Diversifying Data for Learning Contact-Rich Manipulation: Sensors, Kinematics, and Human Factors

Zhanpeng He

Submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
under the Executive Committee
of the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2025

© 2025

Zhanpeng He

All Rights Reserved

Abstract

Diversifying Data for Learning Contact-Rich Manipulation:

Sensors, Kinematics, and Human Factors

Zhanpeng He

Despite advancements in robot learning demonstrating diverse capabilities across various hardware platforms, many systems remain limited to pick-and-place style tasks, characterized by low precision and high compliance. These tasks typically do not require the policy to reason about contact between the object and its environment, which limits the generalizability of such methods to more complex settings where contact reasoning is essential. This limitation stems from two main factors: (1) kinematics – many systems use manipulators limited to simple grasping motions and hence can only obtain action data of simple interactions; and (2) sensing – their hardware platforms often lack contact sensing capabilities and fail to provide enough information in the observations. Furthermore, policies learned by these systems often exhibit low success rates, limiting their practical utility in real-world deployments. A common challenge is distribution shift—a mismatch between the training data and deployment conditions—which undermines the performance of data-driven methods in uncontrolled environments.

To address these challenges, this dissertation investigates several key aspects critical to enabling more effective data collection for contact-rich manipulation tasks, ultimately improving policy deployment performance. We begin by introducing novel policy learning pipelines to learn an extremely difficult task–delicate object in-hand rotation. Our proposed learning pipeline starts with an imitation learning step followed by an off-policy on-robot RL finetuning using semi-sparse rewards. To achieve this, we use a versatile dexterous robotic hand, ROAMHand3, which is equipped with a multi-modal tactile sensor, SpikeATac. This shows the potential of dexterous manipulation using robots that are carefully designed by humans. Next, we explore how to directly optimize hardware configurations for task performance using reinforcement learning, enabling co-design of morphology and control. Finally, we present a framework for leveraging human

assistance efficiently during policy deployment, demonstrating how minimal but strategic human interventions can significantly enhance real-world success rates.

Through these proposed methods, we argue that data for contact-rich manipulation tasks can be improved along multiple dimensions and they are as important as aspects (e.g., scene, object and task diversity) that other works emphasize. Enhanced sensing capabilities enable more robust performance under environmental perturbations by providing richer and more reliable feedback. Task-optimized kinematics – achieved through joint design and control optimization – expand the robot's reachable workspace and facilitate different interaction modes with diverse objects. Finally, human-in-the-loop policies provide targeted corrections in failure-prone states, enabling the system to recover from suboptimal behaviors and adapt more effectively during deployment.

Table of Contents

Dedication xvii

Chapter 1: Introduction 1

1.1 Motivations 1

1.2 Approach 2

1.3 Contributions 4

Chapter 2: Related works 6

2.1 Robotic hand design for contact-rich tasks 6

2.2 Tactile sensors 7

2.2.1 Vision-based and taxel-based tactile sensors 7

2.2.2 Tactile sensing with active acoustic sensing 7

2.3 Hardware-policy co-design 8

2.3.1 Co-design with gradient-free methods 8

2.3.2 Co-design with reinforcement learning 9

2.4 Underactuated manipulator design 10

2.5 Human-in-the-loop policies 11

I Learning Contact-rich Manipulation with Tactile Sensing and Dexterous Hands 14

Chapter 3: Learning Dexterous In-Hand Manipulation with Tactile Sensing 15

3.1 ROAMHand3 Design 16

3.1.1 Kinematic design 16

3.1.2 Contact sensors for RH3 16

3.2 Learning in-hand rotation with imitation learning 17

3.2.1 RL in simulated environment and sim-to-real transfer 17

3.2.2 Real world data collection and imitation learning 19

3.2.3 Results 20

3.3 Delicate object in-hand rotation with RH3 21

3.3.1 Observation and Action Space 23

3.3.2 Results 24

3.4 Chapter Summary 25

Chapter 4: Sensing Beyond the Contact Point with Active Acoustic Sensing 27

4.1 Introduction 27

4.2 Hardware Platform Overview 29

4.3 Sensing Procedure and Data Preprocessing 29

4.4 Estimating Object State and Properties with Acoustic Features 30

4.4.1 Object Classi cation 30

4.4.2 Grasping Position Classi cation 33

4.4.3 Pose Estimation from Internal Structure 35

4.4.4 Contact Type Classi cation 36

4.5 Task learning 38
4.5.1 Task Overview 38
4.5.2 Simulator and Training 39
4.5.3 Rollout on UR5 39
4.5.4 Results 40
4.6 Discussions 41
4.6.1 Robustness to external noises and vibrations 41
4.6.2 Effective frequency ranges 41
4.6.3 Where does acoustic sensing fail? 42
4.6.4 Integration with other tactile sensors 42
4.7 Chapter summary 43

II Hardware-policy Co-design with Reinforcement Learning 44

Chapter 5: Introduction 45

Chapter 6: Mechanical and Computational Co-Optimization using Deep Reinforcement Learning 47

6.1 Preliminaries 49
6.2 Method 50
6.3 A Mass-spring Toy Problem 53
6.4 Co-Design of an Underactuated Hand 54
6.5 Discussion 58

Chapter 7: Design Co-optimization with Reinforcement Learning via a Differentiable Hardware Model Proxy 60

7.1 Method 62
7.1.1 Preliminaries 62
7.1.2 Hardware as constraints 63
7.1.3 Deriving design parameters 64
7.1.4 Objective mismatch between task learning and hardware constraints 65
7.2 Experiments 66
7.2.1 Optimizing a reaching Robot 66
7.2.2 Optimizing robotic hands with manipulation tasks 68
7.3 Results 69
7.3.1 Task performance 69
7.3.2 Discussion 71

Chapter 8: Optimizing an Underactuated Tentacle Robot with MORPH 73
8.0.1 Transmission design 73
8.0.2 Forward actuation model for our transmission design 75
8.0.3 Task-aware co-optimization of design and control 78
8.1 Experimental Set-up 79
8.1.1 Design and control co-optimization 79
8.1.2 Hardware implementation and sim-to-real transfer 82
8.2 Results and Analysis 83
8.2.1 Co-optimization results 83
8.2.2 Sim-to-real accuracy 84
8.2.3 Energy manifold optimization 84

8.3 Chapter Summary 86

Chapter 9: Discovering Synergies for Robot Manipulation with Multi-Task Reinforcement Learning 87

9.1 Approach 88

9.1.1 Multi-task Reinforcement Learning 90

9.1.2 Synergy learning with multi-task RL 90

9.1.3 Implementation Details 91

9.1.4 Learning unseen tasks using learned synergy model 92

9.2 Experiments and results 93

9.2.1 Synergies discovery and baselines 93

9.2.2 Task sets 94

9.2.3 Performance 94

9.2.4 Learning unseen tasks with learned synergies 95

9.3 Discussion 97

9.4 Chapter Summary 98

III Human-in-the-Loop Policies for Robust Deployment 99

Chapter 10: Introduction 100

Chapter 11: Human-in-the-loop Policies with Reinforcement Learning. 101

11.1 Method 101

11.1.1 Estimation for the variance of return 103

11.1.2 HULA: Complete method 105

11.2 Evaluation 106

11.2.1 Environments 106

11.2.2 Evaluation Approach 108

11.3 Results 110

11.3.1 Uncertainty Estimation 110

11.3.2 Task Performance 111

11.4 Chapter Summary 113

Chapter 12: Human-in-the-Loop Policies with Diffusion Models 115

12.1 Method 116

12.1.1 Denoising-based uncertainty metric 117

12.1.2 Uncertainty-based intervention and policy ne-tuning 119

12.2 Experiments 120

12.2.1 Evaluation and Baselines 121

12.2.2 Simulated Environments 122

12.2.3 Ef ciency of Human Interactions 123

12.2.4 Fine-tuning Performance 124

12.2.5 Real robot experiments 125

12.3 Limitations 127

12.4 Chapter Summary 127

Conclusion and Future works 129

References 131

Appendix A: HitL policy learning with diffusion policies 148

.1 Extra baselines 148

.2 Training details, experiment setup and data statistics 150

.2.1 Experiment set up 150

.2.2 Real world environment 150

.2.3 Human-in-the-loop policy deployment 152

.2.4 Steering multi-modal policy 153

.3 Hyperparameter exploration 154

List of Figures

3.1 Sensors available for RH3. SpikeATac is a multi-modal, dynamic and high-frequency tactile sensor that can be use on a ngertip. PopcornFT is a low-cost and compact F/T sensor that can be use on a nger joint. Our design also comprises of three cameras for visual information. 16

