to represent policies and/or value functions, enabling operation in high-dimensional state spaces and with rich sensory inputs (Arulkumaran et al. (2017)). In practice, DRL algorithms can be organized around value-based methods, which learn action values and derive a policy

implicitly, and policy-based methods, which optimize the policy parameters directly; actor–critic formulations combine both principles (Arulkumaran et al. (2017); Sutton and Barto (2018)). Despite their expressive power, DRL methods often face practical challenges such as sample inefficiency, sensitivity to reward specification, and limited robustness when deployment conditions deviate from those experienced during training (Henderson et al. (2018)). These aspects motivate structured training procedures and architectural choices that constrain learning to meaningful action channels and preserve predictable closed-loop behavior when integrated into robotic systems.

2.2.1 RL for Contact-Rich Manipulation

The use of RL has been increasingly adopted for robotic manipulation tasks in which interaction dynamics are difficult to model accurately and challenging to tune manually, particularly when friction, compliance, and intermittent micro-contacts dominate performance. Over the past decade, a substantial body of work has focused on canonical contact tasks such as insertion and assembly, disassembly, polishing, stacking, opening doors, and pushing objects (Belousov et al. (2022); Kuo et al. (2021); Luo et al. (2019); Martín-Martín et al. (2019); Nemec et al. (2017); Serrano-Muñoz et al. (2023); Zhang et al. (2020)), rope manipulation, fabric folding, and tensioning and cutting of flexible materials (Bednarek and Walas (2019); Liu et al. (2021); Luo et al. (2018); Pedram et al. (2020); Petrík and Kyrki (2019); Shin et al. (2019); Tsurumine et al. (2019)), just to cite some examples where sparse rewards and tight tolerances stress exploration and robustness (Elguea-Aguinaco et al. (2023); Schoettler et al. (2020)).

Extended surface-interaction tasks, including wiping, scrubbing, and polishing, require coordinating a moving contact patch while regulating normal force under uncertain and variable friction. In this regime, purely end-to-end learning is often complemented by structure: RL may act within hierarchical pipelines where learned decisions are paired with model-based components that enforce feasibility and force-related constraints, and prior work reports transfer from simulation to real wiping under such structured designs (Lew et al. (2022)). Surface finishing is characterized by broader task and process variability, and research increasingly targets the interplay between force regulation, contact transients, and interaction uncertainty.

In industrial polishing, learning-based approaches often target online adaptation of impedance-related quantities (e.g., stiffness and damping), improving contact stability and process outcomes relative to fixed-gain baselines (Cramer et al. (2025); Ding et al.

(2023)). Overall, these trends support the use of learning to refine contact-relevant behavior within structured control architectures, rather than as a wholesale replacement of task-space controllers (Cramer et al. (2025); Elguea-Aguinaco et al. (2023)).

A prominent family of structured approache is *residual* learning on top of model-based control. In residual RL, a nominal controller (e.g., OSC with an impedance objective) achieves the bulk of the task, while a learned policy contributes a bounded corrective action that specializes to the modeling errors that matter. This division of labor can improve learning efficiency and robustness by constraining the policy to physically meaningful action channels while preserving the stabilizing structure of the underlying controller. In the context of continuous-contact tasks such as polishing, residual formulations are particularly natural when learning is used to modulate impedance-related parameters, since this directly shapes contact stability and force regulation while remaining compatible with task-space control structure and practical safety mechanisms (Cramer et al. (2025)).

At the same time, survey literature highlights persistent limitations, including sample inefficiency, sensitivity to reward design, and brittleness when contact conditions deviate from those encountered during training (Elguea-Aguinaco et al. (2023)).

In summary, several methodologies and tools are proposed to address contact interaction, which, despite extensive work, remains an open problem. For this reason, a software codebase is presented that implements multiple operating modalities, pure OSC as a baseline, OSC augmented with RL, and a pure joint-torque control mode, and can be used as an analysis tool to investigate this class of issues. In addition, as anticipated, initial solutions enabled by the use of this codebase are also presented.

2.3 LfD with Low-Cost Vision Hardware

Traditional motion–capture (mocap) systems offer high-fidelity measurements of human motion at the cost of expensive hardware, body markers and constrained capture volumes. An attractive alternative for collecting demonstrations in unstructured settings is to rely on commodity RGB or RGB–D cameras. Recent advances in human pose and hand-tracking enable the extraction of keypoints and gesture cues from monocular video in real time (Cao et al. (2017); Lugaresi et al. (2019)). When multiple calibrated (Zhang (2000a)) viewpoints are available, simple triangulation (Hartley and

Sturm (1997)) yields metrically consistent 3-D trajectories at a fraction of the cost of mocap.

Within this context, employing low-cost RGB cameras, instead of specialized sensing systems, in combination with reinforcement learning (RL) constitutes an effective and scalable option for perception-driven robotic control. On the one hand, such sensors enable the acquisition of demonstration trajectories and visual signals that can be used to supervise, initialize, or guide the learning process (learning from demonstration and its extensions) (Argall et al. (2009); Hester et al. (2018); Levine and Koltun (2013); Torabi et al. (2018)); on the other hand, the agent, via a learned policy, assumes the role of the closed-loop decision-making module, replacing a conventional controller in selecting the actions required to realize a desired trajectory under perceptual feedback (Kalashnikov et al. (2018); Levine et al. (2016a,b)). In this formulation, the policy is not a mere trajectory replay mechanism, but rather a perception-guided decision rule capable of adapting to environmental variability and unmodeled uncertainties (e.g., disturbances, moderate changes in the scene and objects) (Kalashnikov et al. (2018); Tobin et al. (2017)). Moreover, coupling the policy with general safety mechanisms (e.g., constrained RL, safety layers, shielding) can improve robustness and reduce the likelihood of unsafe or infeasible actions both during learning and at deployment (Achiam et al. (2017); Alshiekh et al. (2017); Dalal et al. (2018); García and Fernández (2015)). This design choice yields practical benefits, including reduced sensing cost and integration complexity, diminished reliance on accurate models and manual controller tuning, and improved transferability across platforms and operating conditions, thereby supporting generalization to variations in objects, initial poses, and moderate perturbations (Kalashnikov et al. (2018); Tobin et al. (2017)).

2.4 Synthesis

The literature reviewed in this chapter shows that the main components of this thesis are individually well established. Classical control provides structured and physically interpretable tools for contact-rich manipulation, Reinforcement Learning offers adaptation under uncertainty, and Learning from Demonstration enables task specification from human examples.

Table 2.1 Qualitative positioning of the thesis with respect to representative research directions discussed in this chapter.

Approach / direction	**Low-cost LfD**	**Contact adapt.**	**Control backbone**	**Safety**	**Interpretability**	**Integrated**
Marker-/teleop-based LfD	partial	no	partial	partial	partial	no
End-to-end RL	no	yes	no	partial	no	no
Structured / residual RL	no	yes	yes	partial	partial	partial
Diagnostic / PCA analysis	no	no	partial	n/a	yes	no
This thesis	**yes**	**yes**	**yes**	**yes**	**yes**	**yes**

However, these directions are often pursued separately. Many RL approaches for contact-rich manipulation emphasize performance, but rely on end-to-end formulations or task-specific settings that reduce interpretability and deployment control. Conversely, many LfD approaches focus on motion acquisition, but do not connect low-cost demonstrations to contact-adaptive control under explicit safety constraints. Likewise, diagnostic analysis is often treated as a downstream evaluation tool rather than as an integral part of controller comparison.

From this perspective, the gap addressed by this thesis is not the absence of methods for control, learning, or demonstration taken in isolation, but the lack of a coherent framework that combines them for contact-rich manipulation. As summarized in Table 2.1, fewer works jointly address low-cost trajectory acquisition, structured RL embedded in an OSC/impedance backbone, robustness under randomized contact conditions, explicit safety constraints, and physically interpretable analysis of learned behavior.

This thesis is positioned at that intersection. Its novelty lies in an interoperable and deployment-oriented methodology that connects low-cost LfD, constrained RL-based adaptation, classical contact control, and PCA-based behavioral diagnostics into a unified toolchain for contact-rich robotic manipulation.

The table is not intended as an exhaustive survey, but as a compact summary of the methodological positioning of this thesis with respect to the main strands of related work.

Part II

Section Two

Chapter 3

A Learning-from-Demonstration Framework for Human-Guided Skill Acquisition

"Plans are worthless, but planning is indispensable."

Dwight D. Eisenhower (1957)

The activities reported in this chapter were developed within the FOCAAL project during the author's Ph.D. programme, and the author acknowledges the Italian National Research Council (CNR-ISTC) for granting access to the *TIAGo* platform and the Rehabilitation Department of the Ospedale di Torrette, Ancona (Italy), for providing the clinical environment and physical space for testing; in particular, the author thanks Dr. Gabriella Cortellessa and Prof. Maria Gabriella Ceravolo as FOCAAL project leads, and Prof. Marianna Capecci and Dr. Gloria Beraldo as technical facilitators during setup and experiments.

3.1 Introduction to Learning from Demonstration (LfD)

The methodology under discussion is known as LfD, or alternatively as imitation learning (IL) or programming by demonstration (PbD) (The term LfD will be used henceforth in order to encompass the various nomenclatures), facilitates the acquisition

of skills in robots through the analysis of exemplars, thereby offering a more efficient learning method in comparison to traditional approaches that require an initial learning phase.

Early work by Schaal (1999) highlighted imitation as a route to humanoid robotics, linking motor primitives and perception–action coupling. Argall et al. (2009) provided the first comprehensive survey, categorizing design choices (demonstrator, policy derivation, problem space) and limitations. More recent treatments broadened the field: Billard et al. (2008) framed LfD as a paradigm for intuitive robot programming, while Chernova and Thomaz (2022) emphasized interaction with human teachers, including naive users.

Across these perspectives, LfD consistently emerges as a means to reduce exploration, accelerate learning, and enable robots to acquire complex skills through natural human interaction. In this scenario, the human expertise of the system administrator is transferred to the craftsman.

LfD can be broadly grouped into three categories, depending on how the demonstrations are provided to the robot (Ravichandar et al. (2019)):

- **Kinesthetic teaching**, where the human physically guides the robot through the motion (Maeda et al. (2017); Pervez and Lee (2018); Shavit et al. (2018)).
- **Teleoperation**, where the human controls the robot remotely via an interface such as a joystick, glove, or VR setup (Abbeel et al. (2010); Mohseni-Kabir et al. (2015); Peters et al. (2003); Whitney et al. (2019)).
- **Passive observation**, where the robot infers actions by watching a human perform the task, often through vision-based pose estimation and retargeting (Codevilla et al. (2018); Dillmann (2004); Schulman et al. (2016); Vogt et al. (2017)).

Each of these modalities comes with advantages and limitations. Kinesthetic teaching is highly intuitive in collaborative robots, as the operator can physically move the robot in free-drive mode to demonstrate a desired trajectory. However, this approach is generally less precise than teleoperation or direct programming, and is often complemented by other methods. In practice, hybrid strategies are common: for example, a rough trajectory can be taught kinesthetically, and then refined either by adjusting waypoint parameters or by using teleoperation tools such as a keyboard,

joystick, or dedicated interface. A representative example is reported in Bajrami et al. (2024), where an initial coarse path is taught through kinesthetic teaching and teleoperation, and subsequently refined through direct programming and fine-tuning of key process variables, including precise waypoint positions, velocities, accelerations, and contact forces. Teleoperation itself scales more easily to remote or hazardous tasks, yet the quality of the demonstration depends strongly on the interface and on the operator's skill. Passive observation is the most natural for humans, as it does not require specialized hardware or direct interaction, but it poses significant challenges in perception and in mapping human motions to the robot embodiment.

Robot teaching through LfD is widely regarded as one of the most intuitive and direct strategies to instruct a robot, including for novice users. Once the teaching interface and the corresponding software framework have been implemented by an expert operator, no additional effort is required from the end user, provided that the software is designed for simplicity of use. Following these principles, the work carried out with the *TIAGo* robot consists of a human gesture mimicry framework in which the motion of the human arm is represented through shoulder and elbow angles, rather than only through wrist position. Gestures are acquired using RGB cameras and converted into trajectories that the robot can execute online.

This activity was developed during the Ph.D. program with a dual objective. On the one hand, it addressed the requirements of the FOCAAL project, where human gesture imitation on a service robot represented a central research challenge. On the other hand, it was intentionally designed to anticipate and support the methodological needs of this thesis, by establishing a reusable LfD architecture on a redundant manipulator.

The resulting framework constitutes the foundation of the LfD component later employed in the robotic polishing pipeline. The design choices and insights discussed in this chapter are therefore directly leveraged and extended in the RL-based pipeline described in Chapters 4 and 5.

3.2 Context, Robot, and Task

This chapter introduces the context in which the LfD component of the proposed interoperable learning and control toolchain was developed. Rather than presenting a standalone application study, the focus is on the methodological and architectural

insights derived from a realistic deployment scenario, which directly inform the human-guided learning components later integrated into contact-rich robotic manipulation tasks.

The work described in this chapter was conducted during the Ph.D. program within the framework of the FOCAAL (FOg Computing in Ambient Assisted Living) project. While FOCAAL targets service and medical robotics applications, it provided a constrained, real-world environment in which requirements such as safety, usability, online execution, and robustness to human variability significantly influenced the design of the proposed LfD architecture. These constraints are central to the broader goals of this thesis and are therefore leveraged here to develop a reusable and transferable learning framework.

The FOCAAL project aims to integrate commercial wearable sensors, social robots, and AI services into a distributed Edge–Fog–Cloud architecture to support intensive inpatient rehabilitation. Within this setting, robotic platforms are not limited to passive monitoring roles, but act as active interlocutors in the delivery of motor and cognitive stimuli, requiring natural and intuitive human–robot interaction. In this thesis, FOCAAL is used as a development testbed to study human-guided motion acquisition and reproduction under realistic interaction constraints, rather than to evaluate clinical outcomes.

Study design The experimental protocol comprises two main sessions with hospital inpatients: (i) an assessment session combining standard clinical scales with instrumented measurements collected via wearable sensors and robotic platforms, and (ii) an interactive session in which the robot guides motor exercises and cognitive tasks while tracking user performance. Wearable devices collect physiological and activity-related signals over extended periods, while the robotic platforms provide kinematic measurements and interaction-related indices.

Beyond the base protocol, an upper-limb interaction scenario was developed using the TIAGo robot, novel non-invasive sensing, and learning-based methods. This activity was intentionally designed to serve a dual objective: addressing the requirements of the FOCAAL project while simultaneously establishing a methodological foundation for the Learning from Demonstration pipeline adopted in this thesis.

(a) TIAGo (PAL Robotics). (b) Ohmni (OhmniLabs).

Figure 3.1 Robotic platforms used in FOCAAL: (a) TIAGo (PAL Robotics) and (b) Ohmni (OhmniLabs).

Edge–Fog–Cloud architecture and services The FOCAAL infrastructure distributes computational responsibilities across three layers. At the edge, wearable devices and robotic platforms perform local pre-processing, such as skeleton tracking and lightweight perception tasks. The fog layer aggregates multi-source data streams and executes more computationally demanding analyses, generating events and indicators to support operators. At the cloud layer, anonymized data are stored and made available through dashboards for longitudinal analysis and monitoring.

From the perspective of this thesis, this architecture provides a realistic integration context in which online learning, data synchronization, and interaction constraints must be addressed. However, the specific distributed implementation is not a contribution of this work and is reported here solely to contextualize the operational environment in which the proposed LfD framework was developed.

Robotic platforms and sensing FOCAAL integrates two commercial robots with complementary roles (Fig. 3.1): TIAGo (PAL Robotics), a social robot equipped with a 7-DoF manipulator, RGB-D sensing, and mobile base; and Ohmni (OhmniLabs), a telepresence robot used primarily for communication and content delivery. Both platforms operate under ROS Noetic and provide reactive skills for perception, navigation, and interaction.

In the present work, TIAGo serves as the reference platform for upper-limb motor interaction. Its kinematic redundancy, combined with onboard perception and manipulation capabilities, makes it particularly suitable for studying human arm motion

representation and imitation. Importantly, TIAGo was selected not for its application domain, but as a development platform enabling the design of an LfD architecture that abstracts human motion and task representation from robot-specific kinematics. This abstraction is later reused in contact-rich manipulation scenarios involving different robotic hardware.

Reference task The reference case study considered in this thesis is the online imitation of human arm gestures by the *TIAGo* robot (Fig. 3.3), using markerless multi-view pose capture and kinematic retargeting to the robot arm. Although the task does not involve physical contact, it addresses fundamental challenges of human-guided skill acquisition, including motion representation, retargeting across embodiments, online execution, and robustness to inter-subject variability.

These aspects are intentionally isolated and studied here, as they constitute pre-requisites for the contact-rich learning and control pipelines developed in subsequent chapters. The resulting LfD framework therefore acts as a foundational component of the interoperable toolchain proposed in this thesis, enabling the transfer of human-guided skills to learning-based controllers operating under contact constraints.

Work development Vision-based retargeting based on non-specialized sensing and open-source software introduces several practical challenges. First, camera accuracy and pose detection quality critically affect the spatial localization of human arm joints. Second, for each robotic arm morphology, a dedicated mimicry module must be implemented and adapted to the specific kinematic structure and conventions. Third, the overall software development is tightly coupled to the adopted sensing and computing stack.

***i)* Camera-accuracy evaluation** Before developing the imitation pipeline and the online mimicry algorithm, the accuracy of the reconstruction was assessed on a set of functional gestures, such as drinking, touching the nose, closing a zipper, and pouring (Fig. 3.3). These gestures were first recorded offline using both a motion-capture system and a multi-view RGB setup. This procedure enabled an initial evaluation of reconstruction accuracy and robustness under realistic conditions, including partial occlusions and inter-subject variability. The work is therefore organized into two main components: *(i)* the development of the end-to-end mimicry framework, from

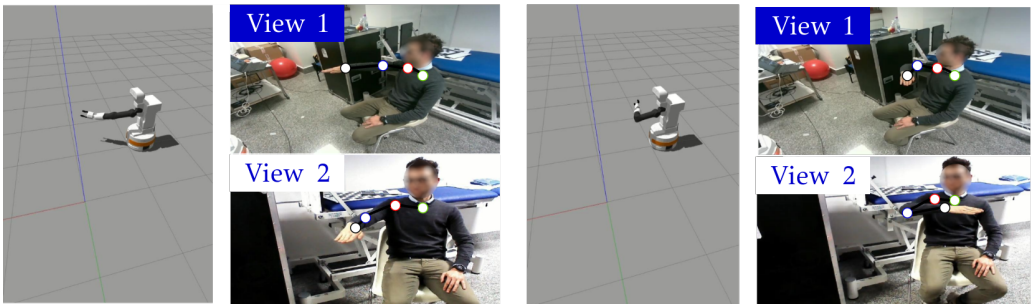


Figure 3.2 Schematic of the reference task: markerless multi-view capture and triangulation are used to reconstruct human arm motion, which is then retargeted to the *TIAGo* arm for execution.

post-triangulation processing through the mimicry module to robot execution; and *(ii)* the analysis of triangulation performance to support reliable gesture reconstruction.

***ii)* Mimicry algorithm development** The development of the mimicry algorithm depends on the morphology of the adopted robotic arm. Although inverse-kinematics methodologies follow the same general principles, each robot requires specific adjustments (e.g., offsets and joint-space mappings) to ensure that the reproduced gesture remains consistent with the demonstrated human motion. The mimicry algorithm was validated using MATLAB and motion-capture data, which provide higher-precision reference measurements.

***iii)* Software implementation** After evaluating the feasibility of the reconstruction and the mimicry module, the software codebase was developed in ROS 1 Noetic. Pose detection was implemented using NVIDIA DeepStream, triangulation was performed via OpenCV, and the robot-specific mimicry script was used for the mapping stage; finally, ROS nodes were used to stream the resulting commands to the robot for execution.

This chapter reports a feasibility study, focusing only on results that are instrumental for the subsequent RL-oriented pipeline. In particular, the reliability and robustness of markerless keypoint detection (and reconstruction) are evaluated by comparison against a motion-capture reference. Based on this analysis, a robustness strategy to improve keypoint detection is also introduced, intended for reuse within LfD systems.

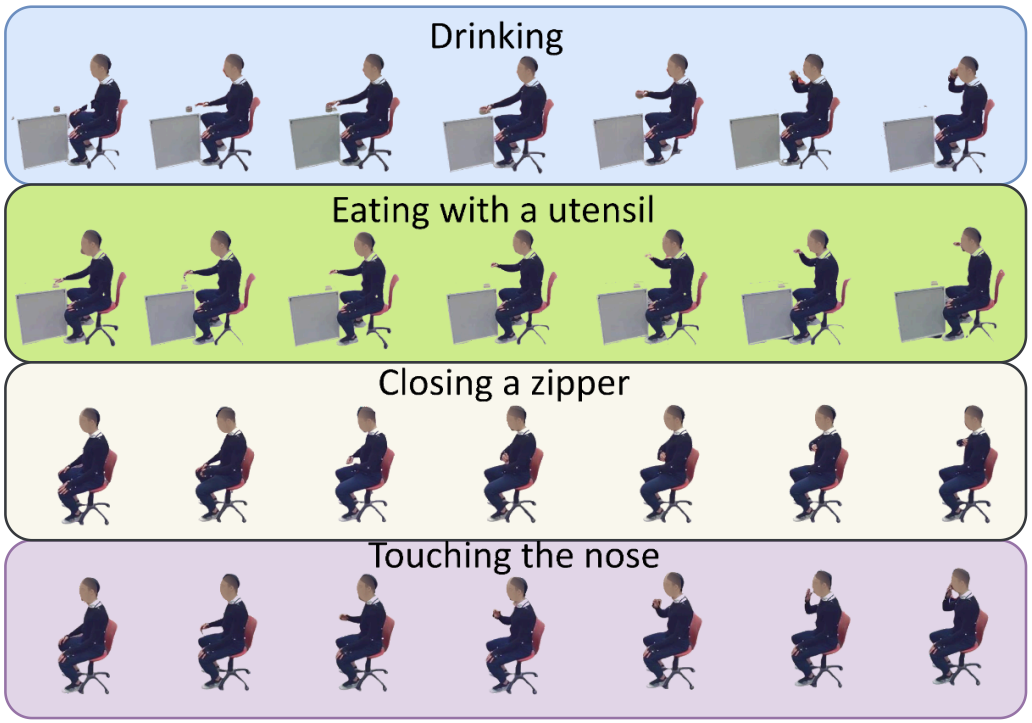


Figure 3.3 Functional gestures used to assess markerless multi-view reconstruction accuracy (from the to bottom: drinking, touching the nose, closing a zipper, and pouring).

3.3 Pipeline Overview

The pipeline (Fig. 3.4) comprises (i) 2D pose detection on two RGB streams, (ii) 3D triangulation of keypoints based on intrinsic/extrinsic calibration, and (iii) retargeting to *TIAGo* via a custom mapping from human joints to robot joint commands (inverse kinematics), followed by smoothing and the enforcement of safety limits. The architecture was tested first in simulation and then on real hardware using synchronized streams and *online* processing.

Pipeline description With reference to Fig. 3.4, two RGB cameras (Intel RealSense D455 and ASUS Xtion) provide 640×480 image streams at 15 fps to a ROS Melodic stack. The streams are processed within a Docker/DeepStream deployment to extract 2D right-arm keypoints from each view. Given intrinsic and extrinsic calibration, these keypoints are triangulated in OpenCV using a linear triangulation method Hartley and Sturm (1997) to recover 3D shoulder–elbow–wrist positions, which are subsequently

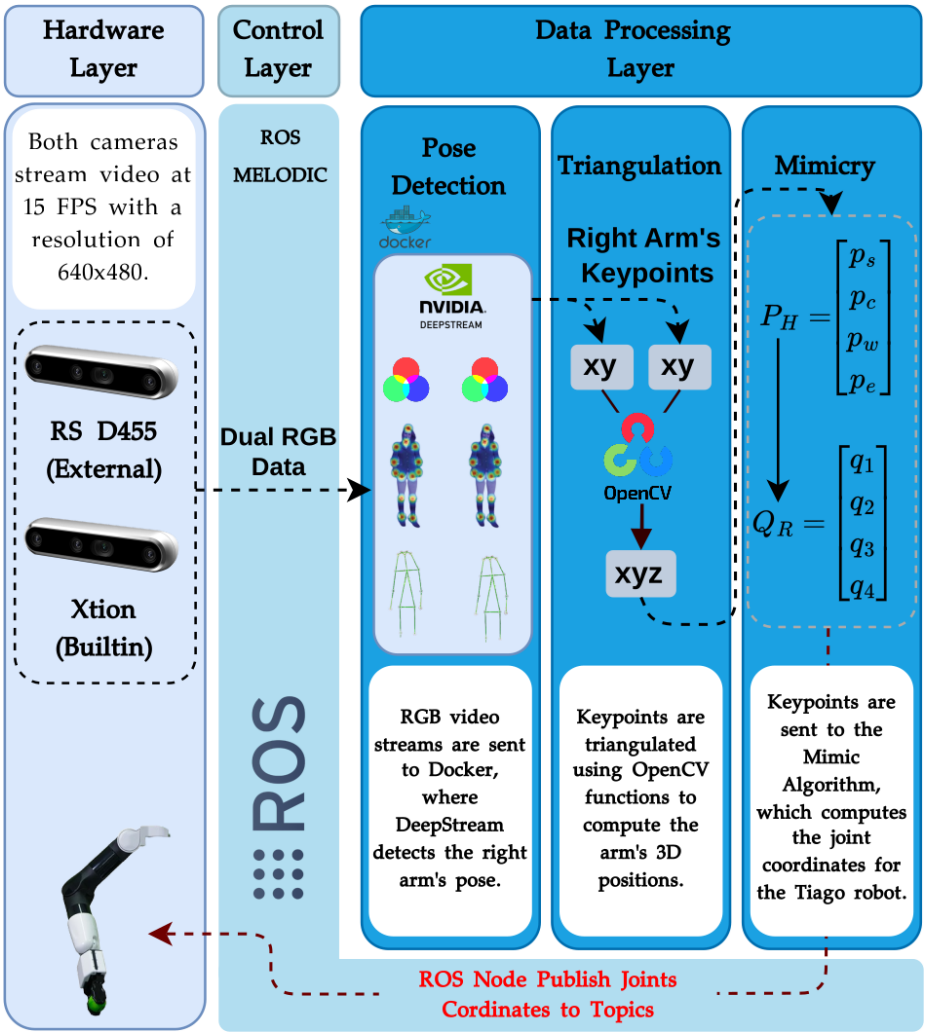


Figure 3.4 System architecture for real-time upper-limb mimicry with TIAGo.

expressed in the robot reference frame. A customized mimicry module then performs kinematic retargeting via damped least-squares inverse kinematics, applies smoothing and safety limits, and outputs *TIAGo* joint commands $q_1, \ldots, q_4$. Commands are published through ROS middleware.

Observations on the pipeline development process During the development of the framework, several practical considerations emerged. In relation to the aspects that are relevant for this thesis, a key factor is the runtime performance (fps) of the adopted pose-detection frameworks. In particular, after an initial attempt based on MediaPipe running on CPU, the implementation was migrated to NVIDIA DeepStream on GPU.

This change enabled a pose-detection rate above 15 fps. The increased detection rate provides the following benefits:

- it enables the mimicry framework to operate online;
- it supports multi-view triangulation in a fully streaming setting;
- it allows additional filtering steps (e.g., outlier rejection and temporal averaging) to improve signal reliability.

3.4 Hardware Specifications

This section summarizes the hardware and computational infrastructure adopted in this work, including the robotic platform, the sensing setup, and the host workstation used to run the compute-intensive modules. The goal is to provide sufficient detail to support reproducibility and to contextualize the performance figures reported throughout the chapter.

TIAGo platform The TIAGo mobile manipulator (Fig. 3.1) used in this work combines a differential-drive base (maximum linear speed 1 m/s, footprint ∅54 cm), a lifting torso with 350 mm stroke, a 7-DoF arm with a 3-DoF wrist, and a 2-DoF pan–tilt head. Overall height ranges from 110 to 145 cm due to the torso lift; total mass is 72 kg. The nominal arm reach is 87 cm with a payload of 2 kg. The end-effector may be equipped with a five-finger underactuated hand (Hey5) or a PAL parallel gripper. Power is supplied by a 36 V, 20 Ah battery. The torso expansion panel exposes GigE (2×), USB 3.0 (1×), USB 2.0 (1×), a 12 V/5 A auxiliary power output, and a CAN service connector. Table 3.1 reports gearbox reductions, maximum shaft speeds, nominal torques, and encoder resolutions for the four arm modules and the three wrist DoFs.

RGB-D sensors Two RGB-D devices are considered in this setup: the head-mounted camera integrated in TIAGo (Orbbec Astra S) and an external Intel RealSense D455. Both are shown in Fig. 3.6. Table 3.2 reports their main specifications.