3.2 Model architecture for imitation learning in-hand rotation. 19

3.3 Quantitative results for in-hand rotation with different initial object heights. 21

3.4 Policy learning pipeline for delicate object in-hand rotation. 22

3.5 Rollout of the ne-tuned policy at three time points. The base policy π_{RL}^{0} π_{IL} quickly crushes the paper object, while the ne-tuned policy using raw SpikeATac signals is capable of long rollouts without damaging the object. 24

3.6 Average rotation amount and object destruction rate with each RL ne-tuning learning iteration evaluated on a paper hexagonal prism and cylinder (gure shows combined results). The images show examples of objects after manipulation. . . . 25

4.1 Overview of VibeCheck. Two piezoelectric ngers on a parallel gripper send and receive acoustic signals through an object, providing information about resonant frequencies and object state (“active acoustic sensing”). By using this information to estimate the type of contact a rod is making with the environment, we demonstrate a peg insertion task using only acoustic tactile feedback. 27

4.2 Design of our active acoustic tactile sensor on a parallel gripper. While the hardware for each nger is the same, one acts as a speaker and the other as a receiver. 28

4.3 Time domain (left) and frequency domain (right) signals when the gripper is fully closed (left inset), as well as a spectrogram of the signal (right inset). The major peak around 8 kHz is consistent with the piezoelectric disk's reported resonant frequency. 30

4.4 Setup for object type and grasping position experiments and examples of signals after FFT. All training data is collected with the rod resting on a table. We show generalization on test sets collected on new surfaces and object orientations. 31

4.5 Setup for internal structure and contact type experiments, with examples of signals after FFT. 32

4.6 Regression results on the pose estimation from internal structure task using 3 frequency ranges. The grey circles denote randomly sampled poses that were in training. 35

4.7 Learning task-speci c classi ers and policy derivation. We use the probability distribution of the contact type classi er in simulation to train an imitation learning policy for peg insertion that incorporates history and is robust to imperfect contact state observations. The example confusion matrix shown here displays classi er test performance at each discrete data collection pose vs 3 contact labels. 38

6.1 Hardware as Policy overview. From the traditional perspective (a), all robot hardware is part of the simulated environment. In the proposed method (b), aspects of robot hardware are formulated as a “hardware policy” implemented as a computational graph, then optimized jointly with the computational policy. 48

6.2 Hardware as Policy-Minimal 52

6.3 The mass-spring toy problem. 54

6.4 Hand design optimization problem. Left: hand kinematics, dimension, and tendon routing. Right: successful grasps executed in simulation and on a real hand prototype. 56

6.5 Training curves for the grasping problem. Left: Z-Grasp with a large hardware parameter search range. Middle: Z-Grasp with a small hardware search range. Right: 3D-Grasp with a small search range. 58

7.1 MORPH overview. MORPH is an iterative training framework for design-control co-optimization: (A) We rst co-optimize both a control policy and a neural-network proxy of the hardware model (Hw-NN) with an RL loss and a constraint loss computed using the more realistic, physics-based hardware model (Hw-Phy); (B) Using the updated Hw-NN, we construct a dataset D of tuples of states, actions, and task actions; (C) Using D, we search for the design parameters that match Hw-Phy to the Hw-NN proxy. 61

7.3 Button pressing tasks. (A) pen-pressing task where the button is located at one end of a long object. (B) pen-pressing as performed by a person. (C) mouse-clicking task where the button is located at a corner of the object. (D) mouse-clicking task as performed by a person. 67

7.2 Optimizing a reaching robot. (A) and (C) show the optimized reacher by MORPH and Transform2Act respectively. (B) and (D) visualize the co-optimized control policies for MORPH and Transform2Act respectively. Both methods are able to optimize the link lengths to complete the reaching task. 67

7.4 Average returns vs. environment steps for the reaching task. Here, GP represents gradient projection. 69

7.5 Optimized hand with pen-pressing. (A) shows the optimized robotic hand using MORPH. (B) shows the optimized hand completing the pen-pressing task. (C) shows an optimized hand using Transform2Act. (D) shows the Transform2Act agent executing its control policy and failing to press the button. 70

7.6 Optimized hand with mouse-clicking. (A) and (C) show the optimized robotic hand using MORPH and Transform2Act respectively. (B) shows the MORPH hand completing the mouse-clicking task. (D) shows the Transform2Act agent failing to lift the computer mouse with short ngers. 70

7.7 Average returns (in log scale) vs environment steps for the button pressing tasks: (A) pen-pressing; (B) mouse-clicking. 71

7.8 Training details for mouse-clicking: (A) shows cosine similarity between hardware constraint gradients and RL gradients; (B) plots constraint loss of Hw-NN and search costs for Hw-Phy. 72

8.1 We optimize an underactuated, tendon-driven transmission for kinematic chains. We formulate and parameterize a general model for N links and M actuators. We apply our model to a three-link, two-actuator tentacle that we co-optimize using reinforcement learning (top row, left). We then validate our results on physical hardware (right). 74

8.2 Our flexible tendon-transmission design for compliant, underactuated kinematic chains. In this design, N links are driven by M actuated flexion tendons. We also implement a passive extension mechanism, which we can precisely pre-tension. The parameters of our transmission are all flexion tendon radii, extension tendon radii, and elastic tendon pre-elongations, as shown on the left. Each actuated tendon gives us access to a single 1-D manifold. By combining the tendons and alternating which tendons are in tension, the workspace of our robot now spans all of the M, one-dimensional manifolds created by each actuated flexion tendon. We are assuming quasi-static motion and ignoring gravity, so the robot will settle at a minimum energy position within these manifolds. 75

8.3 Illustrations of our optimization-based forward actuation model. (a) and (b) show the global energy landscape (z-axis). The yellow regions are manifolds that satisfy our constraints (Eq.6 and Eq.7). The red arrows in (a) represent optimization steps (Eq.5) that find the energy minimum inside the manifold. Changing design parameters and control actions result in a change in the manifold. 78

8.4 Average episode returns for goal reaching via re-fabrication. 81

8.5 Qualitative results for optimizing underactuated robots to reach different goals in simulation (goal reaching via re-fabrication). (a) shows original design and goal locations. (b) and (c) show two optimized underactuated tentacles reaching different goals. 81

8.6 Example of policy rollout on the real robot. Top: real robot rollout. Bottom: comparisons between simulated robots and real robot tracking. The real robot is shown in purple in the bottom row of figures. 82

8.7 Manifold comparisons before (a) and after (b) hardware optimization. Each grid represents the constraint satisfaction (yellow for constraints satisfied, dark purple for not satisfied) of its center. We set the joint angles $\#2$ of to be 0 for visualization purposes. 85

9.1 Example of multi-task synergy discovery. The DiscoSyn algorithm can discover manipulation synergies such as the 3-dimensional linear synergy set shown in (a), while simultaneously deriving control policies that use these synergies for manipulation tasks such as the valve turning task shown in (b). Learned synergies also facilitate subsequent learning of new manipulation tasks, such as the top-down screw-driving task shown in (c). 87

9.2 Overview of DiscoSyn. Observations from each task are fed to the task-dependent policy head $_{task}$ to derive low dimensional actions z. Then, the synergy model p takes the z as inputs and infers the full dimensional actions a that can be directly applied to the environment. 89

9.3 Two task sets are used to test DiscoSyn. Task set #1: Turning valves with different numbers of handles. Task set #2: Four tasks that have different goals (represented as different reward functions) and different dynamics (represented as different transition functions). 93

9.4 Training curves for task set #1. 93

9.5 Training curves of learning the sparse goal-condition valve turning with a pre-trained synergy model (in blue) and in full dimension action space with standard RL (in orange). 96

9.6 t-SNE visualization of low dimensional actions selected by our policy for task set #1 (a) and task set #2 (b). 97

11.1 An illustration of HULA. (A) In a partially-observable environment, an agent without the help of an expert cannot localize itself accurately due to partial observability, goes down the wrong passage and fails to reach the target. (B) A MORPH agent decides to request assistance from an available external expert in the states marked with a red E and achieves the goal (denoted by a green star). In both cases, the agent can only observe a 5x5 grid around its current location; shadow areas represent unobserved regions throughout the navigation. 102

11.2 Fully-observable grid worlds. The trap world (A) requires the agent to navigate around traps. In the shortcut world (B), the agent can leverage the expert to use the shortcut alley to achieve higher returns. 106

11.3 Partially-observable grid world. In this environment, an agent can only observe a 5x5 region around itself. Thus, perfect localization might be impossible in some areas of the map. For example, the blue state and the red state result in identical observations. 107

11.4 A learned variance map for the trap world environment. (A) shows the return variance computed by Monte-Carlo Sampling with the same policy. (B) shows the estimated variance by our method (darker colors represent higher variances). . . . 109

11.5 Comparisons between a variance map learned by MORPH and the ground truth for the shortcut world. 110

11.6 Comparisons between a variance map learned by MORPH and the ground truth for the partially-observable environment. 110

11.7 # of expert calls vs Average returns for the fully-observable experiments. Curves are generated with the rolling mean method (window size of 4) for visualization purposes. 112

11.8 Example rollouts of the MORPH agent and the penalty-based agent in the shortcut world. 113

11.9 # of expert calls vs Average returns for the partially-observable experiments. Curves are generated with the rolling mean method (window size of 4) for visualization purposes. 114

12.1 HitL policies with denoising uncertainty: We propose using denoising uncertainty as a metric for deciding when to request (human) expert assistance. Predicted de-noising vectors in end-effector position space (illustrated here via arrows on end-effector position) are collected in a vector eld, whose inter-mode divergence and intra-mode variance are used to compute policy uncertainty; when this measure exceeds a threshold, operator assistance is requested. We also show that ensuing teleoperation data can be used to ne-tune policies, achieving notable performance improvements with minimal additional data. 116

12.2 Experiments in simulated environments. Left: we consider three scenarios during policy deployment. (a) Distribution shift; (b) Partial observability (c) Action multi-modality. Right: qualitative visualization of predicted uncertainty, with lighter colors indicating higher uncertainty. 120

12.3 Sensitivity to threshold selection. 123

12.4 Average success rate of ne-tuning the Lift-sim task with different number of human intervention steps. 124

12.5 Real robot experiments: we design our experiments to elicit the challenges described in Sec. 12.2 on a real robot. Examples of unassisted, HitL, ne-tuned policy rollouts can be found in the supplementary video. 125

1 Qualitative results using (a) Sentinel [147] and (b) output variance as metrics for HitL policy deployment. 148

2 Example of the training curve for an unstable training with HULA-of ine. 150

3 Data collection pipeline. 151

4 Examples of observations for Ramekin Stacking in different stages. The top row shows image frames from one of the side cameras and the bottom row shows the view of the wrist camera. 152

5 Multi-modal demonstrations and learned modes from diffusion policy in the Open Drawer task. 153

6 Object trajectories visualization: policy random rollout vs policy steering. 153

List of Tables

3.1 Observation features available to the privileged critic and to the actor network. . . 18

4.1 Classi cation accuracy for object, grasping position, and contact type classi cation. “In-distribution" is a test set that is collected in the same conditions as training data. “New surface" and “New orientation" are test sets in which the object is resting on a surface or in an orientation not seen in training. "Interpolated poses" are test sets containing diagonal and line poses randomly sampled from within the range of training angles ("in-distribution") and diagonal poses randomly sampled from a wider range of angles than what is seen in training ("OOD") (see Fig. 4.5 and Section 11.2). 34

4.2 Success rate over 10 trials of insertion. $_x$ $_z$ 45° indicates the same start conditions as training data collection. "Random Start Poses" indicates a randomly sampled start pose. We test the latter in 2 cases: when $_x$ 45°, keeping the insertion in the training distribution, and 40.5° B $_x$ B 81°, where $_x$ values are out of the training distribution (Fig. 4.5). 41

8.1 Final distance to goal position of optimized robot for real-world reaching tasks. Open and closed represent open-loop and closed-loop execution, respectively. All units of distance in this table are mm, and each value is calculated for 20 samples. 83

8.2 Quantitative results for sim-to-real hardware transfer. Average distance between real hardware states and simulated states for both open-loop and closed-loop policy rollouts. Each value is calculated over a total of 40 samples. 83

9.1 Task set #1 results. Each row shows the results of a method and the name of the method represents: the underlying mechanism to extract synergies, the number of synergies and the type of synergy model (L for linear and NL for non-linear). For example, DiscoSyn3-L represent using DiscoSyn with 3 linear synergies. 95

9.2 Task set #2 results. 96

11.1 Quantitative results of the expert call ef ciency. Accuracy of top$N$ variance states estimation. PO represents the partially-observable experiments. 111

12.1 Average # of human assistance steps needed to achieve 100% success rates for simulated tasks. 123

12.2 Average # of human assisted steps needed to achieve 100% success rate during policy deployment. 125

12.3 Fine-tuning performance of the Lift-real task. Results are success rates derived by 20 policy rollouts per object. 126

1 Training details: S, W, F represents side, wrist, front camera views receptively. L represents low-dimensional observations that contains end-effector pose and gripper state. 151

2 Effect of Sampling Radius on Fine-tuning Performance of the Lift-sim task. 155

3 Effect of scaling constant on Fine-tuning Performance of the Lift-sim task. . . . 155

Dedication

I want to share this excitement with my beloved family, who have stood by me from the beginning. My journey of research has been a private and very personal battle I engaged in for years. There had been setbacks and things that seemed insurmountable in this process. It was the unwavering love and support of my family that carried me through and gave me the strength to persevere. To my mom, Yanfen Liang, my dad, Yongxiang He, and my brother, Zhanhong He—this achievement is as much yours as it is mine.

I also want to thank my advisors, Professor Matei Ciocarlie and Professor Shuran Song, for their transformative mentorship. I am deeply grateful to Matei for a partnership grounded in profound respect; he consistently treated me as a colleague, which gave me the con dence to trust my own research instincts. I am equally indebted to Shuran for her remarkable gift for cultivating a vibrant community. Her openness to new ideas fostered a dynamic and collaborative atmosphere that made challenging research feel like a shared adventure. To have learned from you both has been a true privilege.

I am also grateful to the friends I met along the way. They were not only a source of comfort during dif cult times, but also the ones who pushed me forward when momentum faltered. I always enjoy our research conversations, which deepen my understanding of various domains in my research. Their encouragement and companionship have been an essential part of this journey, and I owe much of my progress to them.

The works presented in this dissertation would have been impossible without the help and dedication of my student collaborators. I am extremely fortunate to have worked with groups of students who are not only passionate about research but also brought their own curiosity, creativity, and persistence into every project. Their energy inspired me during dif cult moments, and their fresh perspectives often pushed me to think in new directions. Watching them grow, contribute, and nd joy in discovery has been one of the most rewarding parts of my Ph.D. journey.

Finally, I want to dedicate this dissertation to my dog Nova, who entered my life during the third year of my Ph.D. I still remember our rst encounter—I was in the middle of a summer internship and happened to wander into a pet store in Manhattan after work. That simple visit changed the course of my life. Nova has since been a constant source of companionship, joy, and unconditional love. Through the long nights of experiments, the stress of deadlines, and the inevitable setbacks of research, Nova reminded me of the simple comforts of presence and care. This dissertation carries not only my effort, but also the warmth and solace that Nova has brought into my life.

Chapter 1: Introduction

1.1 Motivations

Data has long been a foundational pillar of robot learning for manipulation tasks. Researchers have applied machine learning across various components of robotic systems. For example, forward dynamics models [1–4] are learned for use in planning, while numerous works focus on learning end-to-end control policies through reinforcement learning [5] or imitation learning [6]. Other efforts aim to develop state estimation models [7] that transform raw sensor signals into meaningful object states. Although machine learning can be applied at different stages of the robotic pipeline, the effectiveness of these approaches fundamentally relies on the quality and appropriateness of the data. Curating task-relevant, diverse, and high-quality data remains essential for enabling robust and generalizable learning.

Recently, scaling up data collection [8–13] has become a prominent focus in robot learning research. One of its key advantages is the potential to provide broader coverage of real-world variability, which in turn improves generalization. A common axis of scaling involves increasing scene and task diversity, allowing robots to operate across a wider range of environments and interact with a broader set of objects. As a result, many studies have demonstrated that simply expanding the diversity of data—across settings, objects, and tasks – can signi cantly enhance the performance and robustness of learned models.

While scaling up data collection has proven to be a reliable strategy for improving policy performance, most existing efforts focus on pick-and-place-style tasks that do not require high precision. This limitation stems primarily from the use of simple end-effector designs. In many of these works, parallel grippers are used exclusively for grasping, and even those that employ high-degree-of-freedom (DoF) hands often treat them as simple binary actuators that can only “open”

and “close.” However, effective tool use in human environments often requires precise control over the relative pose between the end-effector and the manipulated object—a capability that cannot be achieved through quasi-static grasping alone. To support such nuanced interactions, alternative kinematic designs are necessary, as they can enable richer modes of interaction between the robot and its environment.