Only the RGB stream is used for both cameras, while the depth stream is intentionally not used. The objective of the proposed pipeline is to study a general multi-view

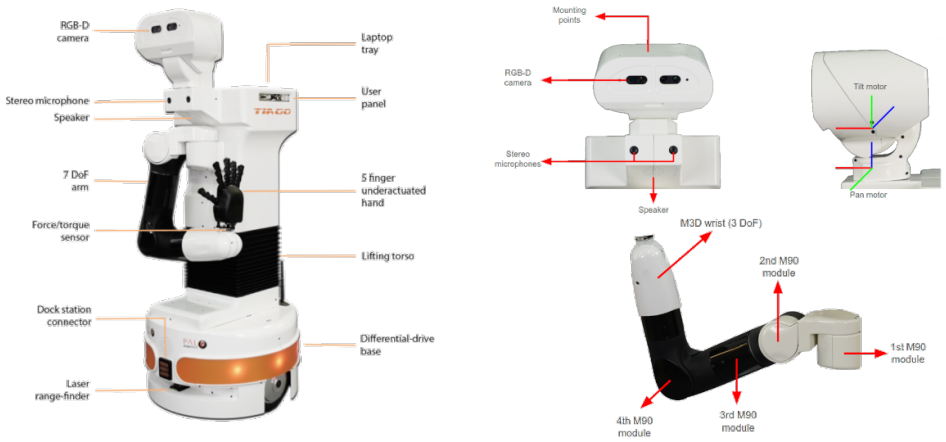


Figure 3.5 TIAGo platform with labelled subsystems.

reconstruction strategy based on 2-D pose detection and calibrated geometric triangulation, rather than a sensor-specific RGB-D lifting approach. This choice makes the method, in principle, transferable to a broader class of low-cost video sensors and allows 3-D reconstruction to benefit from multiple viewpoints, which is advantageous in the presence of occlusions and depth ambiguities. The choice of the external camera model itself was primarily determined by equipment availability rather than by a claim of superior sensing performance.

Computational Host and Software Environment Compute-intensive modules are executed on the host workstation, while the *TIAGo* onboard PC is used only for ROS communication and low-level control. The host runs ROS Noetic and performs OpenCV-based triangulation natively, whereas 2D pose detection is executed inside an

Table 3.1 TIAGo arm and wrist actuator specifications.

Joint	Red.	Max [rpm]	Torque [Nm]	Encoders [bits]	
				Motor	Absolute
Arm 1	100:1	18	39	12	12
Arm 2	100:1	18	39	12	12
Arm 3	100:1	22	22	12	12
Arm 4	100:1	22	22	12	12
Wrist 1	336:1	17	3	11	12
Wrist 2	336:1	17	5	11	13
Wrist 3	336:1	17	5	11	13

(a) TIAGo head RGB-D (Orbbec Astra S).

(b) External Intel RealSense D455.

Figure 3.6 RGB-D sensors used in the setup.

NVIDIA DeepStream–based Docker container with the NVIDIA runtime for direct GPU access. Tables 3.3 summarise the hardware and runtime components adopted in this project.

3.5 Vision System Calibration

Camera calibration is commonly described in terms of *intrinsic* and *extrinsic* components. Intrinsic calibration characterizes the optics and sensor properties of each camera, whereas extrinsic calibration estimates the relative pose between cameras. Both aspects are well established in the literature and widely adopted in industrial practice (Hartley and Sturm (1997); Zhang (2000a)). However, when using *AI-driven* pipelines for markerless 2D pose estimation and custom sensing setups (i.e., not based on industrial, factory-calibrated cameras), dedicated and repeatable calibration procedures are often required to ensure reliable multi-view reconstruction (MRPT Team (2024)). In the present setup, the two aspects are treated separately: intrinsic calibration is reported to characterize the individual sensors, while the metric extrinsic calibration procedure used to align the views is performed independently.

Table 3.2 Specification comparison of the RGB-D sensors.

Specification	**TIAGo head (Orbbec Astra S)**	**Intel RealSense D455**
Interface	USB 2.0	USB 3.1
Field of view	60° H, 49.5° V, 73° D	$\sim$ 86° H, $\sim$ 57° V (depth)
Color stream modes	320×240@30; 640×480@30; 1280×960@10	RGB sensor: yes
Depth stream modes	320×240@30; 640×480@30; 160×120@30	up to 1280×720@90
Depth operating range	0.4–2 m	0.6–6 m
Module dimensions	—	124×29×26 mm
Launch date	—	Q2'20

Table 3.3 System environment summary (host + container).

Item	**Host**	**Container**
OS / Kernel	Ubuntu 20.04.6 LTS; 5.15.0-134-generic	Ubuntu 22.04.5 LTS (Jammy)
CPU	AMD Ryzen 5 5600H (6 cores / 12 threads)	–
System memory	30.7 GiB	–
GPU	NVIDIA GeForce RTX 3050 (4 GiB VRAM)	–
Driver / CUDA / `nvcc`	535.183.01 / 12.2 / 12.2.91	`nvcc` 12.6.77; CUDA 12.x runtime components present
Docker Engine	27.3.1 (runtimes: `nvidia`, `runc`)	–
ROS	ROS 1 Noetic	ROS 1 Noetic
GStreamer (core)	–	1.20.3
Triton Inference Server	–	2.49.0
OpenCV (Python)	4.10.0	4.10.0
librealsense	2.55.1 (+ DKMS 1.3.27)	–
Storage / I/O	NVMe (1 TB, 512 GB) + USB external 1.8 TB; wired Ethernet; USB SS/HS	–

This point is particularly important in the present setup, since the 2-D body keypoints are detected independently in each camera view by separate image-processing stages. Differences in optics, distortion, viewpoint, image quality, and pose-estimation uncertainty imply that corresponding landmarks are not expected to coincide exactly at the image level. In the proposed system, cross-view geometric consistency is not established by assuming identical image measurements, but by expressing both views in a common reference frame through an explicit metric extrinsic calibration procedure. Intrinsic calibration is reported separately to characterize the optical properties of each sensor, whereas the estimation of the relative camera pose is carried out independently.

Practical calibration procedures can be grouped into a small number of recurring classes. *Planar-target* methods rely on checkerboard or ChArUco patterns to estimate intrinsics and multi-camera extrinsics, typically refined via bundle adjustment

Table 3.4 Intrinsic calibration (checkerboard 9×6, 50 mm

Parameter	**ASUS Xtion**	**Intel RealSense D455**
Resolution	640×480	640×480
f_x	521.61	634.86
f_y	520.02	634.10
c_x	321.19	640.77
c_y	235.16	412.07
k_1	0.0872	-0.0551
k_2	-0.4911	0.0702
p_1	0.00163	0.00164
p_2	-0.00475	0.00290
k_3	0.4909	-0.0228
RMS reprojection [px]	0.52	0.48
p_{95} reprojection [px]	1.45	1.35
# images	15	15

(OpenCV Contributors; Triggs et al. (2000); Zhang (2000b)). *3D-target* approaches use wand-based procedures or rigs with 3D spheres/markers, which are particularly suitable for complex geometries or large capture volumes (Mitchelson and Hilton (2003); Shin and Mun (2012)). *Fiducial-based* calibration exploits AprilTag/ArUco markers for rapid registration and hand–eye calibration in the form $AX = XB$ (Garrido-Jurado et al. (2014); Olson (2011); Park and Martin (1994); Tsai and Lenz (1989)). *Self-calibration methods* (SfM) perform target-less photogrammetric optimization constrained by motion, often formulated within a bundle-adjustment pipeline (Hartley and Zisserman (2004); Svoboda et al. (2005); Szeliski (2022); Triggs et al. (2000)). Finally, *learning-based/hybrid* approaches estimate calibration parameters via neural models and/or hybrid fitting, and are typically adopted in non-standard setups (Bogdan et al. (2018); Iyer et al. (2018)).

Intrinsic calibration Intrinsic calibration estimates the parameters of a pinhole camera model with radial–tangential distortion, yielding the camera matrix $\mathbf{K}$ and distortion coefficients. In this work, a planar checkerboard with 9×6 inner corners and 50 mm square size was used, and 15 images per camera were acquired with varied orientations and depths. The procedure was executed with `MRPT camera-calib`, which implements Zhang's method and produces YAML files containing $\mathbf{K}$, distortion coefficients, and a reprojection-error report (MRPT Team, 2024; Zhang, 2000a). Cali-

Table 3.5 Pseudocode of the manual metric extrinsic calibration (RealSense → TIAGo head, ZYX convention).

Step	Operation (pseudocode)
1	`Input:` laser measures $\{H_{\text{TI}}, H_{\text{RS}}, h_{\text{ref}}, d_{\text{TI-ref}}, d_{\text{RS-ref}}, d_{\text{TI-RS}}\}$; side $\in \{\text{SX}, \text{DX}\}$.
2	`// Preprocess heights` $h_{\text{TI}} \leftarrow H_{\text{TI}} - h_{\text{ref}}$; $h_{\text{RS}} \leftarrow H_{\text{RS}} - h_{\text{ref}}$. $\alpha \leftarrow \arcsin\big(h_{\text{TI}}/d_{\text{TI-ref}}\big)$; $\beta \leftarrow \arcsin\big(h_{\text{RS}}/d_{\text{RS-ref}}\big)$. $d^{H}_{\text{TI-ref}} \leftarrow d_{\text{TI-ref}} \cos\alpha$; $d^{H}_{\text{RS-ref}} \leftarrow d_{\text{RS-ref}} \cos\beta$.
3	`// Build points in world (chair reference at origin)` $\mathbf{p}_{\text{ref}} \leftarrow [0,0,0]^{\top}$; $\mathbf{p}_{\text{TI}} \leftarrow [0,\ d^{H}_{\text{TI-ref}},\ h_{\text{TI}}]^{\top}$. Compute $\mathbf{p}_{\text{RS}} = [x_{\text{RS}}, y_{\text{RS}}, h_{\text{RS}}]^{\top}$ using $d^{H}_{\text{RS-ref}}$ and side (SX/DX); enforce $\|\mathbf{p}_{\text{RS}} - \mathbf{p}_{\text{TI}}\| = d_{\text{TI-RS}}$.
4	`// Local frames (TI, RS)` For $S \in \{\text{TI}, \text{RS}\}$: $\hat{\mathbf{z}}_S \leftarrow -\mathbf{p}_S/\|\mathbf{p}_S\|$; $\mathbf{y}_0 \leftarrow [0,0,-1]^{\top}$ `// arbitrary reference vector to define` $\hat{\mathbf{x}}_S$. $\hat{\mathbf{x}}_S \leftarrow \dfrac{\mathbf{y}_0 \times \hat{\mathbf{z}}_S}{\|\mathbf{y}_0 \times \hat{\mathbf{z}}_S\|}$; $\hat{\mathbf{y}}_S \leftarrow \hat{\mathbf{z}}_S \times \hat{\mathbf{x}}_S$. $\mathbf{R}_S \leftarrow [\hat{\mathbf{x}}_S\ \hat{\mathbf{y}}_S\ \hat{\mathbf{z}}_S]$.
5	`// Relative transform (RS → TI)` $\mathbf{R} \leftarrow \mathbf{R}_{\text{TI}}^{\top}\mathbf{R}_{\text{RS}}$; $\mathbf{t} \leftarrow \mathbf{R}_{\text{TI}}^{\top}(\mathbf{p}_{\text{RS}} - \mathbf{p}_{\text{TI}})$. $\mathbf{T} \leftarrow \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^{\top} & 1 \end{bmatrix}$.
6	`// Euler angles (ZYX)` $\psi \leftarrow \text{atan2}(r_{21}, r_{11})$; $\theta \leftarrow \arcsin(-r_{31})$; $\phi \leftarrow \text{atan2}(r_{32}, r_{33})$.
7	`// Outputs` Return $\mathbf{T}$, baseline $\|\mathbf{t}\|$, and (ψ, θ, ϕ).
8	`// Checks` Orthonormality $\|\mathbf{R}^{\top}\mathbf{R} - \mathbf{I}\|_F < 10^{-3}$; distance consistency ($\pm 5$ mm); optional epipolar validation.

bration quality is summarized by the global RMS reprojection error and the per-image p_{95}. The distortion parameters are partitioned into radial (k_1, k_2, k_3) and tangential (p_1, p_2) components. The estimated parameters $(f_x, f_y, c_x, c_y, k_1, k_2, p_1, p_2, k_3)$ for each camera are reported in Table 3.4 and are used for undistortion and normalization in the subsequent triangulation.

Extrinsic calibration Extrinsic calibration estimates the rigid transformation between the camera frames, expressed as a rigid motion in $SE(3)$ parameterized by $(\mathbf{R}, \mathbf{t})$. After intrinsic calibration, relative poses are commonly recovered by jointly observing a known target; the resulting camera–camera transforms are then used for

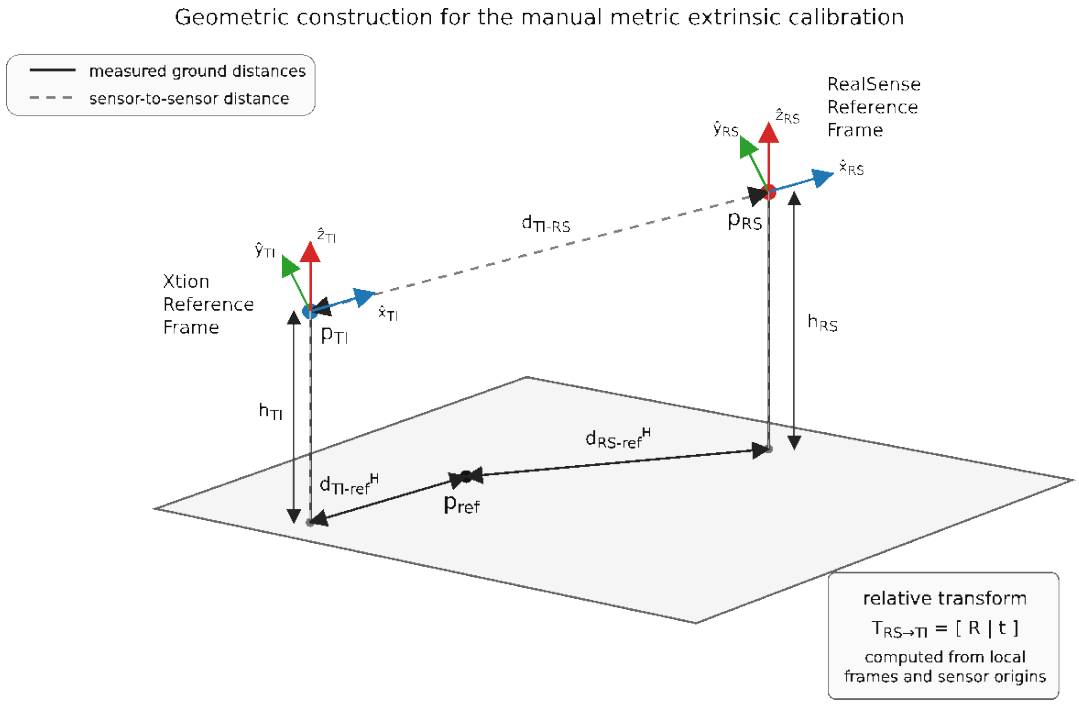


Figure 3.7 Supporting geometric figure for the pseudocode of the manual metric extrinsic calibration procedure.

3-D reconstruction and frame alignment. In this work, due to instability observed with purely vision-based procedures, a manual metric method was adopted. Specifically, key distances and heights were measured with a laser rangefinder by defining three points: the TIAGo head camera (ASUS Xtion), the external camera (Intel RealSense D455), and a shared reference point on the floor (chosen as an intersection between tiles) that is visible from both viewpoints.

From these measurements, the sensor origins $\mathbf{p}_{\mathrm{TI}}$ and $\mathbf{p}_{\mathrm{RS}}$ are first defined through a simple metric construction that uses a shared floor landmark (chosen as an intersection between tiles) as a stable reference for the distance/height measurements. The floor landmark is used only as a metrological reference and is not the operative coordinate origin of the system. For all subsequent processing, the resulting geometry is expressed in a common reference frame chosen as the TIAGo head-camera *optical* frame (ASUS Xtion). Two orthonormal frames are then constructed, and the relative transform is computed as

$$\mathbf{T}_{\mathrm{RS}\to\mathrm{TI}} = \begin{bmatrix} \mathbf{R}_{\mathrm{TI}}^{\top}\mathbf{R}_{\mathrm{RS}} & \mathbf{R}_{\mathrm{TI}}^{\top}(\mathbf{p}_{\mathrm{RS}} - \mathbf{p}_{\mathrm{TI}}) \\ \mathbf{0}^{\top} & 1 \end{bmatrix},$$

Figure 3.8 Final deployment for online mimicry, showing the two RGB viewpoints: TIAGo head camera (ASUS Xtion, used as reference optical frame) and an external Intel RealSense D455. Both sensors are RGB-D devices, but only RGB streams are used.

where $\mathbf{R}_{(\cdot)}$ and $\mathbf{p}_{(\cdot)}$ denote, respectively, the orientation and origin of each sensor frame expressed in the TIAGo head-camera optical frame. The full step-by-step procedure, including the construction of local frames, the computation of $[\mathbf{R}|\mathbf{t}]$, Euler extraction, and basic sanity checks, is reported in Table 3.5.

In the deployed setup (TIAGo head as base; RealSense expressed in TIAGo), the translation and orientation are $\mathbf{t} = [1.5131, -0.2227, -0.0418]^\top$ m with baseline $\|\mathbf{t}\| = 1.530\,\text{m}$, and ZYX Euler angles $[\psi, \theta, \phi] = [-18.6^\circ, -40.1^\circ, 6.6^\circ]$.

It should be noted that 2-D keypoints detected independently from different cameras are not expected to coincide exactly, due to differences in viewpoint, optics, image quality, and pose-estimation uncertainty. For this reason, each camera was first calibrated intrinsically using a checkerboard target, and multi-view consistency was then recovered through extrinsic calibration and triangulation in a common reference frame. Calibration quality was monitored through reprojection statistics, whereas overall accuracy was assessed at the level of the reconstructed 3-D motion rather than by direct pixel correspondence. The resulting reconstruction error, reported as RMSE across gestures and configurations in Sec. 3.7, indicates task-level accuracy sufficient for waypoint generation, while not targeting high-precision biomechanical measurement.

3.6 Final Setup

Figure 3.8 shows the final deployment used for *online* arm-motion mimicry. And Fig 3.9 shows a sequence of gesture imitation.

Figure 3.9 Sequence of gesture mimicry.

During preliminary tests, an important practical observation was that some functional gestures were reconstructed more reliably than others, largely as a consequence of the underlying keypoint configurations and visibility. In particular, for specific arm postures, the triangulation pipeline exhibited failures or oscillatory 3-D estimates despite temporal filtering. These artifacts propagated to the mimicry/retargeting stage and, in some cases, induced discontinuities in the inferred joint configuration (e.g., abrupt switching between inverse-kinematics solution branches), which manifested as instantaneous "flips" to the opposite joint-angle extreme.

Without entering implementation details that are not central to this thesis, the next section focuses on the main contribution of this line of work for the subsequent LfD pipeline: a study of keypoint *data quality* and reliability under low-cost, markerless sensing. This step is relevant both when tracking a single landmark and when tracking multiple points (e.g., human joint keypoints, joint-angle surrogates, or sets of visual markers), and it directly informs robust trajectory acquisition for downstream learning and execution.

The core focus of the remainder of the chapter is the *reliability and robustness* of gesture reconstruction under this kind of low-cost, markerless, multi-view pipeline. In particular, the next section reports the quantitative evidence that motivates the subsequent design choices (e.g., robustness strategies for keypoint detection and handling of occlusions) and provides the methodological foundation later reused for LfD-based trajectory specification in the polishing pipeline.

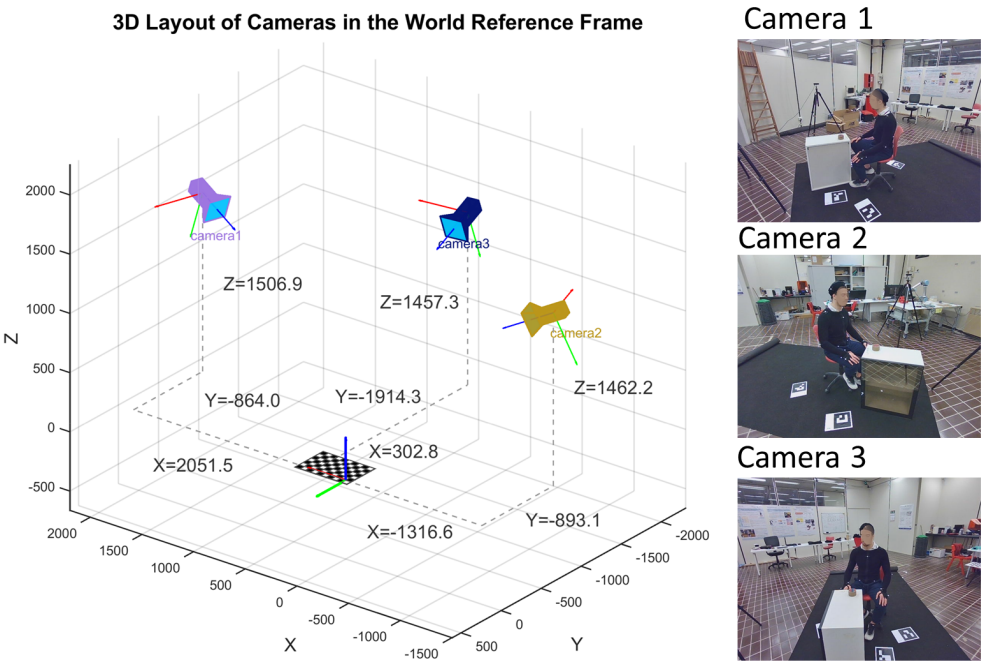


Figure 3.10 Reference experimental setup and gesture set.

3.7 Implications for Learning from Demonstration

The utilisation of motion-capture (mocap) systems for estimating spatial marker poses has been extensively documented in the literature (Merriaux et al. (2017); Moeslund et al. (2006)). These systems have been shown to provide high measurement accuracy, typically at the millimetre level under appropriate calibration and setup conditions (Merriaux et al. (2017); Windolf et al. (2008)). They also support high acquisition rates suitable for real-time tracking (Elfring et al. (2010); Khadem et al. (2000)). Conversely, these systems generally require more complex hardware setups, calibration procedures, and external devices (e.g., wearable markers) than commodity camera-based alternatives (Conconi et al. (2021); Merriaux et al. (2017)).

The utilisation of general-purpose vision systems, augmented by artificial intelligence (AI), is intended to operate within contexts where absolute accuracy is not imperative and inaccuracies are tolerable, with the objective of achieving cost reductions or expedited setup times.

The work presented here, which forms the basis of the RL + LfD section, aims to provide guidelines for the implementation of a multi-camera system (2 or more) that uses triangulation (comparing different strategies) to define an approach that allows

Table 3.6 Summary of the acquisition setup and camera geometry.

Item	**Value**
RGB-D cameras	3 × Orbbec Gemini 335L
RGB frame rate / resolution	30 Hz / 640 × 480 px
MoCap baseline	12 × OptiTrack cameras @ 120Hz
Extrinsic calibration	Chessboard 9 × 6 internal corners, 50 mm squares
Camera distances to world origin	|| CAM1: 2.69 m;
	|| CAM2: 2.16 m;
	||CAM3: 2.42 m
Relative view angles (w.r.t. global Z)	$\angle$(CAM1,CAM3): 70.27°
	$\angle$(CAM3,CAM2): 74.96°

for better detection of one or more points (e.g., a set of 4 highly reflective markers) using only RGB images. This configuration, which necessitates solely RGB images, enables, in principle, the utilisation of any video sensor (smartphone/webcam, etc.). Each case must be considered individually, with regard to both sensor performance and available computing power. However, in principle, images with a resolution of 640×400 at 15 FPS are adequate for a significant number of use cases.

The development of the mimicry framework described above raised several practical questions during implementation. Among these, a central one was the following:

"What is the most effective camera layout and data- strategy for extracting human keypoints using general-purpose sensors and AI-based pose-detection frameworks?"

To address this question, the gesture-analysis setup shown in Fig. 3.10 was implemented in Bajrami et al. (2025a). The experimental platform comprises:

- 12 cameras of an OptiTrack motion-capture system, used as the reference baseline;
- 3 RGB-D cameras (only the RGB stream is considered), arranged according to Tab. 3.7.

The RGB streams from the three cameras are processed to evaluate which spatial reconstruction strategy provides the most reliable 3-D keypoints. Specifically, three approaches are compared: (i) direct use of the pose-detection framework (here referred

to as *MediaPipe*); (ii) fusion of two or more camera-wise 3-D pose estimates obtained in (i) (*Fusion*); and (iii) direct multi-view reconstruction via calibrated geometric triangulation using two or more cameras (*Triangulation*).

The results show that there is no single optimal camera arrangement or reconstruction strategy. Indeed, depending on the spatial configuration of the keypoints, the relative performance of the considered methods can change substantially. For each gesture, the mean reconstruction error was evaluated. A summary of the results is reported in Tab. 3.7, where:

- Sh MA – Root Mean Square Error of the shoulder joint after moving average filtering;
- El MA – Root Mean Square Error of the elbow joint after moving average filtering;
- Wr MA – Root Mean Square Error of the wrist joint after moving average filtering;
- Mean MA – mean RMSE across shoulder, elbow, and wrist after moving average filtering;
- Mean SG – mean RMSE across shoulder, elbow, and wrist after Savitzky–Golay filtering.

Overall, the best-performing method is gesture-dependent. For the Drinking gesture, the multi-view Fusion approach using all three cameras (F123) provides the lowest mean error. In Eating, the monocular MediaPipe configuration (MP3) achieves the best results, indicating higher robustness in that motion phase. Conversely, for Zip Close and Touching Nose, three-view Triangulation (T123) yields the lowest RMSE, consistent with the fact that these gestures involve fewer occlusions and more stable multi-view detections. These considerations lead to the development of the strategy

Table 3.7 Best-performing configuration per gesture (RMSE in mm) Bajrami et al. (2025a)

Gesture	**Method**	**Setup**	w	(n,m)	**Sh MA**	**El MA**	**Wr MA**	**Mean MA**	**Mean SG**
Drinking	Fusion	F123	11	(2,11)	34.9	75.1	99.2	69.7	71.3
Eating	MediaPipe	MP3	15	(2,11)	27.1	68.2	98.1	64.5	66.2
Zip Close	Triangulation	T123	5	(3,9)	22.0	84.8	52.0	52.9	53.5
Touching Nose	Triangulation	T123	15	(3,11)	37.2	129.5	64.8	77.1	77.5

described in the following paragraph, which is also implemented in the RL + LfD pipeline discussed in Chapters 4 and 5.

These results also clarify an important methodological point: independent 2-D detections from different cameras are affected by camera-specific optics, viewpoint changes, and pose-estimation uncertainty, and therefore should not be expected to coincide perfectly at the pixel or landmark level. The role of calibration and triangulation is precisely to compensate for these differences and recover a geometrically consistent 3-D estimate in a shared reference frame.

The reported RMSE values indicate that the proposed pipeline is not intended as a high-precision biomechanical measurement system, but as a low-cost and sufficiently reliable method for generating task-level trajectories and waypoints for downstream robotic execution. In this sense, the objective is not millimetre-accurate anatomical reconstruction, but robust acquisition of demonstration references that can be reused in the subsequent robot-learning pipeline.

Detection switching strategy The comparative results indicate that no single low-cost markerless pipeline is consistently reliable across all gestures and motion phases. Multi-view triangulation provides the highest geometric accuracy when all joints are clearly visible in all views, whereas fusion-based and carefully placed single-view configurations tend to be more robust under self-occlusions and intermittent missing detections. This motivates an online switching strategy based on a *detection manager* that dynamically selects, frame by frame, the most reliable reconstruction method according to geometric consistency, detector confidence, and kinematic/temporal plausibility.

A high-level overview of the proposed detection manager is shown in Algorithm 1. Multi-view keypoints and confidences are processed by a set of candidate estimators (single-view, multi-view fusion, and triangulation). Their 3-D outputs are evaluated through a scoring module, and the selected method is stabilized via hysteresis. A final guard-rail and smoothing stage enforces conservative fallbacks (e.g., hold-last or single-view estimation) when the estimated quality is low.

Filtering also plays a central role in improving gesture-tracking reliability. Across gestures, camera combinations, and pipelines, results in Bajrami et al. (2025b) show that a simple moving-average filter with window lengths between 5 and 15 frames systematically matches or outperforms Savitzky–Golay in terms of RMSE. This

Algorithm 1 Online detection manager with method switching, hysteresis, and guard-rail smoothing

Require: Multi-view keypoints $\{K_t^{(v)}\}_v$, confidences $\{c_t^{(v)}\}_v$, camera params, previous pose X_{t-1}, previous method m_{t-1}
Ensure: Final pose X_t, metadata $\mathscr{M}_t$

$\mathscr{S} \leftarrow \{\text{Triang, MV-Fusion, SV}\}$ ▷ Candidate estimators
$\rho_t \leftarrow \textsc{MissingRatio}(\{c_t^{(v)}\}_v)$
if $\rho_t > \tau_{\text{miss}}$ **then**
  $X_t \leftarrow X_{t-1}$ ▷ Hold-last state
  return $(X_t, \mathscr{M}_t \leftarrow \{\text{fallback} = 1, \text{reason} = \text{'missing'}\})$
end if
for all $m \in \mathscr{S}$ **do**
  $\hat{X}_t^{(m)} \leftarrow \textsc{Estimate3D}(m, \{K_t^{(v)}, c_t^{(v)}\}_v, \text{cameras}, X_{t-1})$
  $J_t^{(m)} \leftarrow w_g J_g(\hat{X}_t^{(m)}) + w_d J_d(\{c_t^{(v)}\}_v) + w_k J_k(\hat{X}_t^{(m)}) + w_\tau J_\tau(\hat{X}_t^{(m)}, X_{t-1})$
end for
$m_t^\star \leftarrow \arg\min_{m \in \mathscr{S}} J_t^{(m)}$ ▷ Select best by score
$m_t \leftarrow \textsc{HysteresisSelect}(m_{t-1}, m_t^\star, J_t, \Delta_{\text{hyst}})$
$\hat{X}_t \leftarrow \hat{X}_t^{(m_t)}$
if $J_t^{(m_t)} > \tau_{\text{bad}}$ **then**
  $\hat{X}_t \leftarrow \textsc{Fallback}(\hat{X}_t, X_{t-1}, \hat{X}_t^{(\text{SV})})$
  $f_t \leftarrow 1$
else
  $f_t \leftarrow 0$
end if
$X_t \leftarrow \textsc{MovingAverage}(\hat{X}_t, W)$ ▷ $W \in [5, 15]$ frames
$\mathscr{M}_t \leftarrow \{m_t, J_t^{(m_t)}, f_t, \rho_t\}$
return $(X_t, \mathscr{M}_t)$

suggests that the dominant error behaves mainly as high-frequency noise, and that low-order temporal smoothing is sufficient to stabilize trajectories while keeping latency compatible with real-time mimicry.

3.7.1 Takeaways for RL and LfD

The analysis highlights several key considerations for the use of markerless motion capture in RL and LfD frameworks.

First, no single sensing or reconstruction strategy is sufficient across all tasks and motion phases. An adaptive selection mechanism that switches between monocular,

fusion-based, and triangulation-based estimators improves reliability and reduces failure cases during demonstration acquisition.

Second, simple temporal smoothing, such as a moving-average filter with limited window size, is effective in stabilizing trajectories while preserving responsiveness, making it suitable for online demonstration and real-time imitation.

Finally, the proposed detection switching strategy enables the use of low-cost cameras and open-source pose estimation frameworks to acquire consistent demonstrations, which can be directly leveraged to initialize policies, constrain exploration, or provide reference trajectories in RL and LfD pipelines.

Chapter 4

Reinforcement Learning-Based Training and Analysis Pipeline for Contact-Rich Tasks

"If there were no suffering, man
would not know his limitations,
would not know himself."

Leo Tolstoy, *War and Peace* (1869)

The activities presented in this chapter were carried out during a seven-month research period abroad as part of the author's doctoral programme, hosted by the VTT Technical Research Centre of Finland, Oulu. The author acknowledges the supervision and support of Dr. Markku Suomalainen and Prof. Tapio Heikkilä throughout this period. VTT provided access to the experimental instrumentation (including force-measurement and motion-tracking systems), the computing infrastructure required for development and testing, and the material resources necessary to conduct the experimental activities. This technical and logistical support was essential for the development and validation of the contact-rich polishing framework described in this chapter.

4.1 Introduction and Motivation

Contact-rich tasks, such as polishing, grinding, and surface finishing, are central to many industrial workflows (Hamdan et al. (2024); Wu et al. (2025)) but they remain difficult to automate in a way that is both flexible and safe. This is due to the fact that skilled human operators are able to continuously adapt their movements and the forces they apply in response to factors such as material properties, surface geometry, and quality requirements. It is noteworthy that contemporary robots continue to experience difficulties in emulating these adaptive capabilities.

A simple illustrative example is shown in Fig. 4.1, which reports the contact forces measured along the x, y, and z axes (force sensor reference frame) during the manual polishing of an aluminium bar component. The data were obtained in a real experiment using a force sensor integrated into the tool and an NDI camera (NDI (2024)) for pose tracking. As shown in the figure, force profiles exhibit substantial oscillations and high-frequency components. An experienced operator can learn to cope with these vibrations and maintain acceptable surface quality, but this requires time, task-specific expertise, and careful tuning of motion and force-factors that may not generalise well across different tools, materials, or surface geometries.

In the context of Industry 4.0 and human-robot interaction (HRI), there is strong interest in robotic systems that can both learn from human expertise and autonomously optimise their behaviour in contact, while respecting strict safety and stability constraints (Argall et al. (2009); Bajrami et al. (2024); Elguea-Aguinaco et al. (2023); Hogan (1985); Keemink et al. (2018)). Ideally, such systems should allow humans to specify *what* task to perform (e.g., through demonstration or high-level programming) while enabling the robot to refine *how* the task is executed in terms of interaction forces, compliance, and trajectory (Argall et al. (2009); Elguea-Aguinaco et al. (2023); Hogan (1985)).

The use of RL, and more recently deep reinforcement learning (DRL), provide a data-driven approach for synthesising adaptive behaviours in robotic systems. While RL refers broadly to algorithms that learn through interaction with an environment, DRL leverages deep neural networks to represent policies or value functions, enabling learning in high-dimensional, continuous spaces. Instead of explicitly programming how a robot should modulate its stiffness, damping, or contact forces, DRL treats the controller as a parametric policy that is iteratively optimised via trial-and-error in

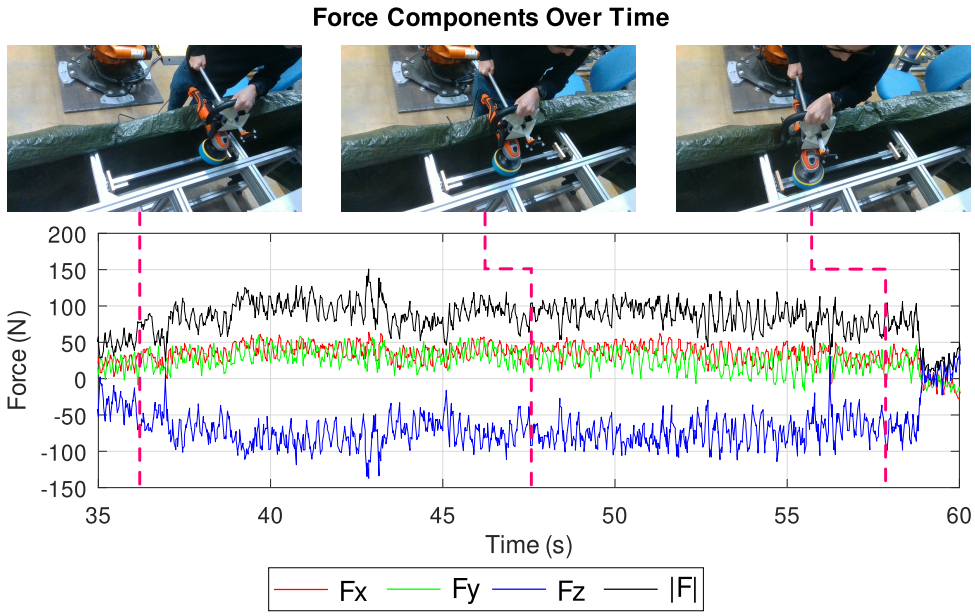


Figure 4.1 Contact-force components F_x, F_y, and F_z (force-sensor reference frame) measured during manual polishing of an aluminium bar, highlighting the oscillatory nature of the interaction. The data were acquired using force-measurement instrumentation available at the VTT Technical Research Centre of Finland, Oulu.

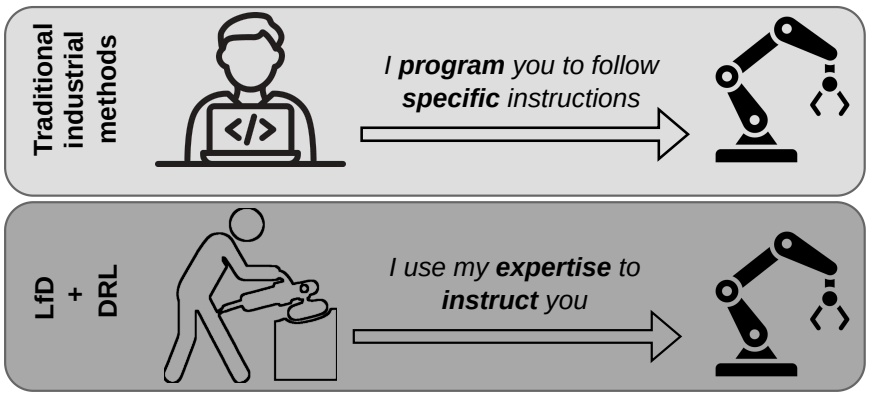


Figure 4.2 Traditional programming vs learning-based approaches for specifying *what* task to perform and learning *how* to execute it.

simulation. Since effective behaviour emerges only after extensive exploration, this process typically requires the robot to perform a large number of trial episodes.

Modern physics-based simulators and "digital twins" enable this process to occur in silico, reducing the risk of hardware damage and supporting large-scale experimentation with control strategies that would be costly or impractical to test directly on

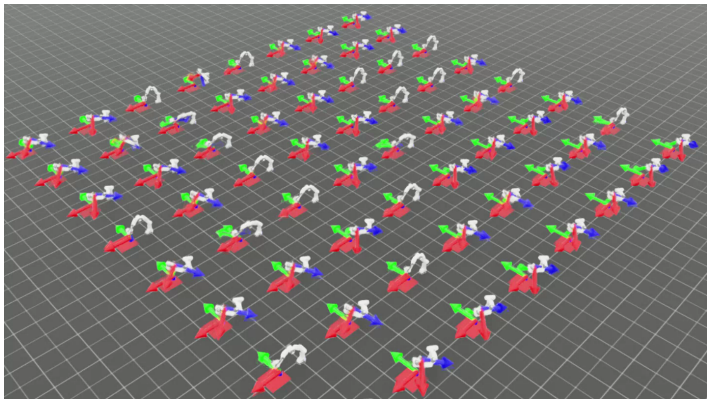


Figure 4.3 Example of multi-environment training setup in NVIDIA Isaac Lab/Sim

physical robots. For example, Fig. 4.3 illustrates multiple polishing environments running in parallel on the GPU, each with slightly different surface and robot parameters. This setup allows policies to be trained under diverse conditions, with the training results from each environment shared through a common learning buffer.

Continuous-contact control and, more broadly, contact-rich manipulation (e.g., polishing/surface finishing) are well-established topics in the literature; however, applying RL to these scenarios remains non-trivial for practical reasons. Exploration typically requires large amounts of experience and, on real hardware, entails non-negligible risk and experimental cost. In this context, modern physics simulators and digital twins enable a significant portion of exploration to be conducted in silico, reducing the likelihood of damage and making the development-and-validation loop more systematic.

That said, "generic" RL benchmarks rarely encode the failure modes that dominate sliding contact and machining-like operations: (i) force regulation and stability at the tool–surface interface; (ii) the coupling between Cartesian motion and contact dynamics; and (iii) sufficiently rich instrumentation to interpret why a policy converges or fails, beyond the final outcome alone. At the time of this work, the thesis identifies a specific gap: within NVIDIA Isaac Lab, there was no publicly reusable framework focused on these tasks, nor tools to systematically log and analyse contact behaviour under variable-impedance control.

Accordingly, the contribution is framed as the implementation of an experimental pipeline and software artefacts aimed at robustness and interpretability: two open-source frameworks (Residual RL + OSC and RL + LfD) and a modular architecture

that separates the environment, variable-impedance control, trajectory management, sensing, reward design, and a dedicated debug/logging subsystem. This structure is intended to support quantitative comparisons against control baselines and post-hoc analysis based on structured logs, so that the contact dynamics driving success or failure become observable.

Continuity with the TIAGo LfD Framework The approach in Sec. 4.5 builds directly on the trajectory acquisition methodology and observational design introduced in Chapter 3. Here, those procedures are implemented in the second RL framework: demonstrations provide reference trajectories, and the agent adapts control parameters to improve the robustness of trajectory tracking.

Polishing as an Industry 4.0 Use Case Polishing tasks represent a relevant use case within the Industry 4.0 landscape, where automation is challenged by small-batch variability, strict quality requirements, and the need for safe human-robot collaboration. The simulation-based pipeline introduced in this chapter serves as a proof of concept for contact-rich manipulation, offering a modular environment where different control strategies, task configurations, and interaction conditions can be systematically explored. Rather than aiming for immediate industrial deployment, the pipeline is designed as a digital lab that supports reproducible experimentation and lays the groundwork for the development of human-aware robotic systems. The results of this exploration are presented in Chapter 6.

Task Description: Manual Polishing Task Figure 4.1 illustrates a sequence of frames from a manual polishing task, in which an operator applies alternating strokes over an aluminium bar rigidly clamped to a table. The motion consists of a series of repeated bidirectional movements, typically starting from a random initial pose, approaching the surface until contact is established, and then letting the tool traverse laterally under the resistance of the surface. This path is reversed at the end of the stroke, creating a cyclic pattern repeated multiple times along a single line.

Pose/Force tracking and transition Shifting from free-space motion tracking to in-contact force regulation is a classical problem in robot interaction control, historically framed by hybrid position/force schemes that explicitly split task directions between

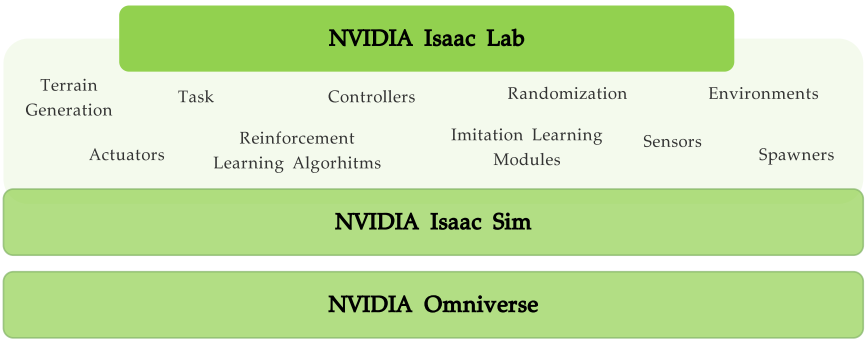


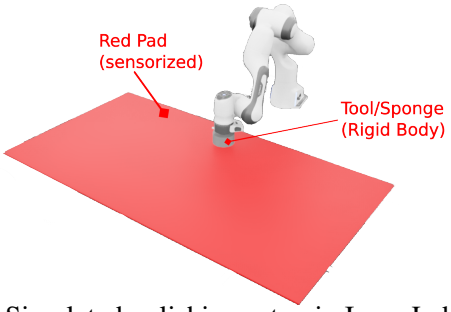
Figure 4.4 Conceptual position of NVIDIA Isaac Lab within the Isaac Sim / Omniverse stack.

motion and force objectives (Raibert and Craig (1981)). A key difficulty is deciding *when* and *how* to switch (or blend) the control objectives: even small static position errors accumulated during free-space tracking can yield undesirable transients at the instant force feedback is engaged, often resulting in an impact on the environment (Brun et al. (2003)). Beyond explicit hybrid switching, alternative families include impedance/force-tracking impedance control, which regulates interaction by shaping the closed-loop mechanical impedance Hogan (1985), and operational-space formulations that unify motion and force objectives at the task level (Khatib (1987)). Overall, the problem admits multiple solution families with different trade-offs (explicit hybrid switching, impedance-based interaction control, and operational-space methods). In this work, for the first framework is usde OSC (Khatib (1987)).

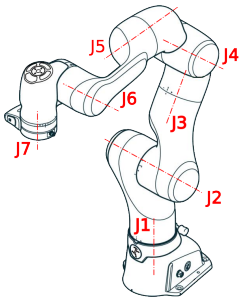
This chapter first introduces the hardware and simulated setup together with the software stack. The framework is then described in two configurations: (i) a *Procedural Trajectory Mode*, based on phase-driven waypoints and OSC execution with fixed or adaptive impedance; and (ii) a *Trajectory by Learning from Demonstration* mode, where demonstrated waypoint trajectories define the task reference and a constrained RL formulation improves reproduction robustness. Observation and action spaces, episode logic, termination criteria, and reward design are detailed for both modes, followed by a roadmap toward contact-aware reproduction.

4.2 Framework Overview and Software Stack

The proposed framework is instantiated on a polishing task with a Franka Emika Panda robot (7-DoF serial manipulator) operating in simulation using NVIDIA Isaac



Simulated polishing setup in Isaac Lab (cylindrical tool and soft red pad).



Franka Emika Panda joint layout (J1–J7).

	J1 (A1)	J2 (A2)	J3 (A3)	J4 (A4)	J5 (A5)	J6 (A6)	J7 (A7)
Range [deg]	$[-166, 166]$	$[-105, 105]$	$[-166, 166]$	$[-176, -7]$	$[-165, 165]$	$[25, 265]$	$[-175, 175]$

Joint position limits from the Franka Research 3 datasheet.

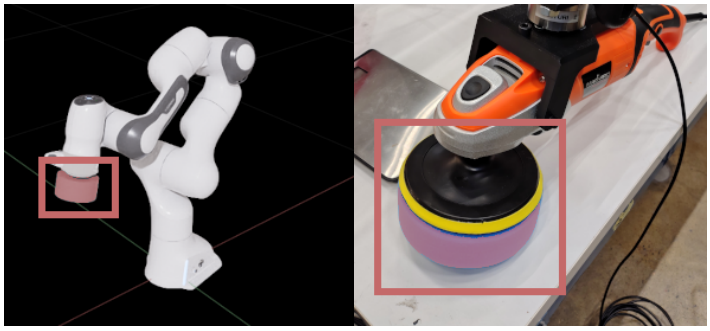
Figure 4.5 Franka Emika Panda robot and simulated polishing setup used in the framework.

Lab/Isaac Sim. As shown in Fig. 4.4, Isaac Lab is a high-level robot learning interface built on top of NVIDIA Isaac Sim and Omniverse (Mittal et al. (2023)). It provides GPU-accelerated physics simulation, robot models, and APIs for defining tasks and connecting them to learning algorithms. In this work, Isaac Lab serves as the entry point for implementing the polishing environment, managing assets (robot, tool, surface), and integrating with reinforcement learning libraries.

Figure 4.5 provides an overview of the simulated polishing setup and the joint layout of the Franka Emika Panda robot used in this work. Since the primary focus is on interaction control rather than detailed cell layout, the simulated scene is intentionally kept minimal. Non-essential elements, such as a full table model or surrounding fixtures, are omitted. The robot is equipped with a simplified polishing tool, modelled as a rigid cylinder that approximates a sponge-based device. The right side of the figure shows the joint configuration, while the lower part reports the joint position limits, as specified in the Franka Research 3 datasheet (Franka Robotics GmbH (2025)). These bounds are used to define safe initial conditions and trajectories in simulation, and to constrain both the operational-space controller and the RL actions to physically plausible ranges.

In the polishing environment, the tool is modelled as a cylindrical mesh that approximates the geometry and mass distribution of the real sponge. Among the

available body types and physics models in Isaac Sim, summarised in Table 4.1, the mesh representation was selected for its improved numerical stability in contact-rich scenarios. The Fig. 4.6 provides a visual comparison between the simulated and real sponge, along with the physical parameters used in simulation.



	Radius r	Height h	Mass m	Density ρ	Geometry	Material
Value	0.10 m	0.05 m	0.05 kg	31.8 kg/m^3	Mesh	Foam-like

Figure 4.6 Simulated (left) and real (right) sponge used in polishing tasks. The table reports the reference physical properties adopted in the simulated environment, based on measurements from the real sponge.

Table 4.1 Summary of geometry types and physics material models available in Isaac Sim.

Concept	Sub-category / Usage	Key Parameters	Main Effect
Mesh	Visual and optional collision mesh	Vertex data, triangle indices, normals, UVs; collision modes (triangle mesh, convex hull, SDF)	High-fidelity visual geometry; can be used for collision, with higher computational cost.
Shape	Primitive collision shape	Size parameters (box extents, radius, height, etc.)	Lightweight proxy geometry for fast and robust contact handling.
Physics Material	Rigid body (USD)	staticFriction, dynamicFriction, restitution, density	Defines surface friction, bounciness, and mass (from geometry volume).
	Rigid body (PhysX extended)	All above + compliantContactStiffness, compliantContactDamping	Enables soft and compliant contact dynamics, suitable for force-sensitive tasks.
	Deformable (FEM)	youngsModulus, poissonsRatio, damping, density	Governs elasticity, volume conservation, damping, and mass for deformable bodies.
	PBD (particles / fluids / cloth)	friction, particleFrictionScale, damping, viscosity, surfaceTension, cohesion, adhesion	Controls particle interactions and fluid-like behaviour (thickness, cohesion).
	PBD contact tuning	particleContactOffset, restOffset, fluidRestOffset	Adjusts effective contact distance and overlap thresholds in particle-based models.

Isaac Lab Environment and Control Architecture In Isaac Lab, RL environments are implemented on top of the base classes *ManagerBasedRLEnv* and *DirectRLEnv*, which realize the *gymnasium.Env* interface. In the present work, the direct workflow is adopted.

Control can be performed using Isaac Lab's Operational Space Controller (OSC), by directly applying joint torques, or through a combination of both approaches. The OSC is encapsulated within a custom *OSCWrapper* module, which computes joint torques to track desired end-effector trajectories and enforce specified interaction properties in operational space. Impedance gains (stiffness and damping) can either be fixed in the task configuration or exposed as variables to be modulated by the RL policy.

For policy training, *PolishEnv* is integrated with external libraries such as RL-Games via dedicated wrappers from the *isaaclab_rl* package, which adapt the Gymnasium-compatible environment to the vectorized interface required by the framework. A single *PolishEnv* instance internally manages multiple parallel sub-environments, configured via the *num_envs* parameter, enabling GPU-based execution and the collection of experience across a diverse set of polishing conditions at each training iteration.

4.3 Framework Architecture

This section outlines the internal organisation of the polishing framework. The architecture is layered to separate environment logic, control, trajectory generation, sensing, reward computation, and logging, enabling modular extensions and controlled comparisons between different control strategies.

4.3.1 Layered Control and Learning Architecture

At a high level, the framework combines: an environment layer that encapsulates simulator dynamics and task logic; a control layer that can be realised either as an OSC with impedance parameters or as a task-space IK-based controller with discrete action commands; a trajectory and phase management layer that defines nominal end-effector motion either in procedural trajectory mode or from LfD; a contact sensing and reward layer that turns interaction signals and task state into learning feedback;

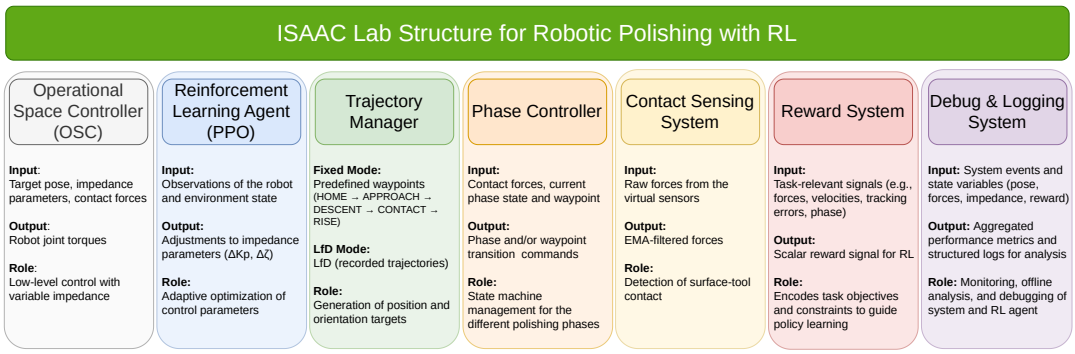


Figure 4.7 Block diagram of the Isaac Lab architecture for robotic polishing with reinforcement learning, showing the main modules for control, sensing, reward computation, and logging.

and a debug/logging subsystem for systematic analysis. These elements correspond to the blocks shown in Fig. 4.7 and are described below.

Environment Layer: `PolishEnv` The environment layer encapsulates the simulation and task logic behind a standard RL interface. The `PolishEnv` class inherits from Isaac Lab's `DirectRLEnv` and implements the polishing task as a Markov decision process. It initialises the simulated scene, configures the robot and workpiece, applies domain randomisation, and manages per-step interaction with the RL library, including action decoding, safety checks, internal task-state updates, and construction of observations and termination signals. From the perspective of an external RL algorithm, `PolishEnv` exposes the standard `reset()` and `step()` methods with `Gymnasium`–compatible observation and action spaces.

Control Layer: Impedance or Discrete Task-Space Control The control layer translates high-level commands into joint torques. In the OSC-based configuration, a custom `OSCWrapper` maps desired end-effector motion and wrench into joint torques under a specified impedance behaviour; impedance gains can be fixed or modulated by the policy. In the discrete LfD configuration, the agent outputs task-space discrete actions (e.g., $\Delta x, \Delta y, \Delta z$), which are converted into motion targets and executed through IK. Both variants rely on the same environment interface and can be compared under consistent simulation settings.

Trajectory and Phase Management Nominal end-effector motion is produced by a trajectory and phase management layer. In procedural trajectory mode, the layer follows hand-crafted sequences of waypoints encoding standard motion patterns structured into phases such as approach, contact, and retreat. In LfD mode, trajectories are recorded from demonstrations and replayed during simulation, possibly with randomised initial conditions. A phase controller acts as a state machine that switches phases and trajectory segments based on tracking errors, contact events, and timeout or stuck conditions. Phase and trajectory progression are treated as part of the environment state and can influence observation design and reward shaping.

Reinforcement Learning Agent The RL agent runs outside the simulator and interacts with `PolishEnv` through the `Gymnasium` API, making the framework agnostic to the specific algorithm. In the experiments presented in this thesis, the agent is instantiated using RL-Games with a PPO backbone. At each control step, the agent observes a vector encoding robot state, task-phase information, and (when relevant) contact and impedance cues, and outputs either impedance adjustments or discrete task-space actions depending on the configuration. Policy parameters are updated periodically from batches of collected trajectories and rewards.

Contact Sensing and Reward System Contact sensing reads raw forces from virtual sensors and applies filtering and frame transformations. When contact information is central to the task, signals such as normal force and force rate are used to detect contact onset/loss and to shape rewards. The reward system combines these processed signals with kinematic and dynamic quantities to compute the scalar feedback passed to the RL agent. Reward definitions are modular and can be swapped depending on the objective (e.g., contact stability, force tracking, or waypoint progression).

Debug and Logging Subsystem A central logger records time series of relevant variables such as joint states, end-effector poses, contact forces, impedance parameters (when applicable), phase indicators, and reward components. These traces are stored in structured files for offline analysis and comparison across controller settings or reward designs. Optional on-line debug facilities, such as console messages or visual markers, can be enabled during development and disabled in large-scale training.

Repository Variants (Summary)

The layered architecture is shared across two public codebases, but the control formulation and trajectory source differ.

`VTT_RL_fixed_public`**.** This variant implements **Pocedural Trajectory Mode** with a phase-based waypoint manager (e.g., approach, contact, retreat). Control is OSC-based, with impedance parameters exposed to the policy for online modulation (typically along selected task-space axes). Contact sensing signals (e.g., filtered normal force and force rate) are used both for phase logic and for reward shaping, which includes terms related to contact establishment and force stability. Logging focuses on impedance evolution, contact forces, phase transitions, and reward components.

`VTT_RL_public`**.** This variant focuses on **Trajectory by Learning from Demonstration** with a discrete task-space action set (i.e., $\Delta x, \Delta y, \Delta z$ combinations) executed through IK. The reward is based on sequential axis tracking (X$\rightarrow$Y$\rightarrow$Z) and waypoint progression along demonstrated trajectories. The main emphasis is trajectory reproduction and generalisation under a discrete action formulation, with a lighter reliance on contact-specific sensing.

4.4 Procedural Trajectory Mode

This section describes the Procedural Trajectory Mode (`VTT_RL_fixed_public`) of the polishing framework . The focus is on how the OSC layer is configured and executed in code, how impedance parameters are represented and optionally modulated by the policy, and how an optional joint-torque layer can be enabled. The goal is to document the implementation as it exists in the current codebase. The section is organised by first detailing the controller formulation and command structure, then the contact-handling logic and masking strategy, and finally the interfaces exposed to the learning pipeline (actions, observations, and termination). Chapter 5 describes an example of how this software is used and the results obtained.

4.4.1 OSC Formulation and Command Structure

The control architecture adopted in this work follows the classical operational-space formulation introduced by Khatib Khatib (1987) and subsequently systematized in

modern robot dynamics references (e.g., the ETH *Robot Dynamics* lecture notes Hutter (2017)).

For a fixed-base manipulator with joint coordinates $q \in \mathbb{R}^n$, the joint-space dynamics are

$$M(q)\,\ddot{q} + b(q,\dot{q}) + g(q) = \tau, \tag{4.1}$$

where $M(q)$ is the inertia matrix, $b(q,\dot{q})$ collects Coriolis and centrifugal terms, $g(q)$ denotes the gravity torques, and $\tau \in \mathbb{R}^n$ is the vector of joint torques. Let $w_e = [v_e^\top \;\; \omega_e^\top]^\top \in \mathbb{R}^6$ denote the spatial velocity (twist) of the end-effector tool frame, and let $J_e(q) \in \mathbb{R}^{6\times n}$ be its geometric Jacobian such that $w_e = J_e(q)\,\dot{q}$.

Assuming a rigid-body fixed-base model and smooth task kinematics, and further assuming that $J_e(q)$ is full row rank in the task directions of interest (i.e., away from kinematic singularities), the end-effector dynamics can be expressed in operational coordinates as

$$\Lambda_e(q)\,\dot{w}_e + \mu_e(q,\dot{q}) + p_e(q) = F_e, \tag{4.2}$$

where $\Lambda_e \in \mathbb{R}^{6\times 6}$ is the operational-space inertia, μ_e collects Coriolis/centrifugal effects in operational coordinates, p_e is the operational-space gravity term, and $F_e \in \mathbb{R}^6$ is the operational wrench. In the nonsingular case, Λ_e is given by the standard expression

$$\Lambda_e(q) = \left(J_e(q)\,M^{-1}(q)\,J_e^\top(q)\right)^{-1}. \tag{4.3}$$

The operational wrench is mapped to joint torques through the Jacobian transpose,

$$\tau = J_e^\top(q)\,F_e, \tag{4.4}$$

which follows from power consistency and provides the basic relationship used by OSC to realize a desired end-effector wrench.