Another key reason for the prevalence of quasi-static actions is the lack of contact sensing. While visual sensors provide valuable state information, it is nearly impossible to position them in a way that captures suf cient detail for all tasks. This challenge is especially pronounced in contact-rich scenarios—effectively using visual input in such tasks would require camera placement that consistently captures the contact interface, which is often occluded or dif cult to access. As a result, visual feedback alone is insuf cient. Contact sensors, which directly capture information at the interaction interface, are therefore essential for enabling contact-rich and high-precision manipulation. They offer localized, task-relevant feedback that visual sensors often cannot provide, making them critical for advancing robotic capabilities beyond simple grasping.

Finally, although scaling up data enables robots to operate across diverse environments, the success rates of current robot learning systems remain insuf cient for economically viable deployment. This challenge is not unique to robotics—many real-world machine learning systems, such as autonomous driving, still rely on human intervention in unfamiliar or edge-case scenarios (e.g., rerouting around double-parked vehicles). To improve deployment performance in robotic manipulation, it is essential to account for human factors during deployment. Incorporating human assistance strategically—whether for correction, intervention, or guidance—can signi cantly enhance system robustness and help bridge the gap between research prototypes and reliable, real-world applications.

1.2 Approach

To address these challenges, this dissertation explores multiple directions for diversifying robotic data: developing manipulators that support complex interactions, leveraging novel contact sensors to capture rich contact information, and designing robot learning systems that incorporate

human-in-the-loop deployment strategies.

To enable complex interactions and contact sensing, we begin by introducing policy learning with ROAMHand3 (Ch. 3), along with the suite of contact sensors it supports, as a representative example of how complex manipulators can be designed to enable rich in-hand manipulation. We demonstrate an extremely dif cult task, delicate object in-hand rotation, on this platform, using a combination of imitation learning pretraining and on-robot RL netuning. This chapter highlights how tactile and force/torque sensing can be leveraged for challenging manipulation tasks. Despite the two types of sensors used on the ROAMHand3, we also investigate other sensing modalities. One example presented in this dissertation is the use of active acoustic sensing (Ch. 4) to infer object-related information such as material properties and external contacts. We show that this sensing approach provides rich, task-relevant feedback and can be effectively used for ne-grained manipulation. Notably, we demonstrate that this sensor alone—without any visual input—enables successful execution of a peg-insertion task, relying exclusively on tactile and acoustic information.

While the above works demonstrate how humans can design sophisticated robotic hardware, we also explore how hardware can be automatically optimized for task performance. Speci cally, we present two variants of hardware-policy co-design – one that utilizes differentiable physics and one that does not. With differentiable simulation, we propose Hardware-as-Policy (Ch. 6), an approach that integrates a parameterized physics model directly into reinforcement learning. To extend co-design methods to a broader range of environments without access to differentiable simulators, we introduce MORPH (Ch. 7), a framework that uses proxy neural networks to model physics and guide co-optimization. We also present a case study (Ch. 8) demonstrating how MORPH can be applied to optimize the design of an underactuated tentacle manipulator. To extend this part to a multi-task setting, we nally show how hand synergy (Ch. 9) can be learn using multi-task RL, where each synergy can be leveraged for robotic hand design.

Finally, to address the challenge of low success rates in deployment, my research investigates how human assistance can be effectively integrated into robot learning systems. Given that human intervention is costly and not always scalable, our goal is to minimize human involvement while

maintaining high task success rates. To achieve this, we estimate the uncertainty of the trained policy during deployment and trigger human assistance only when the uncertainty exceeds a prede ned threshold. This selective intervention framework allows the system to operate autonomously in familiar states while leveraging human input in uncertain or failure-prone situations.

We implement this idea to two major classes of policy learning methods: reinforcement learning and imitation learning. For reinforcement learning, we introduce HULA (Ch. 11), which estimates uncertainty based on the variance of the learned Q-values and uses this information to decide when to request human assistance. For imitation learning (Ch. 12), we propose a novel use of the denoising process in diffusion models as a proxy for uncertainty, enabling ef cient intervention during deployment.

1.3 Contributions

This dissertation makes the following contributions:

1. Learning contact-rich dexterous manipulation tasks with a multi- nger robotic hand [14] Develop novel policy learning methods for highly dexterous tasks using raw tactile information. We demonstrates an extremely dif cult task, delicate object in-hand rotation, using a combination of imitation learning and on-robot reinforcement netuning. Our method leverages semi-sparse reward labels, allowing for applications in diverse tasks without task-speci c state estimation for dense rewards. To the best of our knowledge, we are the rst to demonstrate this task on a real robot platform.

2. Active Acoustic Sensing for Contact Sensing [15] Investigate sensing capability of active acoustic sensors for contact sensing. We demonstrate that our sensor can be used to achieved a peg-insertion task without any visual information.

3. Hardware-policy Co-design with RL [16–19] Introduce two novel algorithms for hardware-policy co-optimization in simulation environments, each based on a different assumption about the hardware model. The rst method assumes access to a differentiable hardware

model, allowing gradients to flow through the physics engine for end-to-end optimization. To the best of our knowledge, we are the first to express hardware and computation as a unified RL policy which allows the optimization algorithm to propagate gradients of the actions w.r.t both hardware and computational parameters simultaneously. The second method operates under the assumption of a non-differentiable hardware model, and instead relies on a learned proxy model to guide the optimization process. Both approaches are designed to jointly optimize hardware parameters and control policies for improved task performance. Furthermore, we validate the effectiveness of these methods on real hardware through sim-to-real transfer, demonstrating their practical applicability beyond simulation.

4. Human-in-the-loop Policies for Robust Deployment [20, 21] Introduce the use of uncertainty estimation in two major classes of policy learning methods—reinforcement learning (RL) and imitation learning (IL)—to enable efficient human-in-the-loop deployment. By estimating policy uncertainty during execution, our approach selectively triggers human intervention only when the agent is likely to fail. This allows us to minimize human effort while maintaining robust deployment performance in complex manipulation tasks. Furthermore, we demonstrate that our method facilitates continuous improvement by using uncertainty-guided interventions as a means to collect targeted data in failure-prone scenarios, thereby refining the manipulation policies over time.

These contributions and associated chapters include reproductions of the following academic papers for which the dissertation author has or shares primary authorship:

- SpikeATac: A Multimodal Tactile Finger with Taxelized Dynamic Sensing for Dexterous Manipulation [14]
- VibeCheck: Using Active Acoustic Tactile Sensing for Contact-Rich Manipulation [15]
- Hardware as Policy: Mechanical and Computational Co-Optimization using Deep Reinforcement Learning [16]
- Design Co-optimization with Reinforcement Learning via a Differentiable Hardware Model Proxy [17]
- Task-Based Design and Policy Co-Optimization for Tendon-driven Underactuated Kinematic Chains [18]
- Discovering Synergies for Robot Manipulation with Multi-Task Reinforcement Learning [19]
- Decision Making for Human-in-the-loop Robotic Agents via Uncertainty-Aware Reinforcement Learning [20]
- Uncertainty Comes for Free: Human-in-the-Loop Policies with Diffusion Models [21]

Chapter 2: Related works

2.1 Robotic hand design for contact-rich tasks

Most of the work presented in this dissertation focuses on dexterous manipulation and is closely tied to manipulator design, a long-standing area of research in robotics. One common approach in robotic hand design is to build anthropomorphic hands [22, 23] that mimic the structure and kinematics of the human hand. For instance, the Shadow Hand [24] and ADROIT Hand [25] have demonstrated complex dexterous manipulation capabilities but they are expensive. The Allegro Hand [26] offers a more affordable, direct-drive solution and has been adopted in various robot learning applications [27–29] due to its lower cost. However, these hands often struggle with many everyday tasks because of weak actuators and limited kinematic con gurations. To strike a balance between functionality and simplicity, the D'Manus hand [30] employs a three- nger design, re ecting the fact that many human tasks can be performed with just a thumb and two ngers. In a similar spirit, Shaw et al. [31] introduced the LEAP Hand, an open-source, low-cost anthropomorphic hand that aims to approximate human capabilities while remaining accessible to the research community.

Our hand design takes a different approach. Rather than aiming for anthropomorphism, we use a symmetric, non-human-like structure, which simpli es certain manipulation skills such as in-hand rotation. Non-anthropomorphic hands [32–34] have also been explored in prior work – for example, DEX-EE [35] (the successor to the Shadow Hand) adopts a three- nger kinematics optimized for durability and robustness rather than biological resemblance. A key differentiator of our design is its support for on-device sensor integration, including both contact sensors and cameras. While many existing dexterous hands lack built-in contact sensing or rely on external vision systems, our platform enables multi-modal perception directly at the point of contact, making it well-suited for

contact-rich tasks.