Command structure in the proposed software stack. At each control step, the environment assembles a command vector that concatenates a desired end-effector pose, a target operational wrench, and (optionally) impedance parameters. The command format depends on the selected impedance mode:

- `fixed`: $[x_e^d,\, F_e^d]$,
- `variable_kp`: $[x_e^d,\, F_e^d,\, K_p]$,

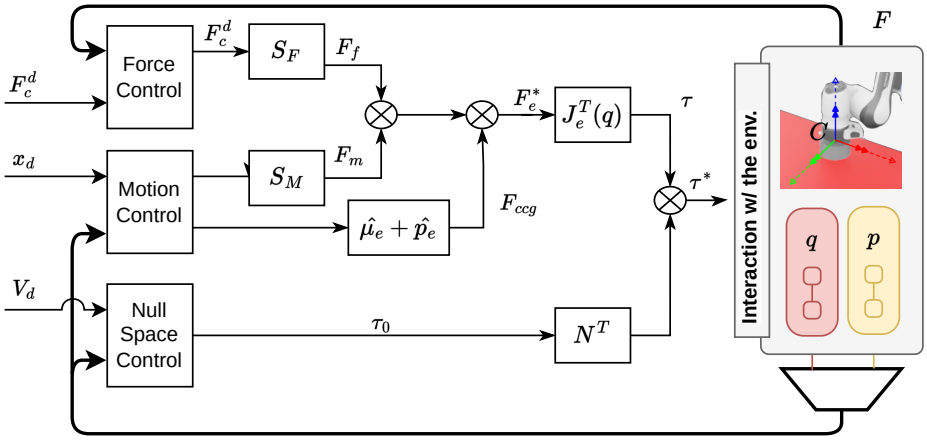


Figure 4.8 Classical OSC block diagram

- `variable`: $[x_e^d,\, F_e^d,\, K_p,\, \zeta\,]$.

Here $x_e^d \in \mathbb{R}^7$ is the target pose (position + quaternion), $F_e^d \in \mathbb{R}^6$ is the target wrench, and $K_p, \zeta \in \mathbb{R}^6$ are diagonal task-space stiffness and damping-ratio parameters. The OSC computes joint torques from the current robot state and model quantities (in particular $J_e(q)$ and $M(q)$). The resulting torques can be applied directly or augmented by an optional joint-torque action layer. Impedance parameters are clamped to configured limits before being passed to the controller.

Motion/force specification matrices (masking matrices) The tasks considered in this study require the robot to transition from a free-motion configuration (i.e., in the absence of external contact forces) to a contact configuration in which interaction with the workpiece is established. The entire trajectory, from approach to sustained contact, is considered. Masking matrices are therefore introduced to formalize the transition from position-based control to force-based control. A Cartesian frame $\{C\}$ is attached to the contact point, with z_c aligned with the surface normal and (x_c, y_c) spanning the tangential plane. The tangential directions are regulated in motion (position/velocity), whereas the normal direction is primarily regulated in force.

Following Hutter (2017); Khatib (1987), binary specification matrices are introduced for translational and rotational motion control:

$$\Sigma_p = \mathrm{diag}(\sigma_{px}, \sigma_{py}, \sigma_{pz}), \qquad \Sigma_r = \mathrm{diag}(\sigma_{rx}, \sigma_{ry}, \sigma_{rz}), \tag{4.5}$$

where each $\sigma_i \in \{0,1\}$ selects whether the corresponding degree of freedom is regulated in motion (1) or assigned to force control/constraint reaction (0). Let $C \in SO(3)$ denote the rotation from the world frame to the contact frame $\{C\}$. The corresponding motion and force selection matrices in $\mathbb{R}^{6\times 6}$ are then defined as

$$S_M = \begin{bmatrix} C^\top \Sigma_p C & 0 \\ 0 & C^\top \Sigma_r C \end{bmatrix}, \qquad S_F = \begin{bmatrix} C^\top (I-\Sigma_p) C & 0 \\ 0 & C^\top (I-\Sigma_r) C \end{bmatrix}, \tag{4.6}$$

so that, for binary Σ_p and Σ_r, it holds that $S_M + S_F = I_6$. In the polishing case, a typical choice is $\sigma_{pz} = 0$ (force regulation along the surface normal) and $\sigma_{px} = \sigma_{py} = 1$ (motion regulation in the tangential plane), with analogous choices for rotational degrees of freedom.

Commanded operational wrench The commanded operational wrench is decomposed into motion, force, and compensation terms:

$$F_e^\star = F_m + F_f + F_{ccg}, \tag{4.7}$$

where F_m specifies the desired motion behavior, F_f specifies the desired contact wrench, and F_{ccg} compensates for gravity, Coriolis, and centrifugal effects. A task-space impedance law is adopted for F_m, and it is combined with a reference contact wrench through F_f:

$$F_m = S_M \Big(\Lambda_d \, (\dot{w}_e^{\mathrm{d}} - \dot{w}_e) + K_p (x_e^{\mathrm{d}} - x_e) + K_d (\dot{x}_e^{\mathrm{d}} - \dot{x}_e) \Big), \tag{4.8}$$

$$F_f = S_F \, F_c^{\mathrm{d}}, \tag{4.9}$$

where x_e and $\dot{x}_e$ denote the end-effector pose (expressed in an appropriate minimal representation) and its time derivative, respectively, $(\cdot)^{\mathrm{d}}$ denotes desired quantities provided by the trajectory layer, Λ_d is a diagonal matrix of desired apparent masses, and K_p and K_d are positive definite stiffness and damping matrices defined in operational space.

The structure in Fig. 4.8 can be interpreted as the composition of three functional branches:

- **Motion-control branch:** evaluates the operational-space impedance law in (4.8) using the reference signals $(x_e^{\mathrm{d}}, \dot{x}_e^{\mathrm{d}}, \dot{w}_e^{\mathrm{d}})$ and the measured state $(x_e, \dot{x}_e, \dot{w}_e)$, and

applies the motion selection matrix S_M so that only motion-regulated axes contribute to F_m.

- **Force-control branch:** generates the desired contact wrench F_c^{d} in the contact frame (e.g., a prescribed normal polishing force) and applies the complementary selection matrix S_F to obtain F_f in (4.9), which is active only along force-regulated directions.
- **Null-space branch (optional):** synthesizes a joint-torque command τ_0 for secondary objectives (e.g., posture regulation, joint-limit/obstacle avoidance) and projects it through the dynamically consistent null-space projector, yielding the contribution $N^\top(q)\tau_0$ in (4.12).

Compensation terms The compensation component follows the standard operational-space inverse-dynamics structure:

$$F_{ccg} = \hat{\mu}_e(q,\dot{q}) + \hat{p}_e(q), \tag{4.10}$$

where $\hat{\mu}_e$ and $\hat{p}_e$ denote, respectively, estimated centrifugal/Coriolis and gravity terms in (4.2).

Null-space control for redundant manipulators In the non-redundant case, the task torque is given by

$$\tau_{\text{task}} = J_e^\top(q)\, F_e^\star. \tag{4.11}$$

For redundant manipulators, null-space torques can be employed to enforce secondary objectives without affecting the primary task dynamics. Let τ_0 denote the secondary-task torque command and $N(q)$ the dynamically consistent null-space projector. The resulting joint-torque command is

$$\tau^\star = \tau_{\text{task}} + N^\top(q)\,\tau_0. \tag{4.12}$$

The projector is defined as

$$N(q) = I - J_e^\top(q)\, J_e^{\#\top}(q), \tag{4.13}$$

where $J_e^{\#}(q)$ is the dynamically consistent pseudoinverse:

$$J_e^{\#}(q) = M^{-1}(q) J_e^{\top}(q) \left(J_e(q) M^{-1}(q) J_e^{\top}(q) \right)^{-1}. \tag{4.14}$$

With this construction, the null-space contribution $N^{\top}(q)\tau_0$ does not generate an additional operational wrench, thereby preserving the primary operational-space dynamics in (4.2). In the polishing experiments considered here, a PD posture controller is adopted:

$$\tau_0 = K_{p0}\left(q_0^{\mathrm{d}} - q\right) - K_{v0}\,\dot{q}, \tag{4.15}$$

where q_0^{d} is a desired joint configuration and K_{p0} and K_{v0} are positive definite gain matrices. When null-space control is disabled, τ_0 is set to zero, yielding $\tau^{\star} = \tau_{\mathrm{task}}$.

Impedance-gain parametrization and learning configurations In the polishing experiments, K_p and K_d are selected as diagonal matrices in the contact frame, so that the six translational and rotational directions are decoupled. When the OSC *impedance layer* is enabled, the complete set of diagonal entries (three translational and three rotational stiffness and damping gains) is always included in the observation vector, irrespective of which parameters are exposed to learning. The action vector, instead, contains only the subset of gains designated as trainable; the remaining entries of K_p and K_d are kept fixed while remaining observable by the policy.

This design yields three families of configurations: (i) a *fixed-impedance* mode, in which all task-space gains are constant and the action space does not include impedance parameters; (ii) a *partially adaptive* mode, in which the RL policy modulates a subset of task-space axes (e.g., only the normal direction, or the normal direction plus selected tangential directions and/or rotations); and (iii) a *hybrid* mode, in which the policy outputs both joint-level torques and a subset of operational-space gains. These configurations are summarized in Fig. 4.9, which illustrates how the impedance and torque layers jointly determine the observation and action spaces exposed to the agent.

Implementation remarks In the experimental pipeline, Isaac Lab provides the Jacobian J_e, the joint-space mass matrix $M(q)$, and the associated task-space quantities, and exposes an interface through which the user specifies x_e^{d}, $\dot{x}_e^{\mathrm{d}}$, K_p, K_d, and, optionally, a desired contact-frame wrench. The trajectory and phase managers provide the desired pose x_e^{d} and the nominal wrench F_c^{d}, the RL policy outputs the selected trainable entries

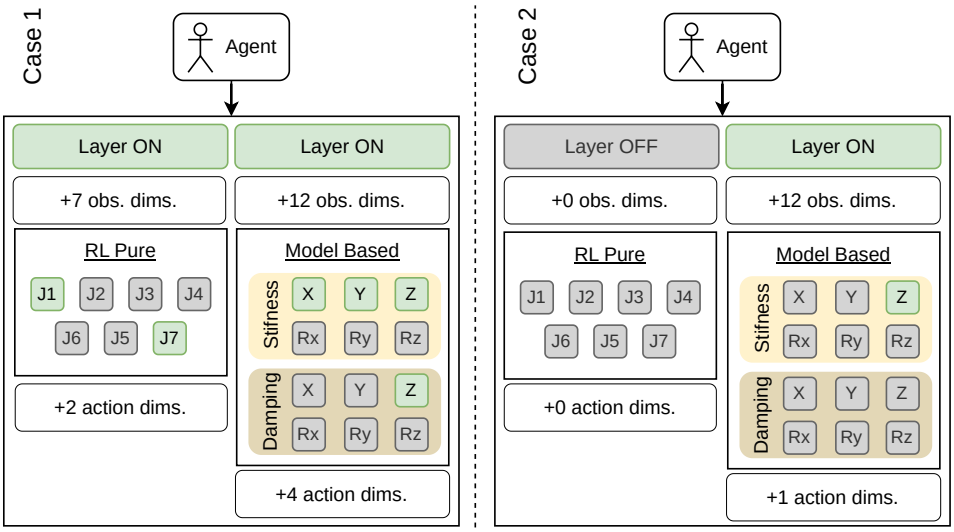


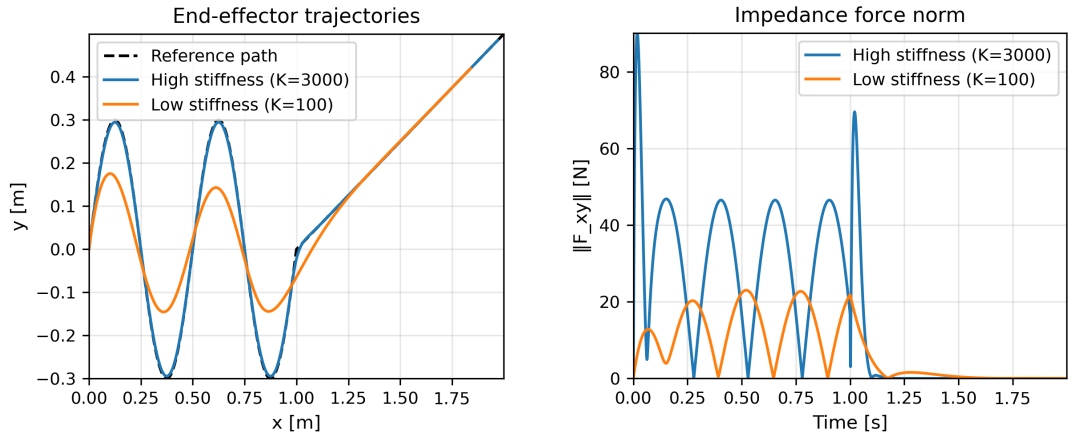
Figure 4.9 Example configurations of the OSC impedance layer and RL policy. When the impedance layer is enabled, all translational and rotational stiffness and damping gains are observed. In Case 1 the policy controls both joint torques and a subset of task-space gains, whereas in Case 2 only selected impedance gains are trainable and the torque layer is disabled.

of K_p and K_d, and the wrapper assembles the commanded operational wrench $F_e^\star$ according to (4.8)–(4.10). The resulting wrench is mapped to joint torques via (4.11), and, when enabled, augmented with the null-space contribution in (4.12).

K and D parameters tradeoff To illustrate the qualitative effect of task space stiffness on tangential motion and control effort, a simple planar impedance model was simulated in the (x,y) plane. The desired end-effector motion $(x_\mathrm{d}(t), y_\mathrm{d}(t))$ is a smooth reference path composed of a curvilinear segment followed by a straight line, parametrised over a finite time horizon of $T = 2$ s. The end effector is modelled as a point mass $m = 1$ kg subject to a virtual spring–damper in each tangential direction,

$$m\ddot{x} = K_x\big(x_\mathrm{d} - x\big) - D_x\dot{x}, \qquad m\ddot{y} = K_y\big(y_\mathrm{d} - y\big) - D_y\dot{y},$$

with damping coefficients chosen as $D_x = 2\sqrt{mK_x}\,\zeta$, $D_y = 2\sqrt{mK_y}\,\zeta$ and damping ratio $\zeta = 0.8$. Two impedance settings are compared: a *high-stiffness* case ($K_x = K_y = 3000$ N/m) and a *low-stiffness* case ($K_x = K_y = 100$ N/m), starting from the same initial state away from the reference path. The corresponding virtual spring



(a) End-effector trajectories for high and low stiffness ($K_x = K_y = 3k$ and $K_x = K_y = 0.1k$) [N/m].

(b) Norm of the planar impedance force $\|F_{xy}(t)\|$ for the same stiffness settings as in 4.10a.

Figure 4.10 Effect of task-space stiffness in a planar impedance model: (a) end-effector trajectory and (b) corresponding planar impedance force norm.

forces $F_x = K_x(x_\mathrm{d} - x)$, $F_y = K_y(y_\mathrm{d} - y)$ are combined into the planar force norm $\|F_{xy}(t)\| = \sqrt{F_x^2 + F_y^2}$.

The resulting trajectories are shown in Fig. 4.10a, while the corresponding force profiles are reported in Fig. 4.10b. In the tangential plane, higher stiffness leads to tighter adherence to the reference path, whereas lower stiffness produces a visibly softer, more compliant motion with larger deviations in (x, y). The force plot highlights the trade-off between trajectory accuracy and control effort: the high-stiffness controller achieves better tracking but at the cost of larger and more oscillatory impedance forces, while the low-stiffness controller yields smoother, lower-magnitude forces.

4.4.2 Reinforcement Learning Formulation and Training Setup

The polishing task is modelled as a Markov decision process (MDP) $(\mathscr{S}, \mathscr{A}, P, r, \gamma)$, where $\mathscr{S}$ is the state space, $\mathscr{A}$ the action space, P the transition kernel induced by the Isaac Lab simulation and the OSC controller, r the reward function, and $\gamma \in (0, 1]$ the discount factor. The internal simulator state $s_t \in \mathscr{S}$ includes the robot joint configuration, joint velocities, contact forces at the tool-surface interface, OSC internal variables, and phase/waypoint indices. The RL policy does not access s_t directly, but receives an observation vector $o_t = \phi(s_t)$ through the `PolishEnv` interface.

Observation Space

At each control step t, the environment builds a "one-step" observation vector $o_t^{\text{step}} \in \mathbb{R}^{n_{\text{step}}}$ by concatenating kinematic, contact, and controller-related quantities. In the procedural trajectory mode, the one-step observation has dimension $n_{\text{step}} = 27$ and is composed as summarised in Table 4.2.

Joint positions and velocities are taken directly from the simulated Panda model. The contact term uses the z component of the contact sensor net force. The controller-related part of the observation consists of the current task-space stiffness gains and damping ratios along the six translational and rotational directions. When damping observation is enabled (default in this codebase), the full set of ζ values is included even if only a subset is trainable.

To provide temporal context, the environment maintains a FIFO buffer of the last L one-step observations and flattens it into a single vector $o_t \in \mathbb{R}^{L n_{\text{step}}}$.

Action Space

The action vector $a_t \in \mathbb{R}^{n_{\text{act}}}$ encodes the control degrees of freedom exposed to learning. In this codebase, the action may include:

- incremental changes in selected task-space stiffness gains K_p;
- incremental changes in selected task-space damping ratios ζ;
- optional additive joint torques at the Panda joints.

Which components are active is determined by configuration flags: stiffness deltas, damping-ratio deltas, selected trainable axes, and optional joint-torque control. The

Table 4.2 Components of the one-step observation vector o_t^{step} used in the procedural trajectory mode.

Quantity	Symbol	Dim.
Joint positions of the Panda arm	$q \in \mathbb{R}^7$	7
Joint velocities of the Panda arm	$\dot{q} \in \mathbb{R}^7$	7
Normal contact force (sensor z component)	F_z	1
Task-space stiffness gains (transl./rot.)	$K_p = \text{diag}(K_x, K_y, K_z, K_{\text{Rx}}, K_{\text{Ry}}, K_{\text{Rz}})$	6
Task-space damping ratios (transl./rot.)	$\zeta = [\zeta_x, \zeta_y, \zeta_z, \zeta_{\text{Rx}}, \zeta_{\text{Ry}}, \zeta_{\text{Rz}}]$	6
Total per-step dimension	n_{step}	27

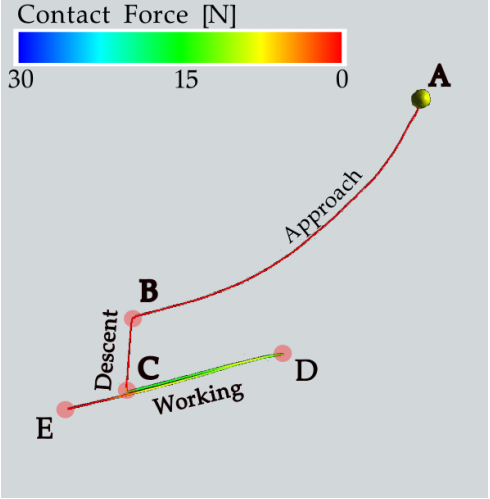


Figure 4.11 Example end-effector path for a polishing stroke in the simulated environment. The colour encodes the magnitude of the normal contact force along the path (blue: higher, red: lower); points A–E denote the waypoints described in the text, and the trajectory is taken from an episode executed with the OSC only.

resulting action dimension is

$$n_{\text{act}} = N_\tau + N_{\text{axes}}^{(K_p)} + N_{\text{axes}}^{(\zeta)},$$

where $N_\tau \in \{0, 7\}$ and $N_{\text{axes}}^{(K_p)}, N_{\text{axes}}^{(\zeta)}$ are either 0 or the number of selected task-space axes. In the typical procedural setup, only the normal axis is trainable, giving a 1D action (ΔK_z) or 2D action ($\Delta K_z, \Delta \zeta_z$) depending on whether damping deltas are enabled.

The possible configurations are schematically illustrated in Fig. 4.9, which highlights how the impedance layer and the torque layer affect the observation and action channels seen by the agent. In all cases the action components are scaled and clipped to lie in $[-1, 1]$ before being mapped to physical ranges for K_p, ζ, and joint torques inside the environment.

4.4.3 Episodes Description, Termination and Progression Conditions

Each training episode starts from a randomised initial configuration in the vicinity of a predefined HOME pose above the workpiece. A small lateral offset is sampled for

the nominal polishing trajectory in the x–y plane, and the phase controller is reset to its initial state. From this configuration, the end-effector is driven along a nominal Cartesian path that passes through five key waypoints A–E (Fig. 4.11); these are

A:HOME B:APPROACH C:CONTACT D:RIGHT_END E:LEFT_END

In the implementation, an additional intermediate waypoint is inserted between A and B to smooth the approach; it is not labelled in the figure. An illustrative example of the resulting end-effector path for a single polishing stroke in the simulated environment is shown in Fig. 4.11. The trajectory is taken from an episode executed with the OSC alone and is used solely to visualise the nominal stroke and the associated waypoints. The colour encodes the magnitude of the normal contact force $\|f_n\|$ along the path, with blue denoting higher forces and red lower ones.

Starting from A, the motion segments are

$$\mathrm{A} \xrightarrow{\textsc{Approach}} \mathrm{B} \xrightarrow{\textsc{Descent}} \mathrm{C} \xrightarrow{\textsc{Working}} \mathrm{D} \xrightarrow{\textsc{Working}} \mathrm{C} \xrightarrow{\textsc{Rise}} ,$$

where the last arrow corresponds to the lifting motion after returning to C.

Waypoint progression is triggered either when the XY position/orientation error falls below a tolerance or when a fixed dwell time is reached, whichever comes first.

Episodes terminate when one of the following conditions is met:

- *Time limit:* the maximum number of control steps per episode is reached.
- *Last waypoint reached:* the final waypoint is reached with position error and orientation below a predefined threshold.

4.4.4 Reward Design (Procedural Trajectory Mode)

The reward in the procedural trajectory mode is phase-structured and combines contact-related objectives with trajectory progression and regularisation terms. Its purpose is to guide a stable approach, achieve and maintain a target normal force, and complete the polishing stroke while discouraging aggressive parameter changes and excessive control effort. The same structure is used across reward variants, with specific terms enabled depending on whether only stiffness (K_z) is adapted or both stiffness and damping (K_z, ζ_z) are adapted. A compact set of formulations is provided in Appendix D.

Kz-Only Reward $\mathscr{R}_{K_z}$

The Kz-only configuration exposes a single action ΔK_z to the policy. The reward is phase-structured and combines descent shaping, contact-force regulation, and global regularisation terms. The main components are:

Descent shaping. During DESCENT, the reward encourages a fast but controlled approach and stable lateral positioning. It includes a descent-speed term, a time-pressure penalty for slow approach, an XY-stability term, a progress-to-surface term, a contact-establishment bonus, and a preparatory term that nudges K_z toward a moderate range before contact.

Contact force regulation. During CONTACT, a ramped target force is used to soften the first contact steps. The reward promotes accurate normal-force tracking and penalises force oscillations through a Δf_z stability term. Additional terms shape K_z toward an optimal range, adapt it based on force error (softer for large errors, stiffer when tracking is good), and favour a soft-contact setting in the first steps.

Contact quality and task progress. The reward includes a contact-quality term, a polishing-work term based on lateral contact forces, and a waypoint-progress term to encourage completing the stroke.

Early-contact penalties. To limit impact, penalties are applied to force overshoot and Δf_z during early contact, together with a hard-limit penalty and a penalty for excessive downward velocity right after contact.

Rise and global regularisation. During RISE, small bonuses reward upward motion and clearance. Global regularisation penalises large ΔK_z changes, adds a light time penalty and a mild energy term, and may include penalties for workspace-bounds violations or flagged contact-loss events.

Stiffness-and-Damping Reward $\mathscr{R}_{K_z,\zeta_z}$

The $K_z + \zeta_z$ configuration extends $\mathscr{R}_{K_z}$ by adding damping-specific shaping during contact. All descent, contact-force tracking, rise, and global terms are retained. The additional components are:

Soft-contact damping preference. During the first contact steps, a bonus keeps ζ_z near a target value to encourage well-damped contact initiation.

Force-oscillation penalty weighted by damping. A penalty on $|\Delta f_z|$ is scaled by the deviation of ζ_z from a nominal damping target, so oscillations are penalised more when damping is far from the preferred range.

Damping change penalty. Large step-to-step changes in ζ_z are penalised to discourage aggressive swings in damping.

Stable damping band. Additional penalties apply when ζ_z falls below or above a defined stability band, reinforcing a moderate damping regime during contact.

4.5 Learning-from-Demonstration and RL

Reinforcement Learning (RL) applied in conjunction with Learning-from-Demonstration (LfD) has emerged as a widely adopted strategy in robotics for problems where hand-engineered controllers struggle to capture complex behaviours, especially under contact-rich interactions, such as surface finishing (Davchev et al. (2022); Shi et al. (2021); Xu et al. (2025)). Despite its promise, RL remains challenging to deploy in a straightforward manner due to the intrinsic complexity of training, including high sample requirements, sensitivity to hyperparameters, convergence instabilities, and substantial computational demands. In this work, demonstrations are used to reduce the effective learning complexity by providing a structured task representation, specifying what should be executed, and thereby constraining the optimisation problem faced by the agent, which focuses on how the task is executed.

The contact-rich polishing pipeline introduced in this chapter therefore combines a structured reference motion, which specifies *what* to execute, and learning-based optimisation, which refines *how* the task is executed. In this context, LfD provides a practical mechanism to encode task intent as a demonstrated end-effector path, represented as a waypoint sequence, while RL enables autonomous improvement through trial-and-error in simulation.

A key practical requirement that drives the design choices in this section is trainability on *commodity* hardware, i.e., standard, non-specialized computing platforms

commonly available in industrial or academic settings. To this end, a discrete tracking formulation is adopted, and exploration is constrained through a compact discrete action space in Cartesian coordinates, rigid safety masks and admissibility constraints that prevent inadmissible moves, and a stable low-level execution pipeline, based on Differential IK and joint-space PD control, which guarantees consistent command tracking and reduces failure-prone rollouts (Fig.4.12

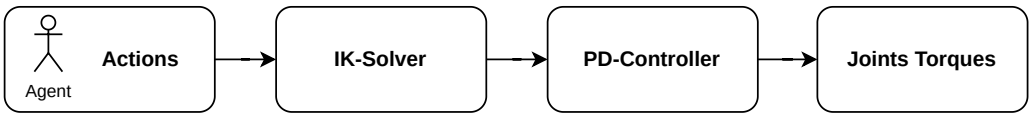


Figure 4.12 Control pipeline adopted to ensure stable execution and reliable command tracking on commodity hardware.

Up to this point, the framework has focused on a deterministic control strategy to manage contact during polishing, favouring stability and safety. By contrast, LfD+RL introduces data-driven decision-making, which, if left unconstrained, can be harder to validate and less predictable under variability, a relevant concern for industrial deployment. For this reason, the contribution of this section is positioned as a preliminary, tightly constrained LfD+RL strategy for surface finishing, the learning component is limited to small Cartesian displacements and is filtered through hard constraints, while low-level tracking remains fully controlled.

The remainder of the section is organised as follows: (i) problem definition, (ii) trajectory generation and pre-processing from demonstrations, (iii) definition of environment constraints, including safety masks and admissible regions, and (iv) reward design. Quantitative results are reported in Chapter 5.

4.5.1 Problem description: tracking and transition to contact

The transition from free-space pose tracking to in-contact force regulation is a classical problem in robot interaction control, historically framed by hybrid position/force schemes Raibert and Craig (1981), impedance-based interaction control Hogan (1985), and operational-space formulations Khatib (1987). As discussed in the introductory part of this chapter (Pose/Force tracking and transition), a central difficulty lies in deciding *when* and *how* objectives should be switched or blended. Even small tracking errors accumulated in free space can lead to undesirable transients at contact onset, for example force overshoots and unstable sliding, which are particularly detrimental

in finishing tasks Brun et al. (2003). In this framework, reinforcement learning is introduced to improve robustness and adaptability of the execution in contact-rich conditions, at the cost of additional training complexity and the need for careful constraint handling.

For continuous-contact finishing, the overall problem can be organised into a sequence of phases that are naturally addressed in a staged manner:

- **Tracking:** first, solve demonstration tracking in free space using a position-based objective, establishing stable waypoint following.
- **Pre-contact transition:** define a transition region around an estimated contact onset, inferred from the force traces recorded during the human demonstration. Within this region, the motion must be modified in terms of velocity and acceleration to reduce impact transients, without disrupting waypoint tracking.
- **Contact:** solve the in-contact behaviour by introducing an explicit interaction objective, for example via impedance-based regulation or a hybrid position/force formulation.
- **Detach:** handle the detachment phase to return to safe conditions and avoid undesired adhesion or surface damage.

This staged formulation supports a cascade development strategy, each phase is studied and validated independently, and only afterwards integrated into a full execution cycle. At the current stage, only the first phase, tracking, is implemented and described in detail in the following subsections, while the remaining phases are included to frame the intended extension of the polishing pipeline. Results for the tracking stage are reported in Chapter 5.

4.5.2 Demonstration interface: from RGB tracking to robot-base waypoints

As described in Chapter 3, demonstrations are acquired using the same camera-based motion-capture setup adopted throughout the thesis. The measurement stack, including pose tracking and the force-sensing instrumentation, is the same as the one illustrated in Fig. 4.1. Human tool motion is reconstructed in a triangulation reference frame via multi-view triangulation and post-processing. The output of the demonstration

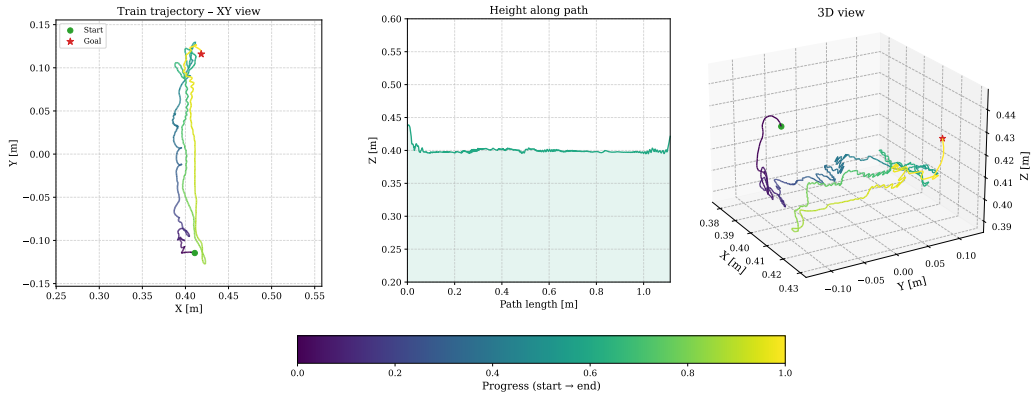


Figure 4.13 Training waypoint trajectory extracted from the camera-based demonstration interface and expressed in the robot base frame. The waypoint sequence defines the free-space tracking reference and is treated as a task-level specification (waypoint progression), not as a high-precision geometric ground truth.

interface is a time-ordered sequence of 3D tool-tip positions, which is then expressed in the robot base frame through the calibration transform and interpreted as a waypoint trajectory encoding task intent in free space.

The resulting waypoint sequence can be resampled and smoothed to obtain a consistent spatial and temporal resolution. While the acquisition pipeline is inherently affected by reconstruction uncertainty, this is tolerated at the current stage because the discrete tracking formulation operates with waypoint tolerances and focuses on reliable waypoint progression rather than millimetre-accurate geometric reproduction. This design choice is reflected in the tracking evaluation reported in Chapter 5.

Multiple trajectories were recorded for purposes that extend beyond the scope of the present discussion. For the results reported in this thesis, two waypoint trajectories are selected, one for training and one for testing. The training trajectory used to fit the policy is shown in Fig. 4.15, while the held-out test trajectory, similar in structure but not identical, is used for evaluation (Chapter 5).

4.5.3 Discrete LfD tracking formulation

This subsection specifies the implemented tracking-only LfD mode as an MDP and execution interface, following the same notation and methodological structure adopted in Sec. 4.4.2. The per-step processing pipeline is summarised in Fig. 4.14; the present subsection focuses on the MDP components that define the tracking task.

Episode definition and time discretisation

An episode corresponds to traversing the full demonstrated waypoint sequence. Control is executed at a fixed frequency $f_c = 120$ Hz, so that each control step has duration $\Delta t = 1/f_c \approx 8.3$ ms. Completing one waypoint sequence typically requires 3000 to 5000 control steps, corresponding to approximately 25 to 42 s of simulated time. Throughout this section, the term *step* denotes one control cycle at f_c, rather than a PPO rollout horizon (cf. Table B.1).

Action space: 27 discrete Cartesian displacements

In the discrete LfD mode, the policy selects one action from a finite set of Cartesian displacements,

$$(\Delta x, \Delta y, \Delta z) \in \{-1, 0, +1\}^3 \cdot s,$$

resulting in $3^3 = 27$ actions per time step, where s is the Cartesian step size expressed in metres. In the experiments reported in Chapter 5, the step size is set to

$$s = \texttt{LFD\_DIS\_STEP\_CM} \cdot 10^{-2}$$

The action affects end-effector *position only*; orientation is kept fixed by the task configuration.

Observations

The policy input is constructed as a temporal observation buffer of length `SEQ_LEN`, stacking the most recent signals required for waypoint tracking and for forward compatibility with contact-aware extensions. At each control step, a one-step feature vector o_t^{step} is assembled from robot- and task-level quantities and then appended to a FIFO history buffer. In the configuration used in this work, `SEQ_LEN` $= 128$ and $o_t^{\text{step}} \in \mathbb{R}^{36}$, yielding a flattened policy input $o_t \in \mathbb{R}^{128\times 36} = \mathbb{R}^{4608}$.

The one-step observation vector includes:

- **Joint state:** joint positions $q \in \mathbb{R}^7$, joint velocities $\dot{q} \in \mathbb{R}^7$, and joint accelerations $\ddot{q} \in \mathbb{R}^7$.
- **Actuation feedback:** applied joint torques $\tau \in \mathbb{R}^7$, corresponding to the torques executed at the previous control cycle.

- **Task-space tracking signals:** end-effector position $ee_{\text{pos}} \in \mathbb{R}^3$ and current waypoint position $goal_{\text{pos}} \in \mathbb{R}^3$.

- **Auxiliary signals:** the filtered normal contact force $f_z \in \mathbb{R}$ (EMA filtered) and a discrete phase indicator $phase \in \mathbb{R}$, with $phase = 0$ in free space, $phase = 1$ in contact, and $phase = 2$ for completion.

The history buffer is updated at every step by shifting the stored sequence and inserting the newest o_t^{step} at the end (implemented as a rolling buffer). While the present tracking-only experiments do not optimise contact behaviour explicitly, including f_z and the phase indicator makes the observation interface consistent with the broader polishing framework and supports subsequent extensions where transition and contact objectives are activated.

Action filtering: hold, directional mask, and clamping

To keep exploration safe and deterministic, the raw discrete action proposed by the policy is post-processed by a small set of hard constraints applied in task space. The overall logic is summarized in Fig. 4.14.

- **Hold (zero-action stabilization).** When the selected discrete action corresponds to a null Cartesian displacement ($\Delta = \mathbf{0}$) and the end-effector is already sufficiently close to the current waypoint ($dist < \varepsilon_{\text{hold}}$), the controller keeps the previously commanded target cmd_{pos}. This avoids command jitter around the goal and improves reproducibility.

- **Directional mask (anti-regression).** To prevent regressions, actions that move *against* the waypoint direction are vetoed. Let

$$goal_{\text{dir}} = \frac{goal - ee_{\text{pos}}}{\|goal - ee_{\text{pos}}\|}.$$

  The candidate displacement Δ is rejected if it points away from the waypoint:

$$\Delta \cdot goal_{\text{dir}} < -tol.$$

  If rejected, the last valid command is retained; otherwise, the new target is accepted.

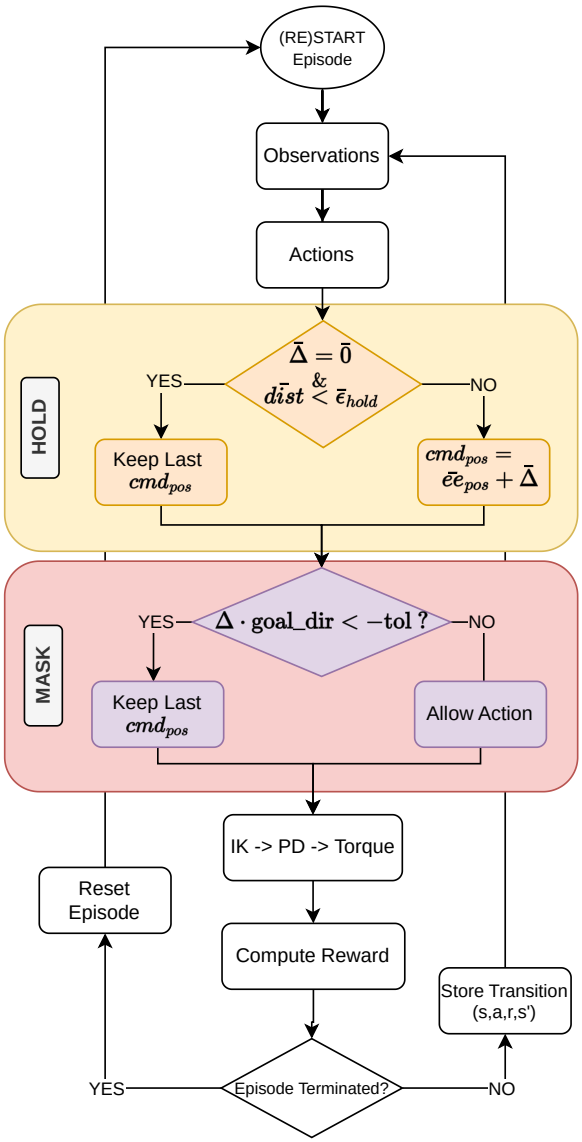


Figure 4.14 Discrete-action post-processing used in the LfD discrete agent.

- **Clamping (workspace/trajectory bounds).** After hold/mask, the resulting target cmd_{pos} is clamped (i.e., projected) into an admissible region defined around the demonstrated trajectory and/or workspace limits. This step is a hard safety constraint and is applied at every control step. For clarity, clamping is not explicitly expanded in Fig. 4.14, but it is always active in the default configuration.

In the default setup these filters are treated as *hard constraints*; they can be disabled only for controlled ablation studies.

Low-level execution: target position → differential IK → joint PD

For each accepted step, the agent operates at the task-space decision level by producing a Cartesian increment Δ, which defines the next end-effector target position:

$$cmd_{\text{pos}}^{t+1} = ee_{\text{pos}}^{t} + \Delta,$$

possibly modified by the hold/mask logic and clamping. The target is then tracked by a standard differential inverse kinematics (IK) layer followed by joint-space PD control. The per-step execution chain can be summarized as:

$$\pi(o_t) \;\rightarrow\; \Delta \;\rightarrow\; cmd_{\text{pos}} \;\rightarrow\; (q^d, \dot{q}^d) \;\rightarrow\; \tau.$$

This separation keeps the RL policy focused on *where to move next* in task space, while relying on a stable low-level tracking interface for torque generation and simulation execution.

Reward design: waypoint tracking

The reward function is designed to solve the free-space waypoint tracking problem under a discrete Cartesian action space. At each step t, the agent is assigned a current goal waypoint selected by the index wpt_idx, $goal = waypoints[wpt\_idx]$, and the absolute per-axis tracking error is computed as $\delta = |ee_{\text{pos}} - goal| = (\delta_x, \delta_y, \delta_z)$.

A key difficulty in discrete tracking is that a fully coupled 3D objective can lead to oscillations or indecisive behaviour, especially when the agent must choose among a small set of axis-aligned displacements. For this reason, the reward uses a *sequential axis priority* mechanism: the agent is encouraged to reduce one axis at a time (X→Y→Z), while still requiring a full 3D hit to advance the waypoint.

Sequential axis priority. The active axis is selected hierarchically. Intuitively, the policy first reduces the error along x until it is within a tolerance ε_{in}; only then it focuses on y, and finally on z. This is encoded by defining a scalar distance $active\_dist$ as

$$active\_dist = \begin{cases} \delta_x, & \delta_x \geq \varepsilon_{\text{in}}, \\ \delta_y, & \delta_x < \varepsilon_{\text{in}} \;\wedge\; \delta_y \geq \varepsilon_{\text{in}}, \\ \delta_z, & \text{otherwise.} \end{cases}$$

As a result, the shaping and progress signals described below remain informative even when only one component is actively optimised at a time.

Reward structure. The per-step reward is constructed to encourage (i) proximity to the current goal along the active axis, (ii) *monotonic* improvement across steps, (iii) precise completion of the waypoint in full 3D, and (iv) timely progression. These objectives are combined as a weighted sum:

$$r_t = w_{wp}\, r_{wp} + w_{prog}\, r_{prog} + w_{hit}\, r_{hit} + w_{time}\, r_{time},$$

where the weights $w_\cdot$ control the relative influence of each term.

Distance shaping (dense). A smooth gradient towards the waypoint is provided by an exponential shaping term computed on $active\_dist$,

$$r_{wp} = \exp\left(-\frac{active\_dist^2}{\sigma^2}\right),$$

so that $r_{wp} \approx 1$ near the goal and decays smoothly as the active-axis distance increases, with σ setting the decay scale.

Progress (dense, directional). To explicitly discourage oscillations and regress, the agent is rewarded when it reduces the active-axis distance compared to the previous step. Let *prev_dist* denote the stored value of $active\_dist$ from the previous control cycle. The progress term is

$$r_{prog} = \mathrm{clip}\big(prev\_dist - active\_dist,\ -c_{\mathrm{clip}},\ +c_{\mathrm{clip}}\big).$$

This contribution is positive when the motion decreases the active-axis error and negative when the agent moves away; clipping by c_{clip} prevents large jumps from dominating the return. At the first step of an episode, r_{prog} is set to zero since no previous distance is available.

Hit bonus (sparse). Waypoint completion is enforced through a binary condition on the *full* 3D tolerance. A waypoint is considered reached when

$$hit \equiv (\delta_x < \varepsilon_{\mathrm{in}}) \wedge (\delta_y < \varepsilon_{\mathrm{in}}) \wedge (\delta_z < \varepsilon_{\mathrm{in}}), \qquad r_{hit} = \begin{cases} 1, & hit, \\ 0, & \text{otherwise.} \end{cases}$$

This term provides an unambiguous completion signal and is the trigger used to advance the waypoint index.

Time cost. Finally, a constant per-step term penalises slow behaviour. In practice $r_{time} = 1$ and $w_{time} < 0$, so that each additional step incurs a small negative cost. This encourages the agent to reach each waypoint efficiently without relying on unnecessarily long rollouts.

Waypoint advancement and buffers. When *hit* is true, the waypoint index advances to the next waypoint,

$$wpt\_idx \leftarrow \min(wpt\_idx + 1,\ M - 1),$$

and the distance buffer is updated at every step as $prev\_dist \leftarrow active\_dist$.

Termination and truncation

Episodes end either because the task is completed (`terminated`) or because an external time limit is reached (`truncated`). In both cases, the PPO buffer stores a single boolean flag $done = terminated \vee truncated$, which is used for bootstrapping and advantage estimation (GAE).

Task termination (success). Task completion is defined at the waypoint level. Let $goal = waypoints[wpt\_idx]$ and $\delta = |ee_{\text{pos}} - goal|$ denote the per-axis absolute tracking error. A waypoint is considered reached when

$$hit \ \equiv\ (\delta_x < \varepsilon_{\text{in}}) \ \wedge\ (\delta_y < \varepsilon_{\text{in}}) \ \wedge\ (\delta_z < \varepsilon_{\text{in}}).$$

The current waypoint index advances *only* on *hit*. Accordingly, an episode is marked as successful and `terminated` only when the final waypoint is reached,

$$success \ \equiv\ (wpt\_idx = M - 1) \ \wedge\ hit,$$

where M is the number of waypoints.

Time truncation (timeout). To prevent unbounded rollouts, each episode is subject to a fixed maximum duration. If the time limit is reached before success, the episode

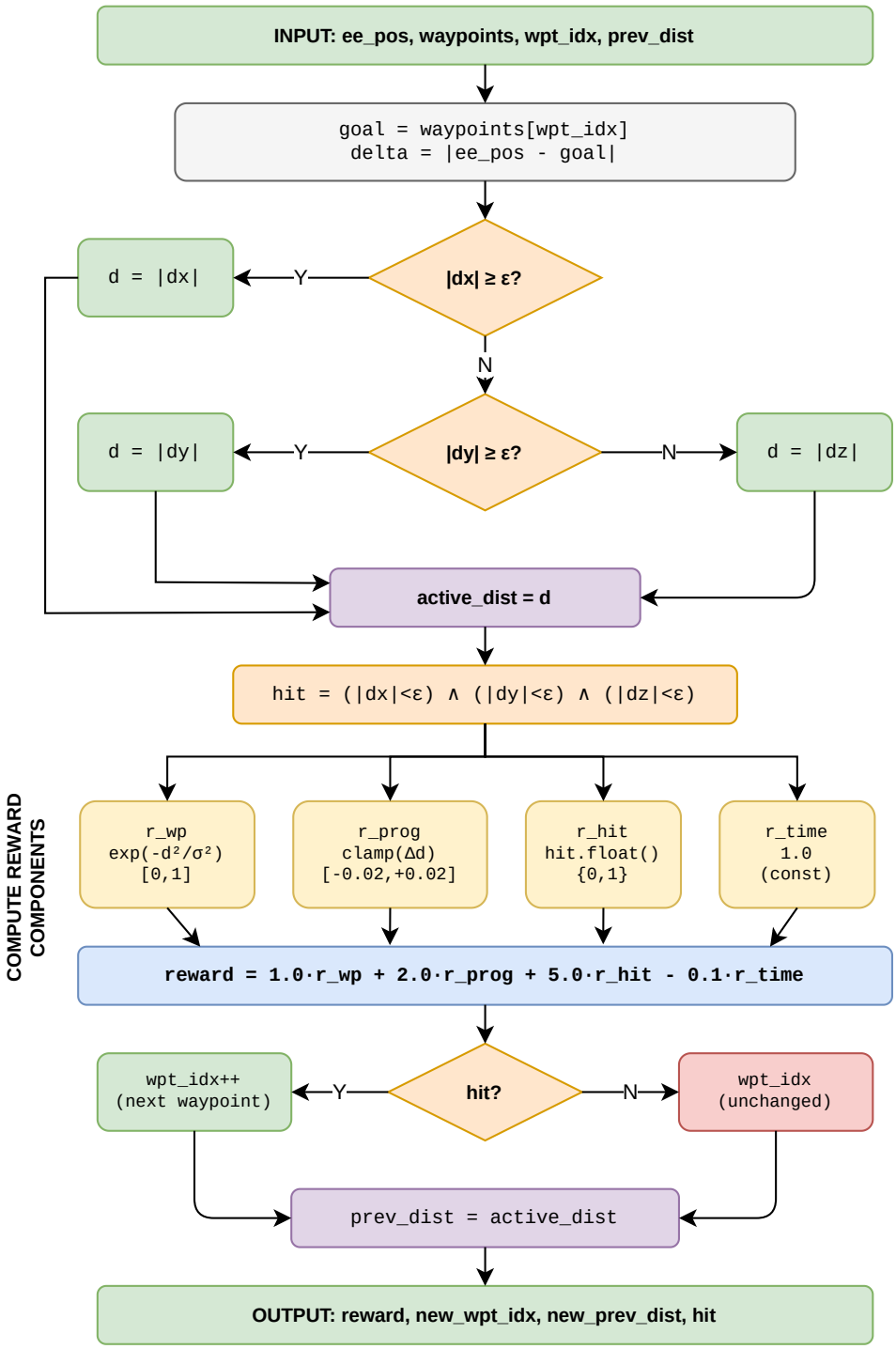


Figure 4.15 Flowchart of the Reward Function.

is `truncated`. This limit is configuration-driven and is independent of the waypoint count.

Stuck detection. A conservative stuck criterion is enabled to handle cases where the policy fails to reduce the tracking error for prolonged periods. Consistently with the sequential-axis formulation of the reward, the criterion is evaluated on the active-axis distance *active_dist* (X→Y→Z priority). If *active_dist* remains above a configurable threshold d_{stuck} for an extended horizon, the episode is marked as `terminated` (failure).

The weighting coefficients in Fig. 4.15 were chosen by empirical tuning rather than by ad hoc manual selection. After first identifying a stable OSC baseline, the reward weights were refined through a grid search over candidate configurations for the RL–k_z and RL–$k_z + \zeta_z$ cases. Each configuration was evaluated by running training and comparing the achieved score and qualitative contact behaviour. In practice, the weighting strategy was designed so that task-critical terms (e.g., waypoint progression, contact acquisition, and stable interaction) dominated the optimization objective, while secondary penalties (e.g., time pressure and regularization terms) acted only as mild shaping components. Full details of the search procedure are given in Chapter 5, and the final default coefficients are reported in Appendix E.

4.5.4 Roadmap toward contact-aware demonstration reproduction

While the current implementation addresses tracking in position space, the target use case remains contact-rich manipulation. The planned extension follows the staged strategy outlined earlier:

- identify a **transition region** around the contact point, estimated from force traces recorded during human demonstrations;
- introduce an additional **transition reward** that shapes velocity and acceleration in this region, without degrading the tracking behaviour learned with `wp_v4_dis`;
- implement and compare alternative **contact-phase strategies** (e.g., model-based contact handling, impedance-based interaction, or hybrid approaches);
- define a robust **detach phase** to ensure safe and repeatable contact release.

Part III

Section Three

Chapter 5

Results

"The present state determines the
future, but does not appear to do so."

Edward N. Lorenz (1993)

This chapter reports the results obtained to date from the software and methodological pipeline described in the previous chapters. The chapter is organized into two parts. First, it presents results and demonstrations that extend beyond the methodological exposition, including the current state of the Procedural Trajectory Mode and the Trajectory by Learning from Demonstration stage. Second, it reports an analysis based on Principal Component Analysis (PCA) aimed at deriving an interpretable behavioural fingerprint for RL-augmented operational-space control, with the goal of supporting early diagnosis and more reproducible comparisons across training runs.

This section consolidates the results currently available from the implemented pipeline, with emphasis on demonstrators that validate individual components before full integration. In line with the staged development strategy described earlier, the present results focus on modes that are already implemented and can be evaluated with the existing logging and diagnostic subsystem.

5.1 Procedural Trajectory Mode

In the Procedural Trajectory Mode, the objective is to utilise a policy to stabilise the contact transient, the force applied on the surface (constant force), and to reduce the operating time. Furthermore, a randomisation of contact parameters, such as friction

between the tool and the surface, has been evaluated in order to ascertain the robustness of the algorithm. The comparison of the results is carried out on the following three tests:

- OSC_{only}: In this case, no RL algorithm is employed, and the controller serves as a performance baseline.
- RL–k_z: The policy is allowed to modify only the stiffness along the Z axis of the TCP frame.
- RL–$k_z + d_z$: The policy is allowed to modify both stiffness and damping along the Z axis of the TCP frame.

So, in the last two cases, the agent should learn how to modify only two parameters. More than these two parameters have been omitted in the evaluation due to the effort required for training. The agent can modify only an increment Δ on RL–k_z and RL–$k_z + d_z$. The experimental procedure followed four main stages:

1. determination of reference parameters for the OSC baseline;
2. grid search of reward-function coefficients for the RL–k_z and RL–$k_z + d_z$ cases;
3. training of the RL policies;
4. generation and analysis of the results.

To obtain a reliable baseline for comparison, the stiffness and damping parameters of the OSC controller were optimized prior to running any RL-based experiments. These optimal values are kept fixed throughout all tests and serve as the initial reference point for both RL configurations. In particular, the policies in the and $K_z D_z$ settings apply incremental adjustments with respect to these baseline parameters rather than learning them from scratch.

The local optimization of the reward-function coefficients was performed through a grid search. By evaluating multiple configurations, it is possible to visually identify which parameters have the strongest influence on reward. However, these observations must be interpreted with caution, as certain coefficients may dominate the reward landscape and overshadow the contribution of other terms.

OSC parameters optimization To find the best parameters of the controller (free motion and in contact), a random grid search has been performed. The process is based on 30 trials of the experiment and the best results are stored. In the following, the process flow is explained:

Process flow of the random grid search used to optimize OSC parameters.

```
START
build_search_space
    - K_z = linspace(500, 6000, n_kp)
    - D_z = linspace(0.3, 2.0, n_zeta)
    - search_space = product(kp_z_values, zeta_z_values)

best_score  = +inf
best_params = None

for each (K_z, D_z) in search_space:
    metrics = run_episodes(K_p,z, ζ_z)

    score = 0.5 * force_error_mean
          + 0.3 * sqrt(max(0, force_stability))
          + 0.2 * (1 - completion_rate) * 50

    if score < best_score:
        best_score  = score
        best_params = (K_z, D_z)
END
```

The parameter search along the Z axis produced the following values:

$$K_z = 5290.7536, \qquad D_z = 0.9131,$$

Reward grid search. The optimal reward configuration is investigated for both RL–k_z and RL–k_z+d_z. Differently from the previous search, this stage evaluates candidate settings by running training and measuring the achieved score. The RL–k_z study includes 62 runs of 200 epochs each, while the RL–k_z+d_z study includes 122 runs of 20 epochs each. This asymmetry reflects different search complexity: RL–k_z exhibited a clearer optimum region, enabling fine-grained refinement with longer runs, whereas RL–k_z+d_z showed a less interpretable response and a larger effective search space, motivating shorter runs to test more configurations.

5.2 Friction Randomization: Rationale and Outcomes

Tests carried out without randomisation showed almost identical results across the various trials. However, by introducing randomisation, the influence of the RL policy can be better appreciated. The results obtained in this test are described below.

First, the trajectory and force outputs are reported. Subsequently, a statistical analysis of the policy's robustness with respect to variations in the friction coefficient is presented. In these tests, the friction coefficient was set such that static and dynamic friction were equal. Protocol: 20 values of μ, 20 episodes per μ, 1 seed per μ.

5.2.1 Results: Trajectories

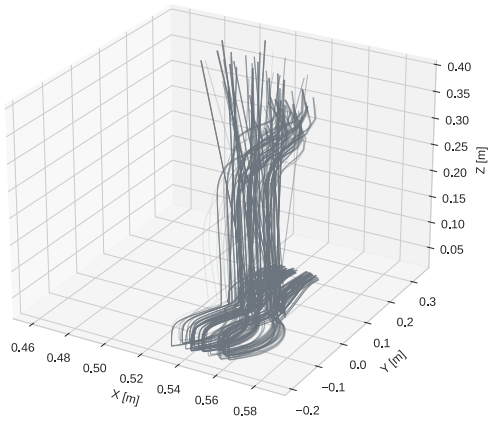
Figure 5.8 shows the trajectories executed by: (a) the OSC controller alone; (b) a policy that adapts the stiffness along the Z-axis; and (c) a policy that adapts both stiffness and damping along the Z-axis. Overall, the trajectories are qualitatively similar across the three conditions, with no substantial differences observed among the tests.

5.2.2 Results: Forces

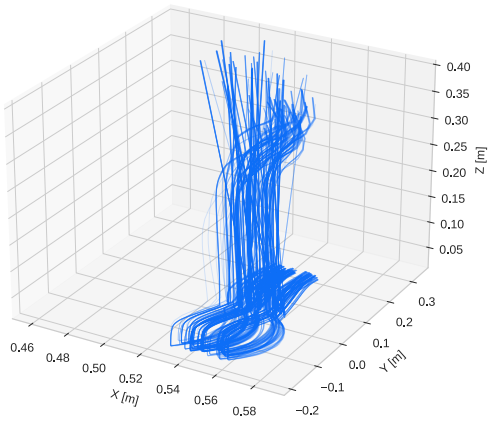
Considering the contact phase (defined from the first instant at which $F_z < -1$ N until contact is lost, i.e., $F_z > 0.5$ N), the contact duration is essentially identical across controllers (approximately 22.4 s). Among the tested strategies, KZ+DZ yields the least negative mean normal force (-14.62 N) and the lowest mean force variation. OSC, in contrast, exhibits a more contained interaction, with smaller oscillation amplitudes, including under changes in friction. The KZ-ONLY policy generally achieves a mean contact force $\bar{F}_z$ closer to the nominal setpoint and a lower peak force-rate (i.e., reduced maximum force variation) than the other cases. Overall, introducing the learned policy in the contact regime improves the ability to adapt to variations in friction. Result are summurized in Tab 5.1 and Fig. 5.2.

5.2.3 Results: Adaptability

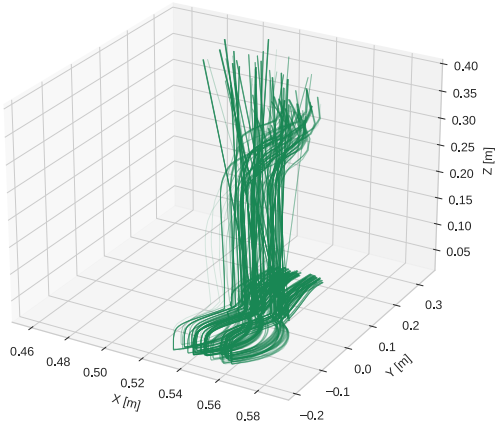
Figure 5.3 reports an *adaptability index* (positive values indicate better performance), defined as the average relative improvement with respect to OSC on the *oscillation* and *contact-ratio* metrics. Table 5.2 reports the corresponding per-metric improvements,



(a) OSC



(b) RL K_z



(c) RL K_z, D_z

Figure 5.1 Variation in trajectories in different tests as friction varies

and additionally includes the improvement on the minimum normal-force peak as a contact-safety indicator.

For a metric m evaluated at a given friction value μ, the relative improvement over OSC is computed as

$$\Delta_m(\mu) = \begin{cases} \dfrac{x_m(\mu) - x_m^{\mathrm{OSC}}(\mu)}{|x_m^{\mathrm{OSC}}(\mu)| + \varepsilon}, & m \in \{\text{contact ratio}, \min F_z\}, \\ \dfrac{x_m^{\mathrm{OSC}}(\mu) - x_m(\mu)}{|x_m^{\mathrm{OSC}}(\mu)| + \varepsilon}, & m \in \{\text{oscillation}\}, \end{cases}$$

Table 5.1 Contact-force metrics up to rising (averaged over μ).