2.2 Tactile sensors

2.2.1 Vision-based and taxel-based tactile sensors

Tactile sensing has gained increasing attention in recent years, particularly with the rise of vision-based tactile sensors in robotic manipulation tasks [36–44]. These sensors have bene ted from advancements in computer vision and offer high spatial resolution, making them easy to interpret both by humans and machine learning models. However, they come with notable limitations. Vision-based tactile sensors are typically bulky due to optical requirements such as focal length, which constrains their adaptability to different surfaces and form factors. Additionally, they require high data bandwidth, making real-time data transfer challenging. Finally, due to their relatively low frame rates, these sensors are often inadequate for capturing fast, dynamic contact events.

In contrast, taxel-based tactile sensors, while sacri cing spatial resolution, address the issues of data bandwidth and mechanical adaptability. These sensors use transducers distributed across speci c surface regions and encode spatial information by increasing the density of sensing elements. This class of sensors includes various transduction mechanisms, such as capacitive [45, 46], resistive/piezoresistive, piezoelectric, magnetic [47, 48], and barometric [49–51] modalities.

In this dissertation, the ROAMHand3 integrates piezoelectric sensors for detecting dynamic contact events, capacitive sensors for spatial localization, and accelerometers for capturing frequency-domain information. This combination provides multi-modal tactile feedback, offering rich and diverse contact information essential for contact-rich robotic manipulation tasks.

2.2.2 Tactile sensing with active acoustic sensing

A few works have demonstrated how active acoustic sensing can be used in robotics. In soft pneumatic actuators (SPAs), a speaker and receiver system can be placed within the air cavity of the actuator, measuring the change in the acoustics due to elastomer deformation. This type of system can be an effective way to measure contact location and actuator shape [52–56]. Beyond SPAs, a

similar system can be used to create a low-cost tactile skin. This work places two piezoelectric contact microphones on a robot arm to do proximity sensing [57] and contact localization [58] over the entire arm, effectively turning the arm into a large tactile skin. Active acoustic sensing can also be implemented with a single transducer. SonicFinger measures proximity using a single piezoelectric transducer to emit a signal into the environment and measure the returning signal [59].

In manipulation, the more common use of acoustic sensors is passive contact microphones, which act as a dynamic tactile sensing modality. For example, SonicSense uses a multi- ngered hand equipped with contact microphones along with a heuristic-based exploration policy for container contents identi cation, 3D shape reconstruction, and object re-identi cation [60]. Other works use contact microphones as tactile feedback for motor learning [61–64]. In comparison to our work, these works rely on dynamic interaction to create a signal and cannot measure object state without a stimulus from a motor or the environment.

A pair of studies have used active acoustic sensing in a parallel jaw gripper to measure object states, as we do in this work [65, 66]. These studies demonstrate material classi cation, grasping position estimation, and contact state measurements in the presence of occlusion. They are the rst to show active acoustic sensing can be useful in a gripper, where the ngers are “communicating” with each other through acoustics. However, Yi and Lee et al. only show how acoustic sensing, as a complementary sensing modality to visual sensors, can improve extrinsic contact identi cation performance. In this study, we streamline the design by using the same component type as both emitter and receiver, and go beyond static classi cation to also demonstrate a complete control policy for a standard long-horizon manipulation task relying exclusively on active acoustic sensing.

2.3 Hardware-policy co-design

2.3.1 Co-design with gradient-free methods

Considerable research has investigated co-optimizing the design and control of a robot [67–73]. Many works use analytical dynamics and classical control [74], a number of which optimized mechanical and control or planning parameters for legged locomotors [75–77]. All studies above

require an analytical model of the complete mechanical control system, which is non-trivial in complex problems. More recent work uses classical control but evaluates and iterates on real hardware [78], applying Bayesian Optimization to microrobots. However, the goal is different from ours: this study aims to decrease the number of real-world design evaluations, which is avoided in our work by training in simulation and performing sim-to-real transfer.

Evolutionary computation provides another way to approach this problem [79]. This research path originated from studies on the evolution of arti cial creatures [80], where the morphology and the neural systems are both encoded as graphs and generated using genetic algorithms. Lipson and Pollack [81] introduced the automatic lifeform design technique using bars, joints, and actuators as building blocks of the morphology, with neurons attached to them as controllers. A series of works from Cheney et al. [82, 83] studied the morphology-computation co-evolution of cellular automata, in the context of locomotion. Xu et al. apply graph heuristic search [70] with RoboGrammar [84], which is a set of graph grammar for robot design, for terrestrial robot design. Evolutionary strategies, which are gradient-free, have signi cant promise but also exhibit high computational complexity and data inef ciency compared to recent gradient-based optimization methods.

Other gradient-free methods have been considered for co-design. Deimal et al. [85] apply particle lter optimization method to optimize the shape of a soft robotic hand and grasping poses in simulation. Liao et al. [86] propose a Bayesian optimization method to tune the design of a microrobot ef ciently to a walking task. However, gradient-free optimization methods suffer from long-horizon reasoning and can not be used for evaluating a robot with complex tasks.

2.3.2 Co-design with reinforcement learning

The recent in ux of reinforcement learning provides a new perspective on the co-optimization problem [87–91]. Wang et al. suggest Neural Graph Evolution (NGE) [92], which models the structure of an agent as a graph neural network (GNN), optimizes the morphology of an agent by changing its graph structure, and learns to control by adapting the parameters of the GNN. Luck et al. [93] propose to learn a latent space that represents the design space and train a design-

conditioned policy. Transform2Act [94] considers a transform stage when actions can modify a robot's kinematic structure and morphology, then a control stage when the design is frozen and the policy only computes control actions. Our work on hardware-policy co-design focuses on optimizing hardware parameters in parameter space and uses direct gradients from an RL objective.

Recent work on auto-differentiable physics [95–98] is also relevant to us, as we rely on modeling (part of) the robot hardware as an auto-differentiable computational graph. We hope to make use of advances in general differentiable physics simulation in future iterations of our method.

Finally, we observe the existence of optimization dif culty caused by gradient interference introduced from the mismatch between RL improvement and learning realistic hardware. This is similar to the observed optimization dif culty in multi-task learning literature [99, 100] if we treat task improvement and being realistic as two tasks. Previous works have explored mitigating the con icts for multi-task learning. For example, Senor and Koltun [101] scale the gradients introduced by different tasks to reduce the scale difference among gradients. GradNorm [102] uses gradient normalization to facilitate multi-task learning. Our work of MORPH is inspired by PCGrad [103], which uses cosine similarity between gradients to measure the con icts and project a gradient to the normal plane of another when con icting. In this work, we treat the task learning and hardware constraints as a dual-task learning problem and project the task learning gradients to the normal of hardware constraint gradients.

2.4 Underactuated manipulator design

An underactuated manipulator is a mechanical system composed of links and joints that has fewer actuators than degrees of freedom [104]. The transmission of these manipulators are often tendon-driven, as a single actuated tendon can be routed to control multiple joints. Reducing the number of bulky actuators enables designers to build more compact and lightweight manipulators. Moreover, tendon-transmission allow designers to dislocate the motors from the joints. Dislocating the actuators reduces the inertia of the links and makes it easier to make the manipulator robust to water, dust, and other dif cult environmental conditions [105]. With so many practical bene ts,

a myriad of diverse underactuated, tendon-driven manipulators have been proposed over the past several decades.

There is an extensive history of underactuated manipulator design. The rst underactuated manipulator, Hirose's soft gripper introduced in 1978, was designed to softly capture a diverse range of objects with uniform pressure [106]. Over time, these manipulator designs became much more advanced and their applications now extend to more robust and intricate grasping behaviors [107–111]. Currently, there is extensive research in the design and control of underactuated manipulators for tasks even more dexterous than grasping, such as in-hand manipulation [112–114]. To realize these more advanced capabilities, the tendon-transmission of these manipulators had to be carefully designed, optimized, and further hand-tuned. This process is cumbersome and time-consuming, but necessary. Co-optimization methods are a possible tool to help design these complicated mechanisms. With recent advancements in reinforcement learning, co-optimization is now more powerful than ever.