Controller	$\bar{F}_z$ [N]	mean $\lvert dF \rvert$ [N]	max $\lvert dF \rvert$ [N]	duration [s]
OSC	-14.95	0.66	36.94	22.47
KZ	-15.94	0.62	34.33	22.42
KZ+DZ	-14.62	0.61	35.01	22.51

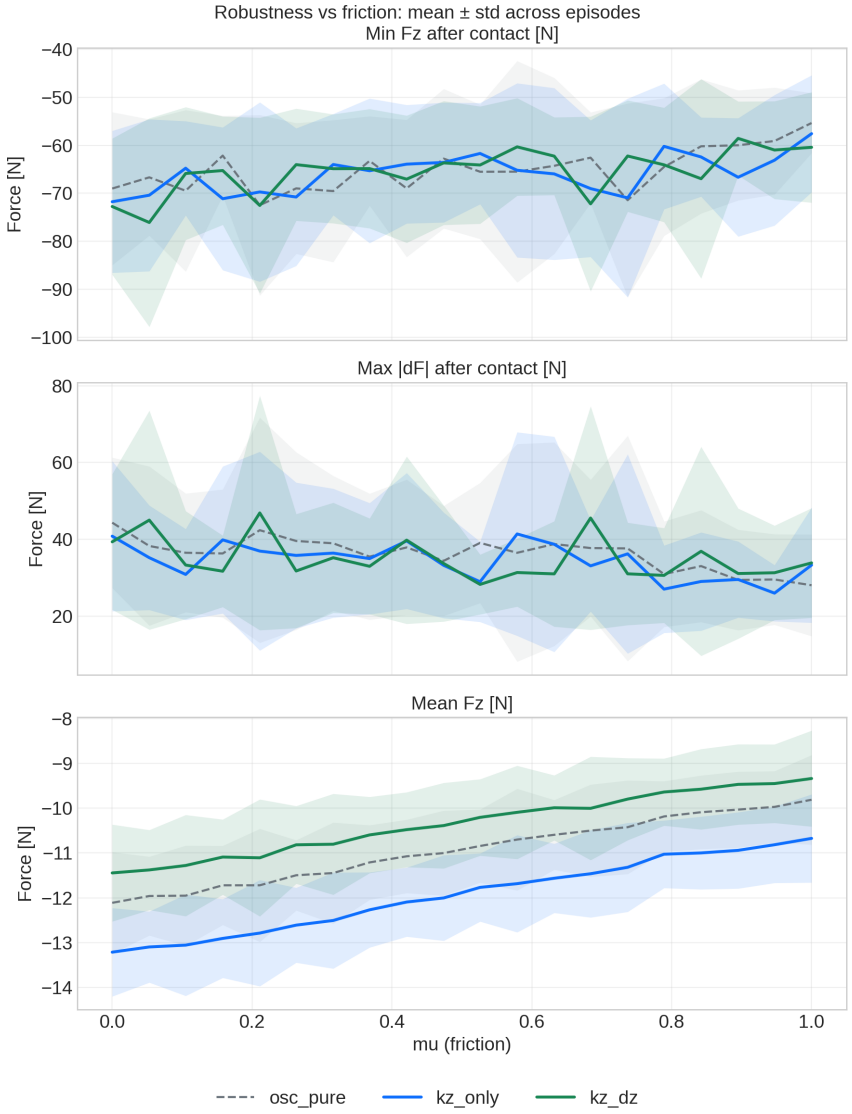


Figure 5.2 Robustness over friction: mean $\pm$ std across μ.

where $x_m(\mu)$ is the metric value for the controller at friction μ and ε is a small constant to avoid division by zero. Here, $\min F_z$ denotes the *signed* minimum normal force

Controller	Overall	Oscillation	Contact ratio	Min F_z
OSC	0.000	0.000	0.000	0.000
KZ	0.038	0.049	0.027	-0.015
KZ+DZ	-0.003	0.027	-0.033	-0.008

Table 5.2 Relative improvements vs OSC (averaged over μ). Overall = mean of oscillation and contact-ratio improvements.

during contact; less negative minima (i.e., larger values) correspond to smaller force overshoots and are therefore considered beneficial.

The values reported in Table 5.2 are averages over μ. The overall adaptability index shown in Fig. 5.3 is then defined as the mean of the improvements for *oscillation* and *contact ratio*.

Key findings and summary. Friction randomization ($\mu \in [0,1]$) is necessary to reveal meaningful behavioural differences across controllers, since non-randomized tests produced nearly identical outcomes. Across the full friction sweep, all controllers exhibit comparable contact durations (about 22.4 s), confirming that the policies primarily affect interaction quality rather than task timing. With respect to force regulation, KZ+DZ achieves the least negative mean normal force and the lowest mean force variation, suggesting a softer average interaction, whereas OSC maintains

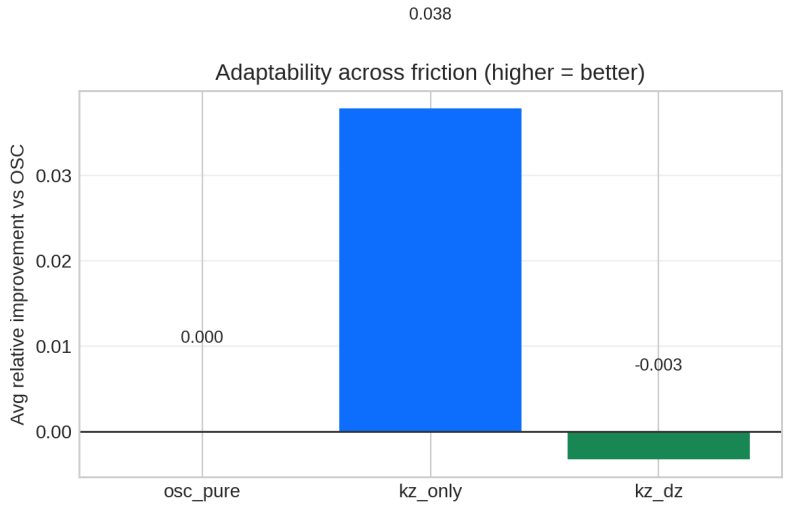


Figure 5.3 Adaptability index across the friction sweep (relative to OSC).

the most contained contact with smaller oscillation amplitudes under friction changes. The KZ-ONLY policy yields a mean contact force closer to the nominal setpoint and the lowest peak force-rate, indicating improved transient shaping at impact and during force regulation. Robustness and adaptability metrics further confirm these trends: KZ-ONLY provides the most consistent improvement over OSC, reducing oscillations for $15/20$ friction values (average $\approx +4.9\%$) while improving contact continuity for $20/20$ values (average $\approx +2.7\%$). Conversely, KZ+DZ reduces oscillations in $12/20$ cases (average $\approx +2.7\%$) but consistently degrades contact ratio ($20/20$, average $\approx -3.35\%$), indicating a trade-off between softer interaction and contact stability. Finally, the minimum normal-force peak ($\min F_z$) shows mixed behaviour for both RL variants, with no consistent improvement across μ. Overall, introducing the learned policy improves adaptability to friction variations; among the evaluated designs, KZ-ONLY offers the most reliable robustness gain, while KZ+DZ favours softer average contact at the expense of contact continuity.

5.3 Trajectory by Learning from Demonstration

This subsection reports the results obtained for the tracking-only Learning from Demonstration mode. At the current stage, the implemented component addresses free-space waypoint tracking. The tracking reference is a demonstrated waypoint trajectory obtained through the camera-based demonstration interface and expressed in the robot base frame. Two trajectories were recorded and selected for the results reported here: a training trajectory used to fit the policy, previously introduced in the thesis, and a held-out test trajectory corresponding to a second demonstration.

The two demonstrated trajectories are summarized to contextualize differences in geometric extent and waypoint discretization. Table 5.3 compares the training and test trajectories in terms of waypoint count, physical length, nominal duration, and basic smoothness proxies derived from successive waypoint increments. The test trajectory is substantially longer than the training trajectory, while exhibiting a comparable average speed. Both trajectories occupy a similar spatial envelope in the robot base frame, as indicated by the bounding-box spans and the reported axis-aligned minima and maxima. The distribution of successive waypoint steps differs across trajectories, with the test trajectory exhibiting a markedly larger maximum step, which indicates the presence of occasional large waypoint jumps relative to its typical discretization. A qualitative visualization of the two trajectories is provided in Fig. 5.4.

Training speed is reported as the number of PPO epochs required to produce the best checkpoint. In the current implementation, the best checkpoint is selected solely based on episode return. The deployed tracker points to a best checkpoint saved at 200 PPO epochs. Training is performed using a single environment instance to prioritize

Table 5.3 Quantitative comparison of the demonstrated training and test waypoint trajectories used in the LfD tracking evaluation.

Metric	Train	Test
Points	1,337	3,750
Total length (m)	1.327	3.590
Duration (s)	26.717	75.439
Avg speed (mm/s)	49.66	47.59
Bounding box span (mm)	$38.6 \times 257.1 \times 44.0$	$45.1 \times 270.1 \times 44.4$
Mean / median / max step (mm)	0.993 / 0.803 / 10.08	0.958 / 0.445 / 77.33
Mean turn angle (deg)	37.09	25.22

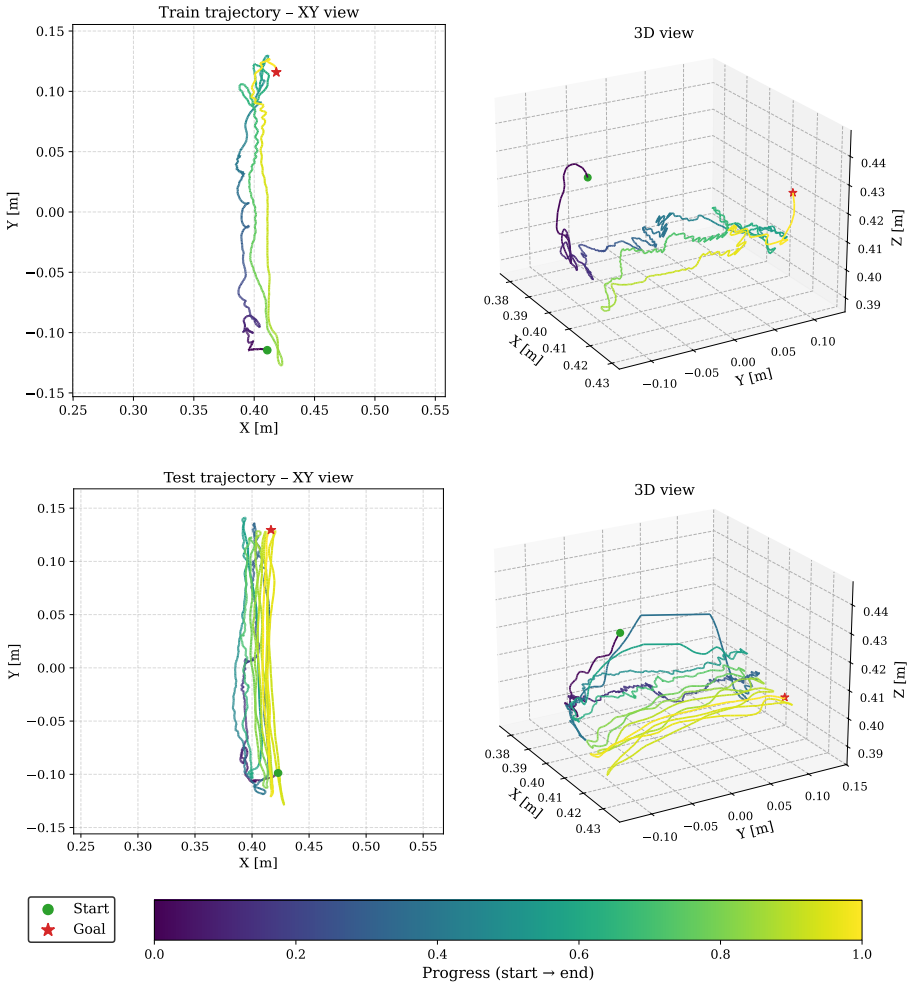


Figure 5.4 Training and held-out test waypoint trajectories used for the LfD tracking evaluation, expressed in the robot base frame.

stability. This epoch count is reported as an empirical indicator of training speed for the present configuration, and it can vary when changing the demonstrated trajectory or adjusting hyperparameters

Execution reliability is evaluated using a strict completion-based success criterion under a fixed time limit. An episode is classified as a success if it reaches the last waypoint of the reference trajectory within the configured episode horizon, operationally corresponding to

$$\text{wpt\_coverage} = 1.0 \text{ and } \text{wpt\_max} = \text{traj\_len} - 1.$$

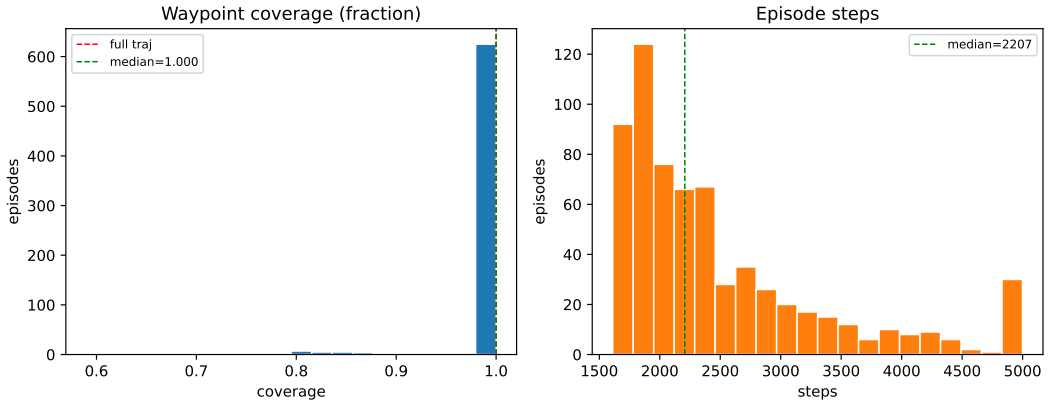


Figure 5.5 Distribution of waypoint coverage (fraction of trajectory reached) and episode length (steps) for the **training trajectory**

Since no episodes terminate due to violations of admissible regions, no additional error thresholds or safety-based termination conditions are used in this success definition.

Train-trajectory evaluation. The training trajectory (1337 waypoints) is evaluated as well. A total of 650 episodes were run. Among them, 623 episodes complete the trajectory within the configured time limit, while 27 do not, yielding a success rate of 95.85%. Waypoint coverage is highly concentrated at full completion, with a median of 1.0000 and an interquartile range of $[1.0000, 1.0000]$; the minimum observed coverage is 0.5898. In absolute terms, waypoint hits have a median of 1336 out of 1337, with an interquartile range of $[1336, 1336]$ and a minimum of 788. Episode-step statistics indicate that failures are due to timeouts: failed episodes terminate at a fixed horizon of 4999 steps, whereas successful episodes terminate earlier, with a median of 2171 steps (interquartile range $[1856, 2705]$).

Held-out test-trajectory evaluation. The held-out test condition is evaluated on a second demonstration trajectory (3750 waypoints). A total of $N = 659$ episodes were run. Among them, 263 episodes complete the trajectory within the time limit, while 396 do not, yielding a success rate of 39.91%. To complement the binary success label, waypoint coverage is used to quantify how far each episode progresses along the demonstrated sequence. Coverage has a mean of 0.9247 and a median of 0.9728, with an interquartile range of $[0.8861, 1.0000]$; the minimum and maximum observed coverages are 0.3798 and 1.0000, respectively. In absolute terms, waypoint hits have

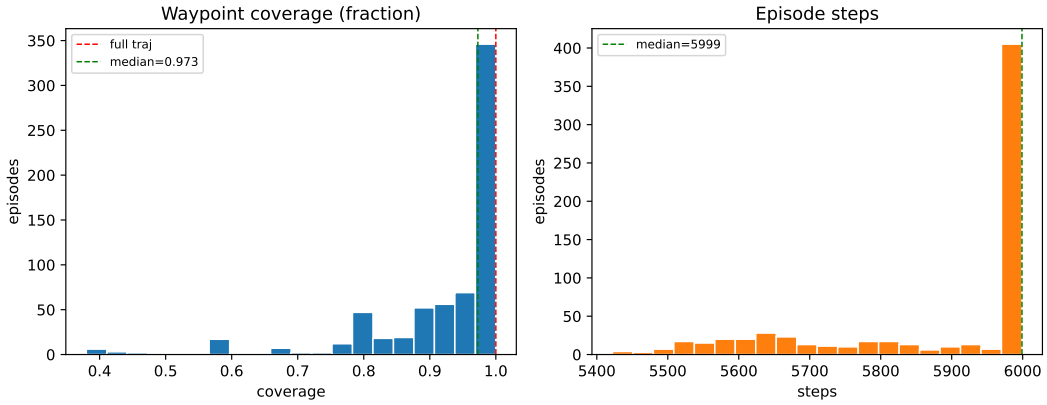


Figure 5.6 Distribution of waypoint coverage (fraction of trajectory reached) and episode length (steps) for the **held-out test trajectory**.

a mean of 3466.85 and a median of 3647 out of 3750, with an interquartile range of $[3322, 3749]$. Episode-step statistics indicate that failures are due to timeouts: failed episodes terminate at a fixed horizon of 5999 steps, whereas successful episodes terminate earlier, with a median of 5677 steps.

Key findings.

- The current LfD component addresses free-space waypoint tracking of a demonstrated trajectory expressed in the robot base frame, using a strict completion-based success criterion under a fixed time limit.

Table 5.4 Summary of LfD tracking performance on the training and held-out test trajectories under the completion-based success criterion.

Metric	Train (traj_1)	Test (traj_2)
Waypoints (traj_len)	1337	3750
Episodes N	650	659
Success rate (%)	95.85	39.91
Coverage (median [q25, q75])	1.0000 [1.0000, 1.0000]	0.9728 [0.8861, 1.0000]
Coverage (min, max)	0.5898, 1.0000	0.3798, 1.0000
Episode steps (median [q25, q75])	2206 [1862.75, 2791.00]	5999 [5749, 5999]
Step cap (failed episodes)	4999	5999

- The held-out test trajectory is substantially longer than the training trajectory (3750 vs 1337 waypoints; 3.59 m vs 1.33 m), while exhibiting a comparable average speed. The spatial envelope is similar, but the test trajectory includes markedly larger occasional waypoint jumps (max step 77.33 mm vs 10.08 mm).
- On the training trajectory, the tracker achieves a high success rate (95.85%, 623/650), with waypoint coverage concentrated at full completion (median 1.000). Failures are attributable to timeouts at the step cap (4999 steps), rather than safety violations.
- On the held-out test trajectory, the success rate drops to 39.91% (263/659) due to timeouts at the step cap (5999 steps). Nevertheless, coverage remains high even in failed episodes (median 0.9728, IQR [0.8861, 1.0000]), indicating that episodes often reach near-completion but exceed the time limit.
- Overall, generalization is partial: performance degrades primarily with increased trajectory length and irregular discretization, with the dominant failure mode being time-budget limitation rather than instability or constraint violations.
- A supplementary uncapped evaluation on the held-out trajectory ($N = 5$) shows full completion in all episodes, supporting the interpretation that the fixed-budget failures are mainly due to premature truncation.

Supplementary result: effect of removing the hard time cap. The fixed-budget evaluation above is retained as the main benchmark, since it reflects the originally configured execution constraint. As an additional analysis, the same held-out test trajectory was re-evaluated without the hard episode time cap, in order to isolate the effect of premature truncation from the trajectory-following capability of the policy. In the current supplementary batch ($N = 5$), all episodes reached the final waypoint, yielding a success rate of 100%. Waypoint coverage was 1.0000 in all evaluated episodes, while episode length ranged from 5439 to 6840 steps, with a median of 5502 steps. This supplementary result indicates that the failures observed under the fixed-budget protocol are primarily attributable to the imposed episode horizon rather than to an intrinsic inability of the policy to track the held-out trajectory. Because this uncapped analysis is currently based on a limited batch, it should be interpreted as supportive rather than as a replacement for the main benchmark.

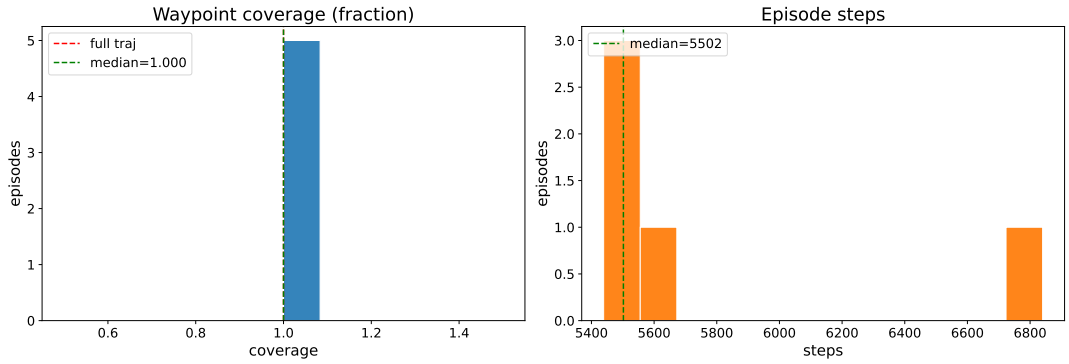


Figure 5.7 Supplementary held-out test evaluation without a hard episode time cap. All episodes in the current batch reach full trajectory completion.

5.4 PCA Fingerprint of RL-Augmented Operational-Space Control

The final result reported in this thesis derives from additional analyses conducted on the Procedural Trajectory Mode and concerns the exploration of diagnostic tools capable of indicating whether an RL agent is learning to execute the task correctly. To this end, Principal Component Analysis (PCA) was investigated on datasets collected during training and execution episodes.

The use of PCA to define behavioural "fingerprints" in dynamic systems is well established. In Lee et al. (2004), validation on a simple multivariate process and a wastewater benchmark shows improved fault detection over linear PCA by capturing nonlinear relationships among variables. Kano et al. (2001) detects operating-condition changes via shifts in principal component directions, while also noting delays in conventional MPCA in both fault detection and recognition of returns to nominal operation, which underscores the need for careful choices of window size and operating regime. For high-dimensional streams, Nabhan et al. (2019) introduces a robust sparse probabilistic PCA that outperforms conventional PCA in handling complexity and improving robustness. Finally, Zhang et al. (2021) proposes a PCA–EWC framework with continual-learning capability for multimode process monitoring.

Motivated by these findings, this thesis poses the following research question:

Is it possible to define a PCA-based fingerprint that an RL agent can exploit as an informative prior to learn the task more directly, thereby simplifying the learning process and enabling earlier diagnosis of undesirable behaviours?

This question is closely tied to the reproducibility challenge in deep RL. As shown by Henderson et al. (2019), algorithmic performance is highly sensitive to hyperparameters, random seeds, implementation details, and environment characteristics, which complicates reliable replication and fair comparison of results. To mitigate these issues, Henderson et al. (2019) recommends systematic use of statistical significance testing (e.g., two-sample t-tests, Kolmogorov–Smirnov tests), together with bootstrap confidence intervals and bootstrap power analysis, as well as comprehensive experimental reporting. In this context, a compact and interpretable representation of hybrid RL+OSC controller behaviour, such as a PCA fingerprint, can support training-time monitoring and policy comparison across experimental settings, thereby strengthening transparency and reproducibility.

5.4.1 Method: PCA-based diagnostic for RL-augmented OSC

A PCA-based methodology is investigated as a diagnostic tool to obtain a compact representation of how a reinforcement learning policy modifies an operational-space controller (OSC) during a continuous-contact episode. Following the approach in Bajrami et al. (2025c), the objective is to construct a behavioural "fingerprint" of the controller from the score trajectory in the plane spanned by the first two principal components.

The procedure, applied to a single episode, is summarised as follows:

1. control- and contact-related signals are logged over the full episode;
2. a feature matrix $\mathbf{X} \in \mathbb{R}^{n\times 21}$ is constructed by stacking:
   - the six Cartesian stiffness gains of the OSC in task space, $k_{p,i}$;
   - the six diagonal terms of the operational-space inertia matrix in task space Λ_i^{diag};
   - the three components of the external force at the tool, $f_j^{\text{ext}} \in \mathbb{R}^3$, including both normal and tangential components.
3. features are standardised via z-score normalisation to remove differences in scale and physical units;
4. PCA is computed and the time sequence is projected onto the first two principal components;

5. the time-ordered trajectory of the resulting scores in the PC1–PC2 plane is taken as the episode-level behavioural fingerprint of the controller.

The working hypothesis is that a targeted RL policy acting along the contact direction (e.g., adapting stiffness, or stiffness plus damping, along z) preserves the dominant coordination structure of the baseline OSC. Under this assumption, differences between fingerprints are expected to appear primarily as similarity transformations in the score plane (translation, rotation, and uniform rescaling), whereas pronounced changes in the geometry or topology of the PC1–PC2 trajectory are interpreted as indicators of behavioural drift or training failure.

Controllers and logged features Three controller configurations are analysed: (i) *OSC*, baseline operational-space control with constant Cartesian stiffness and damping characteristics; (ii) *KZ*, where an RL policy adapts the task-space stiffness along the world-Z axis; and (iii) *KzDz*, where the policy co-adapts stiffness and damping along world-Z under safety bounds.

For each episode, PCA is applied to a feature matrix $\mathbf{X} \in \mathbb{R}^{n\times 21}$ obtained by concatenating:

- task-space stiffness gains $k_{p,i}$, $i = 0,\ldots,5$;
- task-space damping ratios ζ_i, $i = 0,\ldots,5$;
- diagonal terms of the operational-space inertia matrix Λ_i^{diag}, $i = 0,\ldots,5$;
- external force at the tool f_j^{ext}, $j = 0,1,2$.

The axis ordering for both k_p and ζ is $[X,Y,Z,R_x,R_y,R_z]$. The external force is a 3D force $[F_x;F_y;F_z]$ expressed in the world frame; in the considered setup the contact surface is horizontal in the world frame, hence the contact normal coincides with world-Z, and f_2^{ext} corresponds to the normal force component. Each feature is standardised via z-score normalisation *within the episode* (fit on the single episode) prior to PCA.

Interpreting the dominant components via loadings To interpret the principal directions, the PCA loadings are inspected. Table 5.5 reports the explained variance of PC1–PC2 together with the top-5 loadings (by absolute value) for each controller configuration.

Table 5.5 Explained variance and top-5 loadings for PC1 and PC2.

Run	PC1(%)	PC2(%)	PC1 top-5 loadings	PC2 top-5 loadings
OSC	56.1	20.1	$\Lambda_2^{\mathrm{diag}}$ (+0.410) $\Lambda_3^{\mathrm{diag}}$ (+0.407) $\Lambda_0^{\mathrm{diag}}$ (−0.362) $\Lambda_4^{\mathrm{diag}}$ (−0.356) f_2^{ext} (−0.343)	f_1^{ext} (+0.678) f_0^{ext} (+0.661) $\Lambda_4^{\mathrm{diag}}$ (+0.205) $\Lambda_5^{\mathrm{diag}}$ (+0.163) $\Lambda_1^{\mathrm{diag}}$ (+0.152)
K_z	52.2	16.6	$\Lambda_3^{\mathrm{diag}}$ (+0.402) $\Lambda_2^{\mathrm{diag}}$ (+0.396) $\Lambda_4^{\mathrm{diag}}$ (−0.371) $\Lambda_0^{\mathrm{diag}}$ (−0.354) f_2^{ext} (−0.340)	f_1^{ext} (+0.657) f_0^{ext} (+0.648) $\Lambda_4^{\mathrm{diag}}$ (+0.226) $\Lambda_1^{\mathrm{diag}}$ (+0.211) $\Lambda_5^{\mathrm{diag}}$ (+0.181)
K_zD_z	51.0	17.3	$\Lambda_2^{\mathrm{diag}}$ (+0.404) $\Lambda_3^{\mathrm{diag}}$ (+0.390) $\Lambda_0^{\mathrm{diag}}$ (−0.387) $\Lambda_1^{\mathrm{diag}}$ (−0.355) $\Lambda_4^{\mathrm{diag}}$ (−0.309)	f_0^{ext} (+0.552) f_1^{ext} (+0.544) $k_{p,2}$ (+0.420) $\Lambda_4^{\mathrm{diag}}$ (+0.275) $\Lambda_1^{\mathrm{diag}}$ (+0.215)

Across all runs, PC1 is primarily associated with the operational-space inertia terms Λ^{diag}, with a non-negligible contribution from the normal force component f_2^{ext} in the OSC and K_z cases. By contrast, PC2 is dominated by the tangential force components f_0^{ext} and f_1^{ext}, consistent with variations in lateral contact interactions during the polishing motion. In the K_zD_z configuration, the stiffness along world-Z ($k_{p,2}$) becomes a prominent contributor to PC2, reflecting the additional adaptation channel along the contact normal.

Score trajectories as controller fingerprints Let $\mathbf{S} \in \mathbb{R}^{n\times 2}$ denote the PC1–PC2 score sequence obtained by projecting the episode data onto the first two principal components. The time-ordered trajectory of $\mathbf{S}$ in the PC1–PC2 plane is used as an episode-level behavioural fingerprint.

Figure 5.8 reports one representative evaluation episode per controller. The same procedure applied to additional evaluation rollouts yields qualitatively consistent PC1–PC2 trajectory geometries, supporting the interpretation of the score trajectory as a repeatable fingerprint under the considered operating conditions.

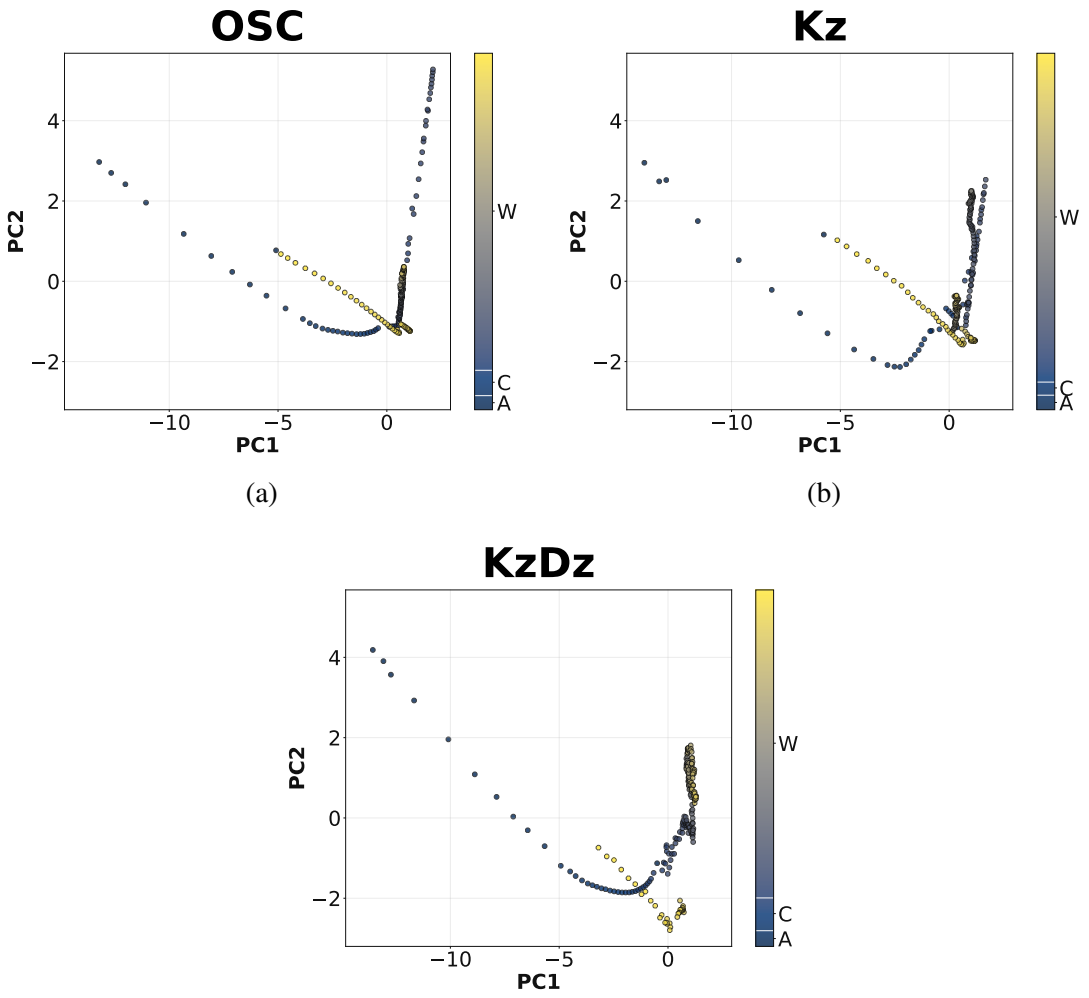


Figure 5.8 PC1–PC2 score trajectories for OSC, K_z, and K_zD_z. Episode phase: Approach (A), Contact (C), and Working (W).

Quantifying trajectory drift: scaled Procrustes alignment in the PC1–PC2 plane
To complement the qualitative comparison of the PC1–PC2 score trajectories, a scaled Procrustes analysis is used to quantify how much the *shape* of the trajectory changes with respect to the baseline controller. Let $\mathbf{S}_0 \in \mathbb{R}^{n_0 \times 2}$ denote the OSC score sequence in the PC1–PC2 plane and $\mathbf{S}_1 \in \mathbb{R}^{n_1 \times 2}$ the corresponding sequence for a target run (K_z or K_zD_z). The comparison is performed as follows:

1. the two sequences are truncated to a common length $n = \min(n_0, n_1)$ by retaining the first n samples;

2. only PC1–PC2 scores are considered;

3. scores are z-scored *within each run* (per component, using mean and standard deviation computed over the n samples);

4. an optimal similarity transform (translation, rotation, and uniform scaling) aligning $\mathbf{S}_1$ to $\mathbf{S}_0$ is computed; the residual mismatch defines the Procrustes distance d_{proc}.

Using this procedure, the distances with respect to OSC are $d_{\text{proc}} \approx 435.55$ for K_z (scale ≈ 308.60, $n = 246$) and $d_{\text{proc}} \approx 431.14$ for $K_z D_z$ (scale ≈ 305.47, $n = 251$). Since translation, rotation, and uniform rescaling are explicitly factored out, d_{proc} should be interpreted as a *shape/topology* indicator of the PC1–PC2 trajectory rather than a measure of absolute operating-point differences.

In the considered evaluation runs, K_z and $K_z D_z$ yield comparable Procrustes distances, suggesting that the RL-augmented controllers preserve a similar PC1–PC2 trajectory geometry to the OSC baseline after normalisation and similarity alignment. In other words, the dominant coordination pattern captured by the first two principal components appears largely conserved, while the main differences observed in the score plots are consistent with changes that can be accounted for by similarity transformations. Finally, it is noted that the present implementation uses a simple truncation to $n = \min(n_0, n_1)$ (first n samples) without phase-based alignment; therefore, d_{proc} is used here as a qualitative comparison signal rather than as a standalone diagnostic threshold.

Chapter 6

Conclusion

"Look again at that dot. That's here.
That's home. That's us."

Carl Sagan, *Pale Blue Dot* (1994).

This thesis tackled a central problem in modern robotic manipulation: how to reliably transfer human skills to a robot and, at the same time, improve those skills through interaction with the environment, especially when the task involves contact and the regulation of physical interaction. The guiding idea was not to pit classical control against learning, but to integrate them into a coherent pipeline: human demonstrations provide structure and intent, while reinforcement learning (RL) acts as an adaptation mechanism on selected variables, increasing robustness and repeatability in scenarios affected by physical variability.

A cross-cutting contribution of this work is the release of *two public, open-source codebases* on GitHub, designed to share reusable tools for both *training* and *experimental analysis*. The two codebases reflect the same layered architecture (environment, control, trajectory/phase management, sensing and reward design, logging) and are provided in two complementary variants: `VTT_RL_fixed_public` and `VTT_RL_public` Bajrami (2026a,b). `VTT_RL_fixed_public` is oriented toward a procedural trajectory with phase management and operational-space control with modulated impedance, whereas `VTT_RL_public` focuses on reproducing demonstration-learned trajectories through a discrete task-space action space executed via IK. In both, a logging and diagnostic subsystem enables structured recording of key signals (states, forces, impedance

parameters, phase transitions, and reward components), supporting comparable and reproducible offline analyses.

From a methodological standpoint, the thesis shows how an "intentional" trajectory representation (waypoints and progression) can serve as an effective bridge between Learning from Demonstration (LfD) and RL-based optimization. Within the simulated polishing framework, this design choice makes it possible to clearly separate what is learned from what remains constrained by control: low-level tracking relies on established schemes (OSC or IK+PD), while learning operates on compact yet interaction-relevant levers, making training more stable and outcomes more interpretable.

The presented results provide an initial body of evidence supporting this approach. In polishing with friction randomization, introducing the learned policy for impedance modulation improves robustness compared to the reference OSC controller, with particularly reliable behavior when adaptation is limited to stiffness along the normal direction. In parallel, in the *Trajectory by Learning from Demonstration* mode (free-space tracking with discrete Cartesian actions), the policy completes the training trajectory with high reliability and, although the completion rate decreases on the longer test trajectory, it typically maintains high coverage of the demonstrated sequence. This suggests that the remaining difficulty is mainly due to time-budget constraints and the increased geometric complexity of the test demonstration, rather than to local control instabilities.

Overall, the contribution of this thesis is twofold. On the one hand, it proposes and validates a pragmatic strategy to "fuse" LfD and RL for contact-rich manipulation tasks, maintaining a clear separation of roles between control and learning and prioritizing observable and diagnosable mechanisms. On the other hand, it provides the community with two open-source software tools intended to accelerate experimentation, support comparisons across design choices (action spaces, rewards, sensing, logging), and improve the reproducibility of results.

Natural directions for extension, already implied by the thesis framework, include completing the integration between tracking and in-contact behavior across the full operational cycle (approach, engagement, contact, release), validating the pipeline on real robots, and expanding, in a controlled manner, the set of variables adapted by the policy (for instance including tangential components or rotations), while preserving safety and interpretability constraints. In this respect, the public availability of the two

codebases is not only a project output, but also a practical enabler for rapid iteration on training and analysis, consolidating the preliminary results into broader and more generalizable evidence.

References

Polaris vega: Advanced optical navigation for oems. https://www.ndigital.com/optical-navigation-technology/polaris-vega/, 2024. Accessed: 2025-12-06.

Pieter Abbeel, Adam Coates, and Andrew Y Ng. Autonomous helicopter aerobatics through apprenticeship learning. *The International Journal of Robotics Research*, 29(13):1608–1639, 2010.

Ahmed Abouelazm, Jonas Michel, and J. Marius Zöllner. A review of reward functions for reinforcement learning in the context of autonomous driving. In *2024 IEEE Intelligent Vehicles Symposium (IV)*, pages 156–163, 2024. doi: 10.1109/IV55156.2024.10588385.

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 22–31. PMLR, 2017. URL https://proceedings.mlr.press/v70/achiam17a.html.

Mohammed Alshiekh, Roderick Bloem, Ruediger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. Safe reinforcement learning via shielding. *CoRR*, abs/1708.08611, 2017. doi: 10.48550/arXiv.1708.08611. URL https://arxiv.org/abs/1708.08611.

Ingy El Sayed Aly and Lu Feng. Logic-based reward shaping for multi-agent reinforcement learning, 2022. URL https://arxiv.org/abs/2206.08881.

Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38, 2017. doi: 10.1109/MSP.2017.2743240. URL https://doi.org/10.1109/MSP.2017.2743240.

Karl Johan Åström and Tore Hägglund. *PID Controllers: Theory, Design, and Tuning*. International Society for Measurement and Control, Research Triangle Park, NC, 2 edition, 1995. ISBN 9781556175169.

Karl Johan Åström and Tore Hägglund. The future of PID control. *Control Engineering Practice*, 9(11):1163–1175, 2001. doi: 10.1016/S0967-0661(01)00062-4. URL https://doi.org/10.1016/S0967-0661(01)00062-4.

Albin Bajrami. Vtt_rl_fixed_public. GitHub repository, 2026a. URL https://github.com/AlbinEV/VTT_RL_PT.

Albin Bajrami. Vtt_rl_public. GitHub repository, 2026b. URL https://github.com/AlbinEV/discrete-lfd-isaac-lab.

Albin Bajrami, Daniele Costa, Matteo Claudio Palpacelli, and Federico Emiliani. Investigating collaborative robotic assembly: A case study of the fanuc crx-10 ia/l in industrial automation at i-labs. *Eng*, 5(2):532–543, 2024. doi: 10.3390/eng5020029. URL https://doi.org/10.3390/eng5020029.

Albin Bajrami, Gloria Beraldo, Matteo Claudio Palpacelli, Tapio Heikkilä, and Gabriella Cortellessa. Markerless upper limb motion tracking: A comparative evaluation of multi-view approaches. Manuscript; presented on 22 October 2025. Not yet available online., October 2025a.

Albin Bajrami, Gloria Beraldo, Matteo Claudio Palpacelli, Tapio Heikkilä, and Gabriella Cortellessa. Markerless upper limb motion tracking: A comparative evaluation of multi-view approaches. MetroXRainen project technical report, 2025b. Under review / in preparation. Update venue and DOI when available.

Albin Bajrami, Markku Suomalainen, Tapio Heikkilä, and Matteo Claudio Palpacelli. PCA fingerprint for RL-augmented operational-space control. In *IROS 2025 Workshop on Contact and Impact-aware Manipulation*, 2025c. URL https://openreview.net/forum?id=M2g5GhwTk1. Workshop paper.

Michał Bednarek and Krzysztof Walas. Comparative assessment of reinforcement learning algorithms in the task of robotic manipulation of deformable linear objects. In *2019 4th International Conference on Robotics and Automation Engineering (ICRAE)*, pages 173–177. IEEE, 2019. doi: 10.1109/ICRAE48301.2019.9043790. URL https://doi.org/10.1109/ICRAE48301.2019.9043790.

Boris Belousov, Bastian Wibranek, Jan Schneider, Tim Schneider, Georgia Chalvatzaki, Jan Peters, and Oliver Tessmann. Robotic architectural assembly with tactile skills: Simulation and optimization. *Automation in Construction*, 133:104006, 2022. doi: 10.1016/j.autcon.2021.104006. URL https://doi.org/10.1016/j.autcon.2021.104006.

Aude Billard, Sylvain Calinon, Rüdiger Dillmann, and Stefan Schaal. *Robot Programming by Demonstration*, pages 1371–1394. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. ISBN 978-3-540-30301-5. doi: 10.1007/978-3-540-30301-5_60. URL https://doi.org/10.1007/978-3-540-30301-5_60.

Oleksandr Bogdan, Viktor Eckstein, François Rameau, and Jean-Charles Bazin. Deepcalib: A deep learning approach for automatic intrinsic calibration of wide field-of-view cameras. In *Proceedings of the 15th ACM SIGGRAPH European Conference on Visual Media Production (CVMP)*, 2018. doi: 10.1145/3278471.3278479.

Xavier Brun, Daniel Thomasset, and Serge Scavarda. Hybrid control for switching between position and force tracking. 2003. URL https://api.semanticscholar.org/CorpusID:215918785.

Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7291–7299, 2017. doi: 10.1109/CVPR.2017.143. URL https://doi.org/10.1109/CVPR.2017.143.

Sonia Chernova and Andrea L Thomaz. *Robot learning from human teachers*. Springer Nature, 2022.

Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 4693–4700. IEEE, 2018.

Michele Conconi, Alessandro Pompili, Nicola Sancisi, and Vincenzo Parenti-Castelli. Quantification of the errors associated with marker occlusion in stereophotogrammetric systems and implications on gait analysis. *Journal of Biomechanics*, 114: 110162, 2021. doi: 10.1016/j.jbiomech.2020.110162. URL https://doi.org/10.1016/j.jbiomech.2020.110162.

Emma Cramer, Lukas Jäschke, and Sebastian Trimpe. Cheq-ing the box: Safe variable impedance learning for robotic polishing. *IFAC-PapersOnLine*, 59(18):325–330, 2025. doi: 10.1016/j.ifacol.2025.10.241. URL https://doi.org/10.1016/j.ifacol.2025.10.241.

Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerík, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe exploration in continuous action spaces. *CoRR*, abs/1801.08757, 2018. URL https://arxiv.org/abs/1801.08757.

Todor Davchev, Kevin Sebastian Luck, Michael Burke, Franziska Meier, Stefan Schaal, and Subramanian Ramamoorthy. Residual learning from demonstration: Adapting dmps for contact-rich manipulation. *IEEE Robotics and Automation Letters*, 7(2): 4488–4495, April 2022. ISSN 2377-3774. doi: 10.1109/lra.2022.3150024. URL http://dx.doi.org/10.1109/LRA.2022.3150024.

Rati Devidze. Reward design for reinforcement learning agents, 2025. URL https://arxiv.org/abs/2503.21949.

Rüdiger Dillmann. Teaching and learning of robot tasks via observation of human performance. *Robotics and Autonomous Systems*, 47(2-3):109–116, 2004.

Yufeng Ding, JunChao Zhao, and X. Min. Impedance control and parameter optimization of surface polishing robot based on reinforcement learning. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 237(1-2):216–228, 2023. doi: 10.1177/09544054221100004. URL https://doi.org/10.1177/09544054221100004.

Robert Elfring, Matías de la Fuente, and Klaus Radermacher. Assessment of optical localizer accuracy for computer aided surgery systems. *Computer Aided Surgery*, 15(1-3):1–12, 2010. doi: 10.3109/10929081003647239. URL https://doi.org/10.3109/10929081003647239.

Íñigo Elguea-Aguinaco, Antonio Serrano-Muñoz, Dimitrios Chrysostomou, Ibai Inziarte-Hidalgo, Simon Bøgh, and Nestor Arana-Arexolaleiba. A review on reinforcement learning for contact-rich robotic manipulation tasks. *Robotics and Computer-Integrated Manufacturing*, 81:102517, 2023.

Franka Robotics GmbH. *Franka Research 3 – Datasheet*, 2025. URL https://franka.de/hubfs/Datasheet%20Franka%20Research%203_R02212_2.4_EN.pdf. Accessed: 2026-01-17.

Javier García and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(42):1437–1480, 2015. URL https://jmlr.org/papers/v16/garcia15a.html.

Sergio Garrido-Jurado, Rafael Muñoz-Salinas, Francisco J. Madrid-Cuevas, and Manuel J. Marín-Jiménez. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47(6):2280–2292, 2014. doi: 10.1016/j.patcog.2014.01.005.

Sara Hamdan, Yusuf Aydin, Erhan Oztop, and Cagatay Basdogan. Robotic learning of haptic skills from expert demonstration for contact-rich manufacturing tasks. In *2024 IEEE 20th International Conference on Automation Science and Engineering (CASE)*, pages 2334–2341, 2024. doi: 10.1109/CASE59546.2024.10711473.

Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2 edition, 2004. doi: 10.1017/CBO9780511811685.

Richard I. Hartley and Peter Sturm. Triangulation. *Computer Vision and Image Understanding*, 68(2):146–157, 1997. ISSN 1077-3142. doi: https://doi.org/10.1006/cviu.1997.0547. URL https://www.sciencedirect.com/science/article/pii/S1077314297905476.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), Apr. 2018. doi: 10.1609/aaai.v32i1.11694. URL https://ojs.aaai.org/index.php/AAAI/article/view/11694.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters, 2019. URL https://arxiv.org/abs/1709.06560.

Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Ian Osband, Gabriel Dulac-Arnold, John Agapiou, Joel Z. Leibo, and Audrunas Gruslys. Deep q-learning from demonstrations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), 2018. doi: 10.1609/aaai.v32i1.11757. URL https://doi.org/10.1609/aaai.v32i1.11757.

Neville Hogan. Impedance control: An approach to manipulation: Part i—theory. *Journal of Dynamic Systems, Measurement, and Control*, 107(1):1–7, 03 1985. ISSN 0022-0434. doi: 10.1115/1.3140702. URL https://doi.org/10.1115/1.3140702.

Marco Hutter. Robot dynamics lecture notes. https://ethz.ch/content/dam/ethz/special-interest/mavt/robotics-n-intelligent-systems/rsl-dam/documents/RobotDynamics2017/RD_HS2017script.pdf, 2017.

Sinan Ibrahim, Mostafa Mostafa, Ali Jnadi, Hadi Salloum, and Pavel Osinenko. Comprehensive overview of reward engineering and shaping in advancing reinforcement learning applications. *IEEE Access*, 12:175473–175500, 2024. doi: 10.1109/ACCESS.2024.3504735.

Ganesh Iyer, R. Karnik Ram, J. Krishna Murthy, and K. Madhava Krishna. Calibnet: Geometrically supervised extrinsic calibration using 3d spatial transformer networks. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1110–1117, 2018. doi: 10.1109/IROS.2018.8593693. URL https://arxiv.org/abs/1803.08181.

Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, and Sergey Levine. QT-Opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *CoRR*, abs/1806.10293, 2018. URL https://arxiv.org/abs/1806.10293.

Manabu Kano, Shinji Hasebe, Iori Hashimoto, and Hiromu Ohno. A new multivariate statistical process monitoring method using principal component analysis. *Computers & chemical engineering*, 25(7-8):1103–1113, 2001.

Arvid QL Keemink, Herman Van der Kooij, and Arno HA Stienen. Admittance control for physical human–robot interaction. *The International Journal of Robotics Research*, 37(11):1421–1444, 2018.

R. Khadem, C. C. Yeh, M. Sadeghi-Tehrani, M. R. Bax, J. A. Johnson, J. N. Welch, E. P. Wilkinson, and R. Shahidi. Comparative tracking error analysis of five different optical tracking systems. *Computer Aided Surgery*, 5(2):98–107, 2000. doi: 10.1002/1097-0150(2000)5:2<98::AID-IGS4>3.0.CO;2-H. URL https://doi.org/10.1002/1097-0150(2000)5:2<98::AID-IGS4>3.0.CO;2-H.

Oussama Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal of Robotics and Automation*, 3(1):43–53, February 1987. doi: 10.1109/JRA.1987.1087068. URL https://khatib.stanford.edu/publications/pdfs/Khatib_1987_RA.pdf.

Cheng-Yu Kuo, Andreas Schaarschmidt, Yunduan Cui, Tamim Asfour, and Takamitsu Matsubara. Uncertainty-aware contact-safe model-based reinforcement learning.

IEEE Robotics and Automation Letters, 6(2):3918–3925, 2021. doi: 10.1109/LRA.2021.3065271. URL https://doi.org/10.1109/LRA.2021.3065271.

Minae Kwon, Sang Michael Xie, Kalesha Bullard, and Dorsa Sadigh. Reward design with language models, 2023. URL https://arxiv.org/abs/2303.00001.

Jong-Min Lee, ChangKyoo Yoo, Sang Wook Choi, Peter A Vanrolleghem, and In-Beum Lee. Nonlinear process monitoring using kernel principal component analysis. *Chemical engineering science*, 59(1):223–234, 2004.

Sergey Levine and Vladlen Koltun. Guided policy search. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1–9, Atlanta, Georgia, USA, June 2013. PMLR. URL https://proceedings.mlr.press/v28/levine13.html.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(39):1–40, 2016a. URL https://jmlr.org/papers/v17/15-522.html.

Sergey Levine, Peter Pastor, Alex Krizhevsky, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *CoRR*, abs/1603.02199, 2016b. URL https://arxiv.org/abs/1603.02199.

Thomas Lew, Sumeet Singh, Mario Prats, Jeffrey Bingham, Jonathan Weisz, Benjie Holson, Xiaohan Zhang, Vikas Sindhwani, Yao Lu, Fei Xia, et al. Robotic table wiping via reinforcement learning and whole-body trajectory optimization. *arXiv preprint arXiv:2210.10865*, 2022.

Yun Li, Kiam Heong Ang, and Gregory C. Y. Chong. PID control system analysis and design: Problems, remedies, and future directions. *IEEE Control Systems Magazine*, 26(1):32–41, 2006. doi: 10.1109/MCS.2006.1580152. URL https://doi.org/10.1109/MCS.2006.1580152.

Xing Liu, Shuzhi Sam Ge, Fei Zhao, and Xuesong Mei. Optimized interaction control for robot manipulator interacting with flexible environment. *IEEE/ASME Transactions on Mechatronics*, 26(6):2888–2898, 2021. doi: 10.1109/TMECH.2020.3047919. URL https://doi.org/10.1109/TMECH.2020.3047919. Early access: 2020.

Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, Wan-Teh Chang, Wei Hua, Manfred Georg, and Matthias Grundmann. Mediapipe: A framework for building perception pipelines, 2019. URL https://arxiv.org/abs/1906.08172.

Jianlan Luo, Eugen Solowjow, Chengtao Wen, Juan Aparicio Ojea, and Alice M. Agogino. Deep reinforcement learning for robotic assembly of mixed deformable and rigid objects. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2062–2069. IEEE, 2018. doi: 10.1109/IROS.2018.8594353. URL https://doi.org/10.1109/IROS.2018.8594353.

Jianlan Luo, Eugen Solowjow, Chengtao Wen, Juan Aparicio Ojea, Alice M. Agogino, Aviv Tamar, and Pieter Abbeel. Reinforcement learning on variable impedance controller for high-precision robotic assembly. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3080–3087. IEEE, 2019. doi: 10.1109/ICRA.2019.8793506. URL https://doi.org/10.1109/ICRA.2019.8793506.

Guilherme J Maeda, Gerhard Neumann, Marco Ewerton, Rudolf Lioutikov, Oliver Kroemer, and Jan Peters. Probabilistic movement primitives for coordination of multiple human–robot collaborative tasks. *Autonomous Robots*, 41(3):593–612, 2017.

Roberto Martín-Martín, Michelle A. Lee, Rachel Gardner, Silvio Savarese, Jeannette Bohg, and Animesh Garg. Variable impedance control in end-effector space: An action space for reinforcement learning in contact-rich tasks. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1010–1017. IEEE, 2019. doi: 10.1109/IROS40897.2019.8968201. URL https://doi.org/10.1109/IROS40897.2019.8968201.

Pierre Merriaux, Yohan Dupuis, Rémi Boutteau, Pascal Vasseur, and Xavier Savatier. A study of vicon system positioning performance. *Sensors*, 17(7):1591, 2017. doi: 10.3390/s17071591. URL https://doi.org/10.3390/s17071591.

James Mitchelson and Adrian Hilton. Wand-based multiple camera studio calibration. Technical Report VSSP-TR-2/2003, University of Surrey, Centre for Vision, Speech and Signal Processing (CVSSP), 2003. URL https://personal.ee.surrey.ac.uk/Personal/A.Hilton/papers/WandCalibTR.pdf.

Mayank Mittal, Calvin Yu, Qinxi Yu, Jingzhou Liu, Nikita Rudin, David Hoeller, Jia Lin Yuan, Ritvik Singh, Yunrong Guo, Hammad Mazhar, Ajay Mandlekar, Buck Babich, Gavriel State, Marco Hutter, and Animesh Garg. Orbit: A unified simulation framework for interactive robot learning environments. *IEEE Robotics and Automation Letters*, 8(6):3740–3747, 2023. doi: 10.1109/LRA.2023.3270034.

Thomas B. Moeslund, Adrian Hilton, and Volker Krüger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(2-3):90–126, 2006. doi: 10.1016/j.cviu.2006.08.002. URL https://doi.org/10.1016/j.cviu.2006.08.002.

Anahita Mohseni-Kabir, Charles Rich, Sonia Chernova, Candace L Sidner, and Daniel Miller. Interactive hierarchical task learning from a single demonstration. In *Proceedings of the tenth annual ACM/IEEE international conference on human-robot interaction*, pages 205–212, 2015.

MRPT Team. Application: camera-calib — mrpt documentation. https://docs.mrpt.org/reference/latest/app_camera-calib.html, 2024. Accessed: 2024-01-01.

Mohammad Nabhan, Yajun Mei, and Jianjun Shi. High dimensional process monitoring using robust sparse probabilistic principal component analysis. *arXiv preprint arXiv:1904.09514*, 2019.

Bojan Nemec, Leon Žlajpah, and Aleš Ude. Door opening by joining reinforcement learning and intelligent control. In *2017 18th International Conference on Advanced Robotics (ICAR)*, pages 222–228. IEEE, 2017. doi: 10.1109/ICAR.2017.8023522. URL https://doi.org/10.1109/ICAR.2017.8023522.

Edwin Olson. Apriltag: A robust and flexible visual fiducial system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3400–3407, 2011. doi: 10.1109/ICRA.2011.5979561. URL https://april.eecs.umich.edu/media/pdfs/olson2011tags.pdf.

OpenCV Contributors. Charuco boards for camera calibration (opencv documentation). https://docs.opencv.org/. Accessed: 2026-01-15.

Frank C. Park and Bryan J. Martin. Robot sensor calibration: Solving ax = xb on the euclidean group. *IEEE Transactions on Robotics and Automation*, 10(5):717–721, 1994. doi: 10.1109/70.326576.

Sahba Aghajani Pedram, Peter W. Ferguson, Changyeob Shin, Ankur Mehta, Erik P. Dutson, Farshid Alambeigi, and Jacob Rosen. Toward synergic learning for autonomous manipulation of deformable tissues via surgical robots: An approximate q-learning approach. In *2020 8th IEEE RAS/EMBS International Conference for Biomedical Robotics and Biomechatronics (BioRob)*, pages 878–884. IEEE, 2020. doi: 10.1109/BioRob49111.2020.9224421. URL https://doi.org/10.1109/BioRob49111.2020.9224421.

Affan Pervez and Dongheui Lee. Learning task-parameterized dynamic movement primitives using mixture of gmms. *Intelligent Service Robotics*, 11(1):61–78, 2018.

Richard Alan Peters, Christina L Campbell, William J Bluethmann, and Eric Huber. Robonaut task learning through teleoperation. In *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, volume 2, pages 2806–2811. IEEE, 2003.

Vladimír Petrík and Ville Kyrki. Feedback-based fabric strip folding. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 773–778. IEEE, 2019. doi: 10.1109/IROS40897.2019.8967657. URL https://doi.org/10.1109/IROS40897.2019.8967657.

Marc H Raibert and John J Craig. Hybrid position/force control of manipulators. *Journal of dynamic systems, measurement, and control*, 103(2):126–133, 1981.

Harish Ravichandar, Athanasios S Polydoros, Sonia Chernova, and Aude Billard. Robot learning from demonstration: A review of recent advances. *Annual Review of Control, Robotics, and Autonomous Systems*, 2019.

Stefan Schaal. Is imitation learning the route to humanoid robots? *Trends in cognitive sciences*, 3(6):233–242, 1999.

Gerrit Schoettler, Ashvin Nair, Juan Aparicio Ojea, Sergey Levine, and Eugen Solowjow. Meta-reinforcement learning for robotic industrial insertion tasks, 2020. URL https://arxiv.org/abs/2004.14404.

John Schulman, Jonathan Ho, Cameron Lee, and Pieter Abbeel. Learning from demonstrations through the use of non-rigid registration. In *Robotics Research: The 16th International Symposium ISRR*, pages 339–354. Springer, 2016.