2.5 Human-in-the-loop policies

The last category of works in this dissertation takes advantage of human assistance by learning a human-in-the-loop policy. Researchers have investigated how to incorporate human feedback to improve policy performance [115–117]. For instance, Knox and Stone [118] propose the TAMER framework, which learns a model to minimize the difference between the reinforcement function from a policy and a human. Celemin and Ruiz-del-Solar [119] propose COACH, an interactive framework that utilizes human corrective advice on the magnitude of continuous actions during training a policy. Compared to these works, which focus on using human assistance to derive high-performance policies in train time, HULA achieves higher task performance by learning a policy that can actively seek assistance from humans in test time.

HitL approaches [120–125] have been widely explored to enhance robot manipulation by incorporating various human feedback modalities, such as interventions [126, 127], preferences [128], rankings [129], scalar feedback [130], and gaze [131]. Recent works like HIL-SERL [132] and

Sirius [133] show strong performance by leveraging human input during training. HitL methods are also common in autonomous driving, e.g., ZOOX's re-routing system [134] allows operators to assist in challenging scenarios. However, most approaches rely on continuous human supervision and monitoring when assistance is actually needed during deployment. We propose an uncertainty-aware diffusion model that selectively requests human help in high-uncertainty states, reducing the need for constant oversight.

Our methods explicitly estimate the uncertainty of an RL agent and use it to make decisions about requesting an expert's assistance. Prior work has explored representing the uncertainty of outcomes during RL training [135–138]. For instance, Kahn et al. use the variance of the dynamics model to represent the uncertainty of collision to avoid damage from high-speed collisions [139]. Ensemble Quantile Networks (EQN) [140] estimates both aleatoric and epistemic uncertainties via the combination of quantile networks(IQN) [141] and randomized prior function(RPF) [142]. These works use the uncertainty of an agent to perform task-speci c decision-making, while our work differs in using it to collaborate with a human. Moreover, HULA represents an agent's uncertainty using the variance of returns, which is a direct indicator of task performance.

Our work is also closely related to interactive imitation learning (IIL), where a learning agent queries an expert for additional labels during policy execution and augments the training dataset with expert demonstrations [143]. An effective data collection strategy for IIL is human-gated DAgger [144], which relies on a human to continuously monitor and intervene during robot execution. However, such continuous supervision is inef cient and undermines the goal of robot learning. To make HitL systems practical, the robot must be strategic in when and how it requests human assistance. Prior work has explored budgeting expert queries in IIL. For example, Hoque et al. reduce human effort by constraining robot queries using models of limited human attention [145] or xed intervention budgets [146]. However, these methods require extra training procedures during training or deployment, increasing computational demands. This overhead is especially problematic during policy execution, where it may introduce additional latency. In contrast, our approach avoids additional training and supports ef cient parallelization during deployment, resulting in minimal

runtime overhead. Other approaches leverage action consistency [147], diffusion loss [148], or online conformal prediction [149] to decide when to query the human. One crucial aspect of this line of work is the selection or online-tuning of their thresholds. To minimize human efforts in a HitL pipeline, our method leverages the multi-modal nature of human demonstrations to detect critical states where the robot is uncertain. We demonstrate that our method is robust to threshold selection in different scenarios.

Part I

Learning Contact-rich Manipulation with Tactile Sensing and Dexterous Hands

Chapter 3: Learning Dexterous In-Hand Manipulation with Tactile Sensing

In this chapter, we demonstrate complex in-hand manipulation tasks that require a robot to modulate forces precisely. To achieve this, we developed a learning pipeline that integrates several key components. We begin by learning a sensorimotor policy through imitation learning, using data collected from a reinforcement learning (RL) policy transferred from simulation to the real world (sim-to-real). Subsequently, we propose an on-robot RL pipeline that integrates reinforcement learning from human feedback (RLHF) with tactile information to further improve the policy's performance.

On the hardware side, we use the ROAMHand3 (RH3), a dexterous robotic hand speci cally designed for contact-rich manipulation. A key feature of RH3 is its adaptability to various contact sensors, including ngertip tactile sensors and six-axis force/torque (F/T) sensors, which enables rich multimodal sensing during manipulation.

By combining these components, the resulting policies demonstrate strong generalization capabilities and can be reliably deployed across different environments. This showcases the effectiveness of pairing advanced hardware with sensor-driven learning. Speci cally, we demonstrate an extremely dif cult task: the in-hand rotation of a delicate object. This task requires the policy to generate complex action trajectories to control a high-degree-of-freedom hand while precisely modulating force. To the best of our knowledge, we are the rst to demonstrate this task on a real robotic platform.

Figure 3.1: Sensors available for RH3. SpikeATac is a multi-modal, dynamic and high-frequency tactile sensor that can be use on a ngertip. PopcornFT is a low-cost and compact F/T sensor that can be use on a nger joint. Our design also comprises of three cameras for visual information.

3.1 ROAMHand3 Design

3.1.1 Kinematic design

RH3 is a non-anthropomorphic robotic hand with four ngers. Each nger has three joints – one Metacarpophalangeal (MCP) joint, one Proximal Interphalangeal (PIP) joint, and a Distal Interphalangeal (DIP) joint. Together, RH3 has 12 degrees of freedom, allowing the hand to grasp an object using two ngers and use the other two to perform in-hand manipulation of the object. We direct-drive all nger joints, and use Dynamixel XC430 motors for the MCP joints as well as Dynamixel XC330 motors for the PIP and DIP joints.

3.1.2 Contact sensors for RH3

RH3 is designed for contact-rich tasks. To achieve this goal, we equip RH3 with different types of contact sensors. Speci cally, we look into nger-tip tactile sensors and 6-Axis force/torque sensors. Here, we give a brief overview of the capabilities of these sensors, and defer the readers to their original papers [14, 150] for the details of sensor design.

For tactile sensors, we use SpikeATac [14], a class of high-frequency and multi-modal sensors. A key feature of these sensors is their ability to capture both static and dynamic contact information. To collect static information, we use a layer of single-Tac, a type of capacitive tactile sensor. On top of it, we have a layer of 16 piezoelectric sensors for high-frequency and dynamic information. Finally, we also have an accelerometer on the back of the this nger.

For force/torque sensors, we use PopcornFT [150], a low-cost and small form factor F/T sensor. This sensor leverage the displacement of LED light signals to sense the displacement. PopcornFT's the standalone performance is characterized by testing the ability of a supervised learning model to predict complete force and torque data from its raw signals, and obtain a mean error between 0.05 and 0.07 N across the three directions of force applied to the nger.

To capture enough visual information that happens on nger-tips, our design has a palm that is large enough to house a Realsense D450 RGBD camera in-palm. We also integrated two webcams two sides of the robotic hand to obtain a larger receptive eld of the environment.

3.2 Learning in-hand rotation with imitation learning

We demonstrate the capabilities of RH3 using an in-hand rotation task. In this task, we ask the robot to rotate an object without external support for gravity. This requires the robotic hand to constantly make and break contact with the object on nger-tip to avoid dropping the object, hence the need for reasoning about contact between the ngers and the hand.

A key challenge of training policies for non-anthropomorphic hands is data collection. Our tactile sensors are not simulate-able, making training robust real-world policies from simulation not viable. Then, collecting real world data via tele-operations with a non-anthorpomorphic hand require non-trivial re-targeting between human hands and robotic hands due to its high degree-of-freedom and unmatched kinematics. To this end, our training strategy begins by developing an initial base policy in simulation only relying on information that we can obtain in the real world. Then, we use this policy, combined with sim-to-real techniques (e.g., domain randomization and system identi cation), to collect data in the real world. Finally, we train an expressive policy using imitation learning from the collected data.

3.2.1 RL in simulated environment and sim-to-real transfer

We build a simulated environment using Isaac gym [151] and use Proximal Policy Optimization (PPO) [152] to train a multi-layer perceptron (MLP) policy. In this work, we use an asymmetric

actor-critic style when we are training the value baseline. For the observation space, we use joint positions, binary contact between the object and the distal link, and the object height on the z axis. For the action space, we use joint position control and the policy outputs a 12-dimensional vector, representing joint set points.

Input feature	Critic	Actor
Hand joint position		
Hand joint velocity		
Target joint position		
Fingertip contact force		
Object position		
Object height (z)		
Object orientation		
Object linear velocity		
Object angular velocity		
Target orientation		
Target orientation		
Joint torque		
Fingertip contact (bool)		
Fingertip contact position		
All contacts (bool)		
Object keypoints		

Table 3.1: Observation features available to the privileged critic and to the actor network.

We select the observation with two major considerations: 1. they need to be useful to produce a robust policy and 2. they can be estimated using the real robot sensors with reasonable accuracy. In this task, we use joint encoders come with the Dynamixel motors for joint positions and estimate the binary contact using the raw tactile signals. Speci cally, we collect 250 steps of capacitive data at the beginning of the real robot pipeline and calculate the median as a baseline. Then, We use the ltered, baseline-subtracted signal to estimate the contact state. When any sensor channel produces a value exceeding a threshold, that nger is marked as “contacted". Here, the threshold is decided by using human nger to poke each sensor. To estimate the object height, we use the in-palm RGBD camera and directly use the value of the center pixel as its height.

The agent is rewarded for spinning the object about a target axis while keeping motion, effort, and contact forcTes low. We clip the angular velocity to avoid runaway incentives, apply a penalty to translational velocity v_t so the object does not drift away from the center as it spins, and use a

Figure 3.2: Model architecture for imitation learning in-hand rotation.

pose-difference term to improve stability. Effort and work penalties depend on the joint torques τ_t. If the object falls, we add an out-of-bounds penalty, and a hard-force penalty discourages the hand from slamming the object. At timestep t,

$$r_t = \text{clip}_{v_{max}}(\omega_t \hat{a}) - \gamma_1 \|\tau_t\|^2 - \gamma_2 d_t - F_t - 1_{OOB}.$$

Here, ω_t is the object's angular velocity projected onto the desired axis $\hat{a}$.

We also perform domain randomization [153] in simulation so the policy can be robust to the error from calibrations and contact prediction. Instead of training a domain randomized policy from scratch, we start by training the policy with a fixed set of parameters. Then we slowly make the randomized range larger over the course of training.

3.2.2 Real world data collection and imitation learning

With a trained RL policy, we then deploy the policy on a real robot for collecting raw sensor signals. In this work, we collect data in 100Hz from all sensors (i.e., tactile sensors and three cameras). In this task, we use the SpikeATacs with only the capacitive sensors and accelerometers since the PVDF version of the sensors is not available at the time of experiment.

The raw tactile signals from the SpikeATacs sensors are noisy. To address this problem, we first calibrate the signals by collecting 2000 steps of raw signals and then grabbing the median. The median is subtracted from all subsequent values. The tactile signals are then normalized using a

scaled sigmoid function:

$$\hat{z}\bullet \quad \frac{1}{1 \quad \exp\hat{} \quad z \bullet}; \text{ where } \quad 0{:}016$$

We chose an alpha value of 0.016 to support a range of [-137, 137], which was determined by hand by looking at the distribution of tactile values in the dataset.

With the collected data set, we then train an imitation learning policy. Speci cally, the policy encodes different modalities of observations using dedicated encoders rst. Then all encoded observations are fed to a transformer model, where it treat each type of observations at a time step as a token. Finally, its output is fed to a MLP for action decoding. Here, we use ResNet18 [154] for visual encoders and MLPs for low-dimensional information tokenization.

3.2.3 Results

We assess our policies by evaluating their robustness. In our context, we de ne robustness as the ability to rotate the lugnut object without failure. Failure is de ned as a mishandling of the lugnut. Mishandling can be further classi ed into two types. First is dropping where the lugnut falls. This case mainly occurs when the lugnut drops too low in the hand. Second is tipping, where the ngers lose grip of the lugnut when it rises too high. Here, the lugnut becomes oriented in the incorrect axis and the hand is unable continue turning the lugnut. To test these policies, we measure how long each policy can manipulate the lugnut before failure occurs. The speed of failure during an episode is highly dependent on the starting position of the object. To systematically test this, we used a adjustable jig to ensures that the starting position and orientation of the object is the same among all trials. We test the policy results in 10 different height settings (labeled 0-9), which allow us to introduce a near-failure state at the start of the episode and evaluate how effective a policy is able to recover.

From our results, we see that our BC (with RGBD) policy consistently outperformed our BC (without RGBD) policy. We also see from the results that the BC (with RGBD) policy performed

Figure 3.3: Quantitative results for in-hand rotation with different initial object heights.

similarly to the RL expert policy at height settings 4 and 8, but outperformed the RL expert by a signi cant margin at extreme heights 9, 2, and 1. These results indicate that our BC (with RGBD) policy is more robust than the RL expert as it is better able to recover from near-failure states on both high and low height extremes.

To gauge real-world robustness we ablated sensing modalities and measured how long the robot could keep the object aloft (mean s.d. over three trials; initial drop height = 4 cm). Covering the palm RGBD camera sharply degrades vision-centric policies, yet the multisensory BC (w/ RGBD) variant still more than doubles baseline performance. Zeroing tactile inputs weakens the BC (w/o RGBD) model, con rming its dependence on contact cues. Surprisingly, BC (w/ RGBD) shows no loss outside the light box, indicating good generalization to visual disturbances. Overall, robustness is preserved provided every modality the policy expects remains available.

3.3 Delicate object in-hand rotation with RH3

In this section, we demonstrate an even more challenging task—delicate in-hand object rotation. Speci cally, we ask the four- nger robot, equipped with the latest SpikeATac using capacitive

Figure 3.4: Policy learning pipeline for delicate object in-hand rotation.

sensors, PVDF and accelerometers, to rotate origami objects without destroying them. This task is extremely dif cult because the robot must precisely modulate contact forces: it needs to apply enough force to securely hold the object, yet avoid excessive force that could damage it (see Table??). The required action trajectories are both complex and dexterous, as the object receives no gravitational support and the hand itself has a high degree of freedom. The robot must coordinate multiple ngers to simultaneously stabilize and rotate the object. To the best of our knowledge, this is the rst demonstration of such a task.

At a high level, we start with a base policy $_{IL}$ that operates on raw sensor signals from our robot. To obtain $_{IL}$, we train a behavioral policy in simulation via RL using a highly simpli ed, binary contact signal (touch vs. no-touch). We transfer this policy to the real robot using standard domain-randomization, and use it to collect a number of demonstrations on our real platform, which are in turn used to train a base policy $_{IL}$ via imitation learning.

While $_{IL}$ works well on hard objects (shown in our supplementary video), it immediately fails on fragile ones, as it has been trained with rigid objects and lacks the ability to produce delicate touch. We thus ne-tune it using RL, speci cally Soft Actor-Critic [155], on raw sensor data acquired from our sensor. This pipeline shows how the highly-sensitive, dif cult-to-simulate sensor signals can be integrated into learning-based manipulation.

To initialize RL ne-tuning, we set $^{0}_{RL}$ $_{IL}$ and deploy it on the real robot to collect data. To

enable effective data collection for training the Q-function, we introduce an exploration strategy by injecting Gaussian noise with small standard deviations into the distal joints. At each iteration, we use the most recent policy π^n_{RL} to collect a dataset D^n_{RL}. For training, we aggregate this RL dataset with the original imitation learning dataset, i.e. $D_{RL} \cup D^n_{RL}$.

For RL fine-tuning, we incorporate two types of reward signals. The first type consists of task-specific rewards r_{task} provided by human annotators. For each trajectory, a human labels segments as "good" if the object is rotating and "bad" otherwise. We refer to this as semi-sparse reward labeling. Compared to sparse rewards, where only the final timestep is labeled based on whether the agent achieves the goal, this approach provides richer feedback across different parts of a trajectory. At the same time, unlike dense rewards, our reward labels do not require overly complex state estimation or engineering in the real-world setup.

The second type of reward signals is derived directly from tactile sensor observations. At a high level, our goal is to encourage the agent to reduce excessive contact forces (captured by capacitive readings) while increasing exploratory contacts (reflected in a higher number of PVDF spikes, i.e., making and breaking contact with the object). Formally, the tactile reward is defined as:

$$r_{tac} = w_1 \sum_{i \in \{1,2,3,4\}} \left(1 - o^i_{cap}\right) - 0.9 \quad w_2 \sum_{i \in \{1,2,3,4\}} \left(1 - s^i_{pvdf}\right) - 0.8 \tag{3.1}$$

Here, w_1 and w_2 are constants ($w_1 \geq 0$, $w_2 \geq 0$). o_{cap} and o_{pvdf} are tactile sensor readings that are normalized to the range of $[0, 1]$ using a sigmoid function, then scaled and shifted to the range $[-1, 1]$. i is the finger index. The combined reward for RL fine-tuning is $r_{tac} + r_{task}$.

3.3.1 Observation and Action Space

We carefully handle raw PVDF sensor signals in the observation because we do not want to discard high frequency information in the sensor data. We sample the PVDF at 450 Hz and capacitive sensors at 250 Hz (capacitive internal update rate is still 40 Hz), while the policy runs at 20 Hz (sampling frequencies are limited by the bandwidth of the microcontroller controlling the

Figure 3.5: Rollout of the ne-tuned policy at three time points. The base policy π^0_{RL} $_{IL}$ quickly crushes the paper object, while the ne-tuned policy using raw SpikeATac signals is capable of long rollouts without damaging the object.

entire hand). The signals have a median lter of width 5. A history buffer of 64 PVDF and capacitive readings incorporates higher frequency information into the policy at each lower frequency timestep.