Marie Schumacher, Janis Wojtusch, Philipp Beckerle, and Oskar von Stryk. An introductory review of active compliant control. *Robotics and Autonomous Systems*, 119:185–200, 2019. ISSN 0921-8890. doi: https://doi.org/10.1016/j.robot.2019.06.009. URL https://www.sciencedirect.com/science/article/pii/S0921889018307772.

Antonio Serrano-Muñoz, Nestor Arana-Arexolaleiba, Dimitrios Chrysostomou, and Simon Bøgh. Learning and generalising object extraction skill for contact-rich disassembly tasks: an introductory study. *The International Journal of Advanced Manufacturing Technology*, 124:3171–3183, 2023. doi: 10.1007/s00170-021-08086-z. URL https://doi.org/10.1007/s00170-021-08086-z.

Yonadav Shavit, Nadia Figueroa, Seyed Sina Mirrazavi Salehian, and Aude Billard. Learning augmented joint-space task-oriented dynamical systems: a linear parameter varying and synergetic control approach. *IEEE Robotics and Automation Letters*, 3 (3):2718–2725, 2018.

Yunlei Shi, Zhaopeng Chen, Yansong Wu, Dimitri Henkel, Sebastian Riedel, Hongxu Liu, Qian Feng, and Jianwei Zhang. Combining learning from demonstration with learning by exploration to facilitate contact-rich tasks. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1062–1069. IEEE, 2021.

Changyeob Shin, Peter W. Ferguson, Sahba A. Pedram, Jianan Ma, Erik P. Dutson, and Jacob Rosen. Autonomous tissue manipulation via surgical robot using learning based model predictive control. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3875–3881. IEEE, 2019. doi: 10.1109/ICRA.2019.8794159. URL https://doi.org/10.1109/ICRA.2019.8794159.

Ki Young Shin and Joung Hwan Mun. A multi-camera calibration method using a 3-axis frame and wand. *International Journal of Precision Engineering and Manufacturing*, 13:283–289, 2012. doi: 10.1007/s12541-012-0035-1.

Markku Suomalainen, Yiannis Karayiannidis, and Ville Kyrki. A survey of robot manipulation in contact. *Robotics and Autonomous Systems*, 156:104224, 2022. ISSN 0921-8890. doi: https://doi.org/10.1016/j.robot.2022.104224. URL https://www.sciencedirect.com/science/article/pii/S0921889022001312.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, Cambridge, MA, 2 edition, 2018. ISBN 9780262039246. URL https://mitpress.mit.edu/9780262039246/reinforcement-learning/.

Tomáš Svoboda, Daniel Martinec, and Tomáš Pajdla. A convenient multicamera self-calibration for virtual environments. *Presence: Teleoperators and Virtual Environments*, 14(4):407–422, 2005. doi: 10.1162/105474605774785325.

Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer, 2 edition, 2022. doi: 10.1007/978-3-030-34372-9.

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 23–30. IEEE, 2017. doi: 10.1109/IROS.2017.8202133. URL https://doi.org/10.1109/IROS.2017.8202133.

Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)*, pages 4950–4957. International Joint Conferences on Artificial Intelligence Organization, 2018. doi: 10.24963/ijcai.2018/687. URL https://doi.org/10.24963/ijcai.2018/687.

Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. Bundle adjustment — a modern synthesis. 1883:298–372, 2000. doi: 10.1007/3-540-44480-7_21.

Roger Y. Tsai and Reimar K. Lenz. A new technique for fully autonomous and efficient 3d robotics hand/eye calibration. *IEEE Transactions on Robotics and Automation*, 5 (3):345–358, 1989. doi: 10.1109/70.34770. URL https://kmlee.gatech.edu/me6406/handeye.pdf.

Yoshihisa Tsurumine, Yunduan Cui, Eiji Uchibe, and Takamitsu Matsubara. Deep reinforcement learning with smooth policy update: Application to robotic cloth manipulation. *Robotics and Autonomous Systems*, 112:72–83, 2019. doi: 10.1016/j.robot.2018.11.004. URL https://doi.org/10.1016/j.robot.2018.11.004.

Antonio Visioli. *Practical PID Control*. Advances in Industrial Control. Springer London, 2006. doi: 10.1007/1-84628-586-0. URL https://doi.org/10.1007/1-84628-586-0.

David Vogt, Simon Stepputtis, Steve Grehl, Bernhard Jung, and Heni Ben Amor. A system for learning continuous human-robot interactions from human-human demonstrations. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2882–2889. IEEE, 2017.

Yixiang Wang, Yujing Hu, Feng Wu, and Yingfeng Chen. Automatic reward design via learning motivation-consistent intrinsic rewards. *arXiv preprint arXiv:2207.14722*, 2022.

David Whitney, Eric Rosen, Elizabeth Phillips, George Konidaris, and Stefanie Tellex. Comparing robot grasping teleoperation across desktop and virtual reality with ros reality. In *Robotics research: the 18th international symposium ISRR*, pages 335–350. Springer, 2019.

Markus Windolf, Nils Götzen, and Michael Morlock. Systematic accuracy and precision analysis of video motion capturing systems–exemplified on the vicon-460 system. *Journal of Biomechanics*, 41(12):2776–2780, 2008. doi: 10.1016/j.jbiomech.2008.06.024. URL https://doi.org/10.1016/j.jbiomech.2008.06.024.

Haotian Wu, Jianzhong Yang, Si Huang, and Jiahui Li. Adaptive optimal admittance control for robotic precision grinding based on improved normalized advantage function. *IEEE/ASME Transactions on Mechatronics*, pages 1–13, 2025. doi: 10.1109/TMECH.2025.3553496.

Xin Xu, Kun Qian, Aohua Liu, Zhaokun Yue, and Wuling Huang. Polishing via odes: Adaptive admittance control for robot polishing based on neural odes. *Journal of Manufacturing Processes*, 155:428–442, 2025. ISSN 1526-6125. doi: https://doi.org/10.1016/j.jmapro.2025.10.024. URL https://www.sciencedirect.com/science/article/pii/S1526612525011077.

Wanqi Xue, Bo An, Shuicheng Yan, and Zhongwen Xu. Reinforcement learning from diverse human preferences, 2024. URL https://arxiv.org/abs/2301.11774.

Heng Zhang, Rui Dai, Gokhan Solak, Pokuang Zhou, Yu She, and Arash Ajoudani. Safe learning for contact-rich robot tasks: A survey from classical learning-based methods to safe foundation models. *arXiv preprint arXiv:2512.11908*, 2025.

Jingxin Zhang, Donghua Zhou, and Maoyin Chen. Monitoring multimode processes: A modified pca algorithm with continual learning ability. *Journal of Process Control*, 103:76–86, 2021.

Tie Zhang, Meng Xiao, Yanbiao Zou, and Jiadong Xiao. Robotic constant-force grinding control with a press-and-release model and model-based reinforcement learning. *The International Journal of Advanced Manufacturing Technology*, 106 (1-2):589–602, 2020. doi: 10.1007/s00170-019-04614-0. URL https://doi.org/10.1007/s00170-019-04614-0.

Zhengyou Zhang. A flexible new technique for camera calibration. Technical Report MSR-TR-98-71, Microsoft Research, 2000a. URL https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/tr98-71.pdf.

Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000b. doi: 10.1109/34.888718.

Weichao Zhou and Wenchao Li. Programmatic reward design by example. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(8):9233–9241, Jun. 2022. doi: 10.1609/aaai.v36i8.20910. URL https://ojs.aaai.org/index.php/AAAI/article/view/20910.

Appendix A

Franka Research 3 Specifications

This appendix reports the main specifications of the Franka Research 3 arm that are directly used in the polishing framework: overall capabilities, joint limits, and workspace dimensions.

A.1 Overall Specifications

Table A.1 Franka Research 3 arm: overall specifications (subset).

Item	**Value**
Degrees of freedom	7
Rated payload	3 kg
Maximum reach (flange)	855 mm
Force/torque sensing	Link-side torque sensor in all 7 axes
Arm weight	$\sim$ 18.3 kg
Cartesian velocity at TCP	up to 2 m/s
Position repeatability	$< \pm 0.1$ mm (ISO 9283)
Adjustable translational stiffness	10–3000 N/m
Adjustable rotational stiffness	1–300 Nm/rad

A.2 Joint Limits

These limits are the reference for: (i) kinematic bounds on initial configurations and trajectories; (ii) torque and velocity saturation in the controller; (iii) the definition of safe ranges for RL actions.

Table A.2 Joint position, torque, and velocity limits (J1–J7 correspond to A1–A7 in the datasheet).

Joint	**Position range [deg]**	**Torque limit [Nm]**	**Velocity limit [deg/s]**
J1 (A1)	$[-166, 166]$	± 87	150
J2 (A2)	$[-105, 105]$	± 87	150
J3 (A3)	$[-166, 166]$	± 87	150
J4 (A4)	$[-176, -7]$	± 87	150
J5 (A5)	$[-165, 165]$	± 12	301
J6 (A6)	$[25, 265]$	± 12	301*
J7 (A7)	$[-175, 175]$	± 12	301

*For FCI use, the datasheet reports 239 deg/s for joint A6.

Appendix B

Reinforcement Learning Hyperparameters and Network Configuration

This appendix reports the main training hyperparameters and network settings used for the polishing experiments with the RL-Games PPO implementation.

The policy and value networks share a multilayer perceptron (MLP) encoder with the configuration reported in Table B.2. The input dimension corresponds to the flattened observation history $o_t \in \mathbb{R}^{Ln_{\text{step}}}$ defined in Section 4.4.2.

Table B.1 PPO training hyperparameters used in the polishing experiments.

Parameter	Symbol / key	Value
Discount factor	γ	0.99
GAE parameter	τ	0.95
Learning rate (initial)	η	5.0×10^{-5}
Learning-rate schedule	`lr_schedule`	adaptive (standard)
PPO clipping parameter	$\varepsilon_{\text{clip}}$	0.2
KL threshold	`kl_threshold`	6.0×10^{-3}
Entropy coefficient	`entropy_coef`	2.0×10^{-3}
Critic loss coefficient	`critic_coef`	2.0
Gradient-norm clipping	`grad_norm`	1.0
Value clipping	`clip_value`	enabled
Reward scaling	`scale_value`	1.0
Rollout horizon	`horizon_length`	256 steps
Sequence length (obs buffer)	`seq_length`	128 steps
Mini-batch size	`minibatch_size`	256
PPO epochs per update	`mini_epochs`	3
Number of parallel environments	N_{env}	32–64 (typical)
Maximum training epochs	`max_epochs`	200
Observation normalisation	`normalize_input`	enabled
Value normalisation	`normalize_value`	enabled
Advantage normalisation	`normalize_advantage`	enabled
Mixed precision	`mixed_precision`	enabled
Action bounds loss	`bounds_loss_coef`	1.0×10^{-4}

Table B.2 Neural network architecture used for policy and value function.

Component	Key	Value
Network type	`network.name`	`actor_critic`
Shared encoder	`separate`	`False` (shared policy/value)
Input size	`mlp.input_size`	Ln_{step} (e.g. 3456)
Hidden units (layer 1)	`mlp.units[0]`	512
Hidden units (layer 2)	`mlp.units[1]`	128
Hidden units (layer 3)	`mlp.units[2]`	64
Activation function	`mlp.activation`	ELU
Output distribution	`continuous`	Gaussian with fixed log-std
Log-std initialisation	`sigma_init.val`	-0.5
Fixed standard deviation	`fixed_sigma`	`True`

Appendix C

Survey of Reward Design Methods in Reinforcement Learning

The problem of designing reward functions is not new Kaelbling et al. (1996). However, in recent years, driven in part by the widespread adoption of AI-based tools in industrial and service applications, it has attracted renewed attention. As a consequence, the research community is actively working on methods and technologies that are better adapted to these emerging needs. Contemporary trends in *reward design* can be broadly grouped into a few main directions.

A first line of work focuses on *adaptive and interpretable* approaches, in which the reward function is dynamically adjusted as a function of the agent's current policy. The aim is to improve convergence properties while maintaining a clear and interpretable relationship between reward structure and the resulting behaviour Devidze (2025). Rather than relying on a fixed, hand-crafted reward, these methods modify or reweight its components during training to emphasise different aspects of the task as learning progresses.

A second direction concerns *automatic or self-driven* reward construction. Here the goal is to generate intrinsic rewards that are consistent with the agent's internal motivation, thereby reducing the dependence on manually engineered signals Wang et al. (2022). Examples include curiosity-based bonuses, novelty incentives, and self-supervised objectives that encourage the agent to explore or model its environment more effectively, even in the absence of dense task-specific feedback.

A third family of approaches introduces *programmatic or language-based* rewards. In these methods, structured representations (such as logical or programmatic

specifications) or natural language descriptions are used to define objectives and subtasks Kwon et al. (2023); Zhou and Li (2022). The underlying idea is to provide a more direct and human-readable interface between high-level intentions and the scalar signals observed by the agent, thereby facilitating alignment between human goals and learned behaviour.

A fourth line of research is centred on *human-preference-oriented* rewards. Instead of specifying a reward function explicitly, these methods infer it from human judgements, comparisons, or preferences over trajectories and behaviours Xue et al. (2024). This class includes RLHF (Reinforcement Learning from Human Feedback) and related techniques, which aim to ensure that the learned policy is aligned with subjective notions of quality or acceptability that may be hard to encode directly.

Finally, *multi-agent and contextual* reward formulations address scenarios in which multiple agents interact in cooperative or competitive settings, often under safety constraints and multiple priorities Abouelazm et al. (2024); Aly and Feng (2022). In these cases, rewards must capture not only individual task performance but also the effects of interaction, coordination, and conflict between agents, as well as context-dependent requirements such as fairness, safety margins, or resource sharing.

Taken together, these research directions contribute to a new paradigm of *reward design* that is more flexible, interpretable, and adaptive, and that aims to integrate human knowledge, task structure, and the emergent dynamics of the agent. Building on the taxonomies and analytical frameworks proposed in Devidze (2025) and Ibrahim et al. (2024), Table C.1 summarises the main conceptual dimensions along which contemporary reward designs can be categorised, with examples of typical subtypes and associated descriptions.

Table C.1 Overview of the state of the art in the design, modelling, and shaping of reward functions in Reinforcement Learning.

Category	Subtype	Description
Reward source	Human-Provided Rewards	Rewards designed or learned from human feedback, including demonstrations, explicit goals, preferences, ratings, or interactive interventions.
	AI-Generated Rewards	Rewards generated by foundation models (LLMs, VLMs), typically based on semantic similarity or automatically produced feedback consistent with the task objective.
Motivation mechanism	Extrinsic Rewards	Rewards explicitly defined by the task designer to reflect the final objective of the task (e.g., success, performance metrics, constraint satisfaction).
	Intrinsic Rewards	Rewards related to the agent's internal motivation, such as curiosity, surprise, empowerment, or incentives to explore the state space beyond immediate task goals.
Reward-learning paradigms	Learning from Demonstrations	Techniques based on Inverse Reinforcement Learning (IRL) or related methods, which infer reward functions by observing expert demonstrations.
	Learning from Goals	Rewards modelled as the achievement of explicit goal states or of states defined in a latent space (e.g., spatial or temporal distance to a target configuration).
	Learning from Preferences	Reward learning from comparisons between trajectory segments according to human or surrogate preferences; a standard approach in RLHF and related settings.
Reward engineering and shaping	Manual Reward Engineering	Explicit design of the reward function by combining hand-crafted signals (e.g., forward velocity, control penalties, safety margins) with chosen weights.
	Potential-Based Shaping	Modification of a base reward via a potential function $\Phi(s)$, $R'(s,a,s') = R(s,a,s') + \gamma\Phi(s') - \Phi(s)$, which under suitable assumptions preserves the optimal policy.
	Count-Based Exploration	Use of visit counts or visitation estimates to encourage exploration of novel or rarely visited states through additional shaping terms.
	Information Gain / Surprise	Intrinsic shaping based on prediction error, epistemic uncertainty, or changes in an internal dynamics/world model, rewarding informative interactions.
	Semantic Similarity Rewards	Rewards derived from similarity between multimodal embeddings (e.g., text–video, text–image) computed by VLMs or CLIP-like models.
	Preference-Based Shaping	Use of sets of learned reward models with explicit uncertainty (e.g., ensembles) to balance exploration and exploitation under preference-based supervision.
Common issues	Reward Sparsity	Rewards are rare or heavily delayed in time, which slows down learning and makes credit assignment more difficult.
	Reward Hacking	The agent discovers unintended behaviours that maximise the reward while violating the underlying task intent or safety requirements.
	Misalignment	The designed reward does not faithfully capture the true objectives of the task, leading to systematically undesirable or suboptimal behaviours.
	Ambiguity / Noise	Human feedback or sensor-based rewards can be ambiguous or noisy, requiring mechanisms for disambiguation and robustness.
Applications	RLHF, Robotics, Sim-to-Real, Autonomous Control	Dialogue systems, robotic manipulation, complex 3D environments, and multitask agents that require adaptive or learned rewards, often with sim-to-real constraints.

Appendix D

Reward Formulations and Coefficients (Procedural Trajectory Mode)

D.1 Reward Formulations

This appendix provides compact formulations for the reward terms used in the procedural trajectory mode. Parameters and weights are configurable.

Kz-Only Reward $\mathscr{R}_{K_z}$

Descent shaping.

$$r_t^{\text{speed}} = m_t^{\text{DESC}}\ \phi_v(-v_z(t)), \tag{D.1}$$

$$r_t^{\text{time}} = -\,m_t^{\text{DESC}}\ \phi_T\big(T_{\text{desc}}(t)\big), \tag{D.2}$$

$$r_t^{\text{xy}} = m_t^{\text{DESC}}\ \exp\big(-\alpha_{xy}\,\|e_{xy}(t)\|\big), \tag{D.3}$$

$$r_t^{\text{prog}} = m_t^{\text{DESC}}\ \exp\big(-\alpha_z\, d_z(t)\big), \tag{D.4}$$

$$r_t^{\text{contact}} = m_t^{\text{DESC}}\ \mathbf{1}_{\{f_z(t)<F_{\text{TOUCH}}\}}\ \alpha_c, \tag{D.5}$$

$$r_t^{K_z\,\text{prep}} = m_t^{\text{DESC}}\ \exp\Big(-\frac{|K_z^{\text{norm}}(t) - K_z^{\text{prep}}|}{\sigma_K}\Big). \tag{D.6}$$

Ramped contact target.

$$\rho_t = \mathrm{clip}\Big(\frac{n_{\mathrm{cont}}(t)}{N_{\mathrm{ramp}}}, 0, 1\Big), \tag{D.7}$$

$$f_z^\star(t) = f_{z,0} - k_v\,\mathrm{clip}\Big(\frac{\|v_{xy}(t)\|}{v_{\mathrm{ref}}}, 0, 1\Big), \tag{D.8}$$

$$\tilde{f}_z^\star(t) = \rho_t\, f_z^\star(t). \tag{D.9}$$

Contact force regulation.

$$r_t^{\mathrm{force}} = m_t^{\mathrm{CONT}}\,\alpha_f\,\exp\Big(-\frac{|f_z(t) - \tilde{f}_z^\star(t)|}{\sigma_f}\Big), \tag{D.10}$$

$$r_t^{\Delta f} = m_t^{\mathrm{CONT}}\,\alpha_{\Delta f}\,\exp\Big(-\frac{\Delta f_z(t)}{\sigma_{\Delta f}}\Big), \qquad \Delta f_z(t) = |f_z(t) - f_z(t-\Delta t)|. \tag{D.11}$$

Stiffness shaping.

$$r_t^{K_z\,\mathrm{opt}} = m_t^{\mathrm{CONT}}\,\exp\Big(-\frac{(K_z^{\mathrm{norm}}(t) - c_K)^2}{w_K^2}\Big), \tag{D.12}$$

$$r_t^{K_z\,\mathrm{adapt}} = m_t^{\mathrm{CONT}}\,\exp\Big(-\frac{|K_z^{\mathrm{norm}}(t) - K_z^{\mathrm{adapt}}(t)|}{\sigma_{K_a}}\Big), \tag{D.13}$$

$$r_t^{K_z\,\mathrm{soft}} = m_t^{\mathrm{CONT}}\,(1-\rho_t)\,\exp\Big(-\frac{|K_z^{\mathrm{norm}}(t) - K_z^{\mathrm{soft}}|}{\sigma_{K_s}}\Big). \tag{D.14}$$

Additional contact terms.

$$r_t^{\mathrm{soft}} = m_t^{\mathrm{CONT}}\,(1-\rho_t)\,\exp\Big(-\frac{|f_z - \tilde{f}_z^\star|}{\sigma_f}\Big)\,\exp\Big(-\frac{\Delta f_z}{\sigma_{\Delta f}}\Big), \tag{D.15}$$

$$r_t^{\mathrm{qual}} = m_t^{\mathrm{CONT}}\,\mathbf{1}_{\{f_z < F_{\mathrm{LOST}}\}}\,\exp\Big(-\frac{|f_z - \tilde{f}_z^\star|}{\sigma_f}\Big), \tag{D.16}$$

$$r_t^{\mathrm{work}} = m_t^{\mathrm{CONT}}\,\mathrm{clip}\Big(\frac{\sqrt{f_x^2 + f_y^2}}{f_{xy}^{\mathrm{ref}}}, 0, 1\Big), \tag{D.17}$$

$$r_t^{\mathrm{prog}} = m_t^{\mathrm{CONT}}\,\frac{\mathrm{wpt\_idx}}{T-1}. \tag{D.18}$$

Early-contact penalties.

$$p_t^{\text{impact}} = m_t^{\text{CONT}}\,(1-\rho_t)\Big(\frac{\max(0,\tilde{f}_z^{\star}-f_z-\delta_f)}{\sigma_f}+\frac{\Delta f_z}{\sigma_{\Delta f}}\Big), \tag{D.19}$$

$$p_t^{\text{hard}} = m_t^{\text{CONT}}\,\mathbf{1}_{\{n_{\text{cont}}\le N_{\text{hard}}\}}\,\max\Big(0,\frac{f_{\text{hard}}-f_z}{\sigma_h}\Big), \tag{D.20}$$

$$p_t^{v_z} = m_t^{\text{CONT}}\,(1-\rho_t)\,\max\Big(0,\frac{-v_z-v_z^{\max}}{\sigma_v}\Big). \tag{D.21}$$

Rise and global regularisation.

$$r_t^{\text{rise}} = m_t^{\text{RISE}}\,\phi_{\text{rise}}(v_z,z), \tag{D.22}$$

$$p_t^{\Delta K_z} = m_t^{\text{CONT}}\,\frac{|K_z(t)-K_z(t-\Delta t)|}{K_z^{\max}-K_z^{\min}}. \tag{D.23}$$

Global penalties include a light time penalty, a mild energy term based on torque norm, and optional penalties for workspace-bounds violations or contact-loss events when flagged.

Stiffness-and-Damping Reward $\mathscr{R}_{K_z,\zeta_z}$

Additional damping-specific terms.

$$r_t^{\text{soft\_}\zeta} = m_t^{\text{CONT}}\,(1-\rho_t)\,\alpha_\zeta\exp\Big(-\frac{|\zeta_z(t)-\zeta_z^{\star}|}{\sigma_\zeta}\Big), \tag{D.24}$$

$$p_t^{\Delta f_z} = m_t^{\text{CONT}}\,\rho_t\,\Big(\frac{|\Delta f_z|}{\sigma_{\Delta f}}\Big)\big(1+|\zeta_z(t)-\zeta_z^{\text{osc}}|\,k_\zeta\big), \tag{D.25}$$

$$p_t^{\Delta\zeta} = m_t^{\text{CONT}}\,\frac{|\zeta_z(t)-\zeta_z(t-\Delta t)|}{\sigma_{\Delta\zeta}}, \tag{D.26}$$

$$p_t^{\text{stab}} = m_t^{\text{CONT}}\,\rho_t\Big[\lambda_{\text{low}}\max(0,\zeta_{\min}-\zeta_z)+\lambda_{\text{high}}\max(0,\zeta_z-\zeta_{\max})\Big]. \tag{D.27}$$

Appendix E

Reinforcement-learning reward coefficients

E.0.1 Common control and normalisation parameters

Table E.1 Common control and normalisation parameters used in the procedural-trajectory reward (default values).

Symbol	Description	Value
Δt_{sim}	Simulation time step	0.01 s
Δt_{ctrl}	Control step (decimation = 10)	0.10 s
$K_z^{\min}$, $K_z^{\max}$	Normal stiffness clamp (procedural mode)	500, 3000 N/m
$\zeta_z^{\min}$, $\zeta_z^{\max}$	Admissible damping ratio range	0.01, 2.0
$f_{z,0}$	Base target normal force	-20 N
f_z^{ε}	Force-tracking tolerance	2.0 N
F_{TOUCH}, F_{LOST}	Reward thresholds (from config)	-2.0, -0.5 N
$\tau^{\max}$	Torque limits (for normalisation)	87 N m (shoulder), 12 N m (forearm)

E.0.2 Procedural-trajectory reward: stiffness-only configuration

Table E.2 Main coefficients used in the stiffness-only reward $\mathscr{R}_{K_z}$ (defaults).

Term	Eq.	Coefficients	Notes
Descent speed shaping r_t^{speed}	(D.1)	target speed 0.06, max 0.12, scale 6.0, penalty -2.0	Uses normalised speed ratio; active only for downward motion.
Descent time pressure r_t^{time}	(D.2)	$T_{\max} = 50$ steps, scale 4.0	Penalty after a grace period in DESCENT.
XY stability r_t^{xy}	(D.3)	weight 2.0, $\alpha_{xy} = 25$	Encourages stable lateral positioning.
Progress to surface r_t^{prog}	(D.4)	weight 3.0, $\alpha_z = 15$	Rewards proximity to the surface.
Contact bonus r_t^{contact}	(D.5)	$W_{\text{CONTACT\_BONUS}} = 10.0$	Bonus on first contact detection.
Kz prep (descent) $r_t^{K_z\,\text{prep}}$	(D.6)	target $K_z^{\text{norm}} = 0.65$, width 0.3, weight 2.0	Encourages moderate stiffness before contact.
Force tracking r_t^{force}	(D.10)	$W_{\text{FORCE}} = 8.0$, scale $0.7 f_z^E$	Exponential tracking of ramped target.
Force stability $r_t^{\Delta f}$	(D.11)	weight 4.0, scale 2.0	Penalises rapid Δf_z.
Kz optimal band $r_t^{K_z\,\text{opt}}$	(D.12)	center 0.6, width 0.3, weight 3.0	Keeps K_z in a preferred range.
Adaptive Kz $r_t^{K_z\,\text{adapt}}$	(D.13)	soft 0.4, stiff 0.7, width 0.2, weight 5.0	Softer for large force error, stiffer when tracking is good.
Soft Kz (early) $r_t^{K_z\,\text{soft}}$	(D.14)	target 0.35, scale 0.2, weight 2.0	Early-contact compliance.
Soft contact r_t^{soft}	(D.15)	$W_{\text{SOFT}} = 4.0$, bands $4.0/4.0$	Rewards small error and low Δf_z early on.
Contact quality r_t^{qual}	–	weight 3.0	Requires $f_z < F_{\text{LOST}}$.
Polishing work r_t^{work}	–	weight 2.0	Based on lateral force norm.
Waypoint progress r_t^{prog}	–	weight 2.0	Encourages completion of the stroke.
Impact penalty p_t^{impact}	(D.19)	$W_{\text{IMPACT}} = 6.0$, margin 2.0, scales $5.0/5.0$	Penalises force overshoot and Δf_z at contact.
Z-velocity penalty $p_t^{v_z}$	(D.21)	$W_{\text{ZVEL}} = 4.0$, limit 0.02, scale 0.05	Penalises excessive downward velocity after contact.
Hard-contact penalty p_t^{hard}	(D.20)	$W_{\text{HARD}} = 10.0$, limit -35, window 25, scale 10.0	Clamps large force spikes early.
Rise bonuses r_t^{rise}	(D.22)	speed 2.0, height 1.5, completion 8.0	Rewards upward motion and clearance.
Phase transition bonus	–	15.0 (phase), 25.0 (reach CONTACT)	Added when phase advances.
Kz smoothness $p_t^{\Delta K_z}$	(D.23)	$W_{\text{KPZ\_CHANGE}} = 1.0$	Penalises large ΔK_z.
Time penalty	–	weight 0.3	Linear in episode progress.
Effort penalty	–	weight 0.1	Proportional to normalised torque norm.

E.0.3 Fixed-trajectory reward: stiffness-and-damping configuration

Table E.3 Additional coefficients used in the stiffness-and-damping reward $\mathscr{R}_{K_z,\zeta_z}$ (defaults).

Term	Eq.	Coefficients	Notes
Soft-contact damping bonus $r_t^{\text{soft\_}\zeta}$	(D.24)	$W_{\text{SOFT\_DZ}} = 2.0$, target 0.9, scale 0.25	Encourages near-target damping early in contact.
Damping-weighted Δf_z penalty $p_t^{\Delta f_z}$	(D.25)	$W_{\text{DZ\_DF}} = 2.0$, scale 6.0, target 0.9, gain 2.0	Penalises oscillations more when ζ_z deviates from target.
Damping change penalty $p_t^{\Delta\zeta}$	(D.26)	$W_{\text{DZ\_CHANGE}} = 1.0$, scale 0.2	Discourages aggressive damping swings.
Stable band penalties p_t^{stab}	(D.27)	$\zeta_{\min} = 0.6$, $\zeta_{\max} = 1.4$, scales 0.2, weights $2.0/3.0$	Penalises too low or too high damping during contact.