The observation contains the length 64 history buffer of sensor signals (16 PVDF signals and 7 capacitive signals per each of 4 ngers) and the 12 joint encoder readings. The action space is the hand's 12 joint position set points for its 12 degrees of freedom (1 roll joint and 2 exion/extension joints per nger).

3.3.2 Results

To evaluate performance, we roll out the policy at each iteration on a paper hexagonal prism ($d = h = 40$ mm) and paper cylinder ($d = 40$ mm, $h = 60$ mm) for 5 rollouts each. We evaluate amount of rotation and destruction rate for each policy (Fig. 3.6).

The initial π^0_{RL} $_{IL}$ policy almost immediately crushes the objects (Fig. 3.5), however through this learning pipeline the policy learns to make visibly softer contacts over the learning iterations, as evidenced by the object state after the experiments (Fig. 3.6), while amount of rotation also increases. See the supplementary video (also available for convenience at the anonymized address https://spikeatac.github.io) for visual results of improved rotation over learning iterations.

We compare RL ne-tuning results to an IL baseline that is trained on all available demonstration data after all ne-tuning iterations, with the IL baseline achieving 5.7 radians rotation and 0.7

Figure 3.6: Average rotation amount and object destruction rate with each RL ne-tuning learning iteration evaluated on a paper hexagonal prism and cylinder (gure shows combined results). The images show examples of objects after manipulation.

destruction rate, notably lower than the RL ne-tuning results. We emphasize that the data we use for the IL ne-tuning baseline is usually hard to obtain in an IL setting, given the quick failures of the base policy. Here, we lter the RL ne-tuning dataset using human labels and only use the good trajectories for imitation learning.

These experiments show that this system and pipeline as a whole enable this dif cult, delicate task. We note that the resulting policy in these experiments may be over t to these particular objects and object sizes, and that we also don't have a de nitive analysis of what each sensor contributes. Our primary takeaway, however, is that this pipeline, which is designed for the real robot and is using real sensor signals in the reward, allows effective use of SpikeATac data for new data-driven capabilities, despite the sensor being, as of now, very dif cult to simulate.

3.4 Chapter Summary

In this chapter, we presented a learning system for acquiring dexterous, contact-rich manipulation skills. We rst show that RL in simulation, combined with proper sim-to-real technique can be used to train a policy to collect data on a real hardware platform. These data are then used for training sensorimotor policies using imitation learning. Finally, we demonstrate an ef cient and effective

on-robot RL pipeline for continuous improvement of a learned policy. To achieve these tasks, we developed two hardware platforms, RH3 and SpikeATac. Our experiments demonstrate success on one of the most challenging tasks in dexterous manipulation—delicate in-hand object rotation.

This work shows that with carefully designed hardware and learning frameworks, robots can master tasks requiring intricate low-level motor skills. In contrast to much of recent robot learning research, which has focused on scaling policy learning from diverse tasks and scenes with imitation learning, our results highlight the effectiveness and importance of scaling data modalities and incorporating reinforcement learning ne-tuning.

Chapter 4: Sensing Beyond the Contact Point with Active Acoustic Sensing

4.1 Introduction

Figure 4.1: Overview of VibeCheck. Two piezoelectric ngers on a parallel gripper send and receive acoustic signals through an object, providing information about resonant frequencies and object state ("active acoustic sensing"). By using this information to estimate the type of contact a rod is making with the environment, we demonstrate a peg insertion task using only acoustic tactile feedback.

Tactile sensors typically acquire data by making and breaking contact with the environment, providing local information about contact points and surfaces. While many tasks bene t from knowing where contact occurs and understanding local geometry and forces, passive, intrinsic contacts (i.e. contacts between a sensor and the environment) are limited by the sensing area of the sensor and leave a lot of useful information unutilized. For instance, material properties and extrinsic contacts (i.e., contacts an object makes with the environment, not the sensor) are dif cult to infer.

One way to provide such information to a local tactile sensor is to introduce an active component [60, 156], conceptually similar to how a child would shake or tap a boxed present to gure out its contents. In other words, one can introduce a signal into the system and observe the response.

This is the principle of active acoustic sensing, in which an active emitter element (i.e. a speaker) emits an acoustic signal and a receiver element (i.e. a microphone) records the response [52, 157–162]. In the context of a multi- ngered robotic gripper, this means sending an acoustic

Figure 4.2: Design of our active acoustic tactile sensor on a parallel gripper. While the hardware for each nger is the same, one acts as a speaker and the other as a receiver.

signal through an object from one nger to another. This signal can be useful for learning about object composition and geometry, extrinsic contact state (i.e. how the object is making contact with the environment), and even internal structure or mass, all qualities that typical tactile and vision modalities struggle with [65, 66].

In this chapter, we show that we can leverage active acoustic sensing to capture diverse information for manipulation tasks. To achieve this goal, we build a parallel gripper containing a piezoelectric speaker and contact microphone. We use the same hardware for the two ngers, taking advantage of the piezoelectric property that allows the transducer to be both an actuator and a sensor. We use this system to explore the capabilities of active acoustic sensing for object classi cation, grasping position estimation, pose estimation from internal structure, and contact state classi cation. Moving beyond isolated classi cation tasks, our key result is using this sensing modality to tackle a long-horizon manipulation task, i.e. the classic problem of peg insertion, learning a policy that successfully inserts a peg using active acoustic sensing as the only sensing modality.

4.2 Hardware Platform Overview

Our active acoustic sensing system operates by transmitting an acoustic waveform from an emitter on one gripper nger through a held object to a receiver on the opposing nger. The received signal is modulated by the object's unique acoustic resonance properties, providing rich information about its material composition, geometry, and how it is being held. This principle allows the gripper to sense an object's intrinsic and extrinsic properties through sound.

The hardware design builds upon previous work with several modi cations to enhance signal robustness. We utilize compact piezoelectric disks for both signal emission and reception. Functioning as contact microphones, these transducers are largely immune to ambient noise. They are mounted in custom 3D-printed housings that incorporate sorbothane pads to isolate the sensors from parasitic vibrations originating from the robot's body. Signal generation and acquisition are controlled by two Teensy 4.0 microcontrollers, which handle processing and communication with a host PC.

For a more detailed explanation of the hardware components, electronic circuit design, and the empirical analysis leading to these design choices, please refer to the original publication [15].

4.3 Sensing Procedure and Data Preprocessing

The sensing process involves generating an excitation on one nger while simultaneously recording the signal received by the other nger. During each sensing cycle, a sinusoidal signal is linearly swept from 20 Hz to 20 kHz for 1 second. Linear sweep signals increase in frequency at a constant rate over time, enabling us to focus on a range of frequencies during of ine analysis. Before training, we retain only the rst 42,000 points of the sweep (due to data corruption issues), effectively terminating the sweep at 19.029 kHz.

We convert the raw time domain data to the frequency domain using fast Fourier transform (FFT). We then use kernel principal component analysis (kernel PCA) with a cosine kernel to reduce the dimensionality of the data before model training. We further elaborate on the selection of

Figure 4.3: Time domain (left) and frequency domain (right) signals when the gripper is fully closed (left inset), as well as a spectrogram of the signal (right inset). The major peak around 8 kHz is consistent with the piezoelectric disk's reported resonant frequency.

principle components in the following section (Sec. 11.2). An example of the data when the gripper is fully closed is shown in Fig. 4.3 in both the time and frequency domain.

4.4 Estimating Object State and Properties with Acoustic Features

We investigate the types of information that our sensor can capture by training a model for each of the tasks outlined in Fig. 4.4 and 4.5. For all tasks, the dataset is divided into training, validation, and test sets. Although each subset may originate from multiple data collection sessions, we ensure that the test sets are derived from entirely separate sessions to account for distribution shifts between sessions, while the validation sets are sampled from the same datasets as the training set with a 9:1 split. For each task, we train a multi-layer perceptron (MLP) model of size (400, 250, 100) to do classi cation or regression.

4.4.1 Object Classi cation

We classify 9 rods of different materials and/or geometries (Fig. 4.4). We collected rods as close to 21 21 130cm as possible, however, we recognize that some of the rods differ slightly from these dimensions. We collect 100 and 25 training and test samples per object (900 and 225 total),

Figure 4.4: Setup for object type and grasping position experiments and examples of signals after FFT. All training data is collected with the rod resting on a table. We show generalization on test sets collected on new surfaces and object orientations